Development and simulation of an automatic tool changer for an ABB robot

Paulo Jorge Leitão e Sousa

Master's degree dissertation

Orientation: Prof. Germano Veiga PhD

Co-Orientation: Luís Freitas Rocha PhD

U. PORTO

FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Master's in Mechanical Engineering

September 2019

Resumo

O objetivo desta dissertação consiste no desenvolvimento e teste de uma aplicação de troca de ferramentas automática que é utilizada para facilitar a programação de um robô da ABB. Esta troca de ferramentas compreende diferentes elementos, incluindo hardware e software.

No contexto desta dissertação foram desenvolvidos diferentes protótipos para um suporte para ferramentas que possa ser utilizado de acordo com esta aplicação e com uma troca de ferramentas específica. Os diferentes protótipos propostos foram então comparados entre si, e a sua utilização foi justificada consoante a aplicação.

Em seguida foi desenvolvida uma aplicação de troca de ferramentas, esta aplicação compreende 3 partes distintas, uma primeira parte onde o utilizador define as suas ferramentas, uma segunda parte onde o utilizador pode testar e ajustar os movimentos para pegar e largar as ferramentas, e a terceira fase onde o utilizador se encontra livre para utilizar as funções desenvolvidas. Para as duas primeiras etapas desta aplicação foram desenvolvidas HMIs para o teach pendant do robô de forma a facilitar a sua programação.

Depois, foram desenvolvidos vários protótipos para uma ferramenta especial que tem como objetivo pegar num blister e manipular peças separadamente. Esta ferramenta tem 2 grippers colocados a 90^o entre eles. Esta parte do trabalho tem como objetivo funcionar como exemplo para fazer futuras ferramentas que possam funcionar com este sistema de troca de ferramentas. A parte final do trabalho consiste em construir uma estação para simular os diferentes elementos construídos e aferir a sua utilidade para um programador.

A aplicação criada foi simulada e provou facilitar as necessidades de programação, os diferentes protótipos criados aumentam a flexibilidade e dão opções de escolha para suportes na maioria das estações. A HMI desenvolvida é bastante simples de utilizar.

Abstract

The objective of this dissertation was to build and test an automatic tool changer to facilitate the programming of an ABB Robot. This system comprehends different elements, such as the holder, the necessary tools, and an HMI to facilitate the programming.

In the context of this dissertation, several prototypes were developed for a tool holder that can be used with this application, and with a specific tool changer. They were then compared with each other and a brief explanation of where and how each of them could be used was made.

Afterward, it was developed a software application which can be divided into 3 different parts, a first part where the user defines his tools, a second part where the user can test and adjust the motion to pick and place the tools, and the third part where the user can directly use the developed routines. It was also developed for the first two parts an HMI for the robot's teach pendant, to facilitate the programming.

It was then developed several prototypes for a special tool to manipulate a blister and the tools within it. This tool uses two grippers which are placed 90° apart, as an example on how to develop future tools that can be used with this tool changer system. The final part of this was to build a station to demonstrate the correct usage of all the different elements involved and developed during this dissertation, and how they can facilitate the robot programmer's work.

The developed application was tested and proved to facilitate the programming necessities, the high number of developed prototypes also creates some flexibility on choices of what to use in most stations. The HMI was also quite simple to understand and use on normal work.

Acknowledgments

I'd like to thank my supervisor, professor Germano Veiga and my co-supervisor Engineer Luís Freitas Rocha for the chance to develop this dissertation, for all the holder demonstrated throughout this last months, and for the help solving some problems that arose.

I want to dedicate this dissertation to my mother for all the love and care in the world. I would also like to thank all of my family.

I want to thank all my friends for all the laughs, and love shared throughout this journey. I would also like to leave a special thanks to Leonor "The President" and Manel, my dissertation partners, JB, the best friend this man has had, Catarina "Spooky", my silly adventures sharer, and Teresa, the girl who gave me so much love and holder during this last few months.

Contents

Glossary

API	Application Programming Interface
AGI	Artificial General Intelligence
EOAT	End Of Arm Tooling
FEM	finite element method
HMI	Human Machine Interface
HTML	Hypertext Markup Language
RTC	Robotic Tool Changer
SDK	Software Development Kit
UR	Universal Robots

List of Figures

List of Tables

1 Introduction

1.1 Motivation

The western industries are under constant market pressure created from arising markets, and therefore most factories must modernize the production to allow these units to compete for a place in the global market.

To do this, companies implement robotic stations to reduce the cost of labor in their factories, as well as speed up production, allowing them to better compete and gain space in the business.

Another large problem in the current industries panorama is the ever-shifting society needs. Every day the necessity for customization and individualization of products increases, as most people use material possessions to express their individuality.

The modern industry must be able to respond effectively to these short-timed changes, being able to produce small amounts of very different products. This brings a very important concept, the *flexibility*.

A flexible production system is one that can quickly change its main structure, to respond to any fluctuations in the market requirements, and it is critical for any high-end factory to be as flexible as possible, and to allow a quick response to adversities and changes in the landscape of production[1].

1.2 Thesis Objectives

The main objective of this dissertation was to develop an automatic tool changer system that could work with different tools and different dispositions of tools, so it would be as flexible and scalable as possible.

The developed system was constituted by different components, that work to facilitate the application of this system by the final user. These parts are:

- A holder for the tools to be stored when they are not being used;
- Different programs and routines that can be called by the user to pick, place, and define the tools;
- Different programs and routines to help the user define how the robot should pick and place the defined tools,
- A human-machine interface, developed for the robot flex pendant, to allow a more direct and intuitive application of the created routines.

Another goal of this dissertation was the creation of a document that would provide an example of how a new tool could be developed and used with this tool changer system.

This dissertation also aimed to develop and simulate a very simple production line to pick and place specially designed products, so the developed application could be tested.

1.3 Project Methods

In the first part of this dissertation, it was analyzed the different aspects of the station where the tool changer was going to be implemented.

It was studied:

- The constructive aspects of the robot that would be using different tools. These aspects included, performance parameters, size of the robot, maximum payload and number of joints.
- The tool changer that was going to be coupled to the different tools to standardize the picking and placing of very different tools.

It was then analyzed the requirements for the tool changer holder, designed different prototypes for this system, followed by a comparative analysis of the different proposed prototypes.

Afterward, it was exposed the requirements for the programs that were necessary for the robot to perform the changing of the tools. At this point it was also defined what components should the HMI have.

It was then simultaneously developed the HMI and the programs inherent to the robot for picking, placing and defining tools, as well as the implementation of a flexible system that would allow the user to define how these tools should be picked.

Afterward, it was studied a real case of a component of a product line that required a new tool to be manipulated. This study was done with the objective of both creating a relevant tool for a real line, and exposing and explaining the different steps that need to be taken to develop a tool that can be used with this tool changer application

It was then designed different prototypes of this system, analyzed and planned different ways that this component could be manufactured. It was also made a brief structural analysis of one of these prototypes.

It was then hypothesized a station with different requirements, sensors, and actuators where all of the created components could be simulated and tested.

The final part of this dissertation was to use software that could simulate the hypothesized station and the implementation of the created program in this station.

1.4 Structure of the thesis

In the first chapter, the tool changer holder problem will be analyzed. To do this the different components involved in this structure will be studied. This is the tool changer and the robot used an automatic tool changer. It will then be proposed a series of different options for the tool holder, finally, it will be made a critical analysis of the various solutions.

The second chapters explains the creation of the program used to make the automatic tool changer, this is constituted by three sections, an initial section where the first part of the program where the user calibrates all the tools that are going to be used in the station, is explained and a second section where the second part of the program, where the user test the created routines is explained, and a third part where there is an explanation oh how the user utilizes freely the created routines.

The first two sections of the second chapter are constituted by sub sections that will be analyzed sequentially and separately. The first sub section is the creation of an HMI to facilitate the usage of the generated programs. The second sub section is the program that runs the robot.

In the third chapter, it is defined as a special tool that works as an example of how should a tool be developed for this system. This tool consists of a dual gripper adapter, this is an adapter that allows the usage of 2 grippers with a 90° angle between them. This chapter starts with detailed exposure of the problems involved with this tool. Then there is an overlook of the different components required to build this tool, as well as different possibilities for the main structure that is going to be used.

The last section consists of a simulation made to test and validate the different components created throughout this dissertation. It consists of the terminal end of a simple production line. This section is divided into 2 moments, in the first part the line is introduced and the requirements set, the second part explains how the application was used in this program.

2 State of the Art

2.1 Programming for robot controllers/teach pendants

Industrial robots are usually programmed trough domains specific languages, that are highly specialized in robotics tasks. Some manufacturers also include programming tools (API's) in a generic programming language that allow the customization of the robot controller, namely the development of advanced user interfaces. This section analyses some of these tools.

Yaskawa company is a Japanese manufacturer of servos, motion controllers, AC motor drives, switches, and industrial robots, which are denominated by "*MOTOMAN*", seen in figure 1.

Figure 1: Motoman Robot [2]

Motoplus SDK is a software package for the development of expansion modules (MotoPlus Apps) for MOTOMAN controllers. This enhances the functionality of MOTOMAN robot controllers through functions which are not available with the standard Inform motion command/Ladder, and allow the user the creation of communication protocols with other devices as well as advanced mathematical calculations. The expansions module is developed in a PC environment, using C programming language [2].

The main features of this program include:

- Integrated development environment
- MotoPlus library
- Collection of examples

The Advanced PP Customization SDK allows the creation of application-specific user interfaces for the teach pendant (PP). This system has several important characteristics such as the creation of an interface with accessible positioning of important information and control elements, a holder of a language switching function, a logbook function or several authorization levels to your application, and a device that allows the notification when certain variables reach certain values[2].

Another important manufacturer of robots its *fanuc*, which is an agglomerate of companies spread for different parts of the world. This company also provides their own teach pendant solutions and personalization options to their teach pendant called *iPendant*[3].

The *iPendant* allows the user to browse the internet allowing an easy search of solutions, it has a multi-window color interface to facilitate the quick reaching of information. The *iPendant*, also allows the development of an HTML graphic interface, with a customization function inside the pendant. This function provides a "panel wizard" to facilitate the development of this HMI's[3]. It is possible to seen in figure 2 an example an HMI created with this system.

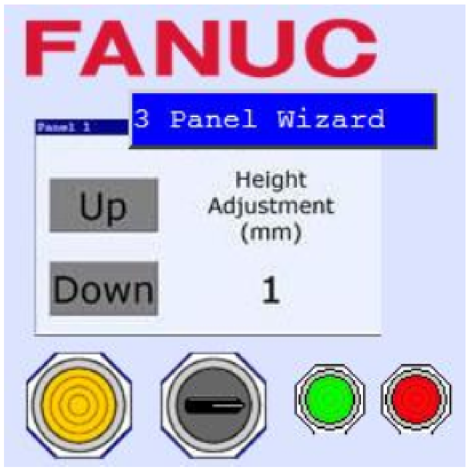


Figure 2: Panel Wizard Function [4]

Another industrial robot manufacturer is the ABB, this is a Swiss-Swedish multinational corporation. This company also provides teach pendants for robot interface, this systems pendants are denominated by "*flex pendant*". The software to develop HMI for this kind of pendant is called *"screenmaker"* [5].

2.2 Software Solutions for Tools Changer Systems

In this section, different existing options for tool changer software will be analyzed. The first solution is part of the RoboDK software bundle. RoboDK is an offline programming system and includes a software module to help on the programming of tool changer applications.

To use this software the user must specify the type of robot, the tool changer (male and female), and tool that will be used. Once all of these components have been downloaded to the program the user must attach the tool changer to the robot, define the end-effector using a tool center point imbibed in the program. This can be considered as a calibration stage of the tool changer program [6].

After this first stage, the user must import the tool rack holder to the user frame and define the targets where flange of the male tool changer should go to pick or place the robot, this is, he must define the location of the different tools that will be used. The user must also ensure the compatibility between the tool changer with the tool, and the tool changer with the robot that was chosen.

The user can now simulate in the program the picking and placing of the different tools, there are 3 different ways to do this, the ”simple way”, where the tools, tool changer, and stand are invisible, the ”half visual way” where only one of the tools is visible and the ”full animation way” where hole simulation is visible[7]. It is possible to see in figure 3, an example of this software.

The main advantage of this system is its flexibility, as it can work with most of the hardware available, and the biggest disadvantages is being an offline program.

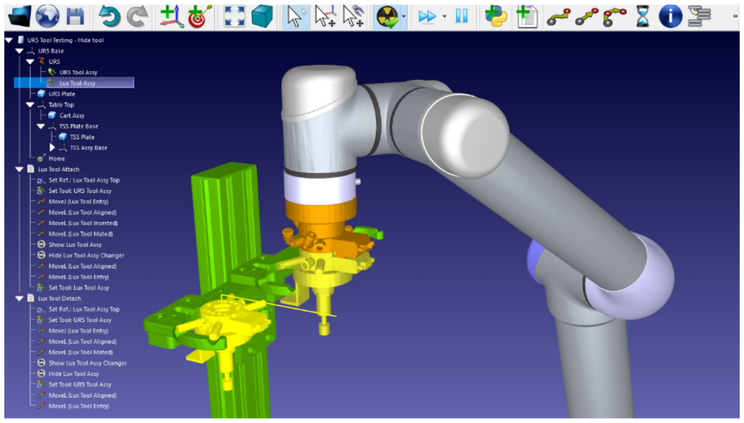


Figure 3: RoboDK Full Animation [6]

Another software solution is provided by Universal Robots, the QC-11 automatic tool changer, this solution also provides the hardware for its work, however, it only works with a selection of robots provided by this manufacturer [8].

The QC-11 Automatic Tool Changer works by using a pneumatically-actuated piston locking mechanism for coupling end-effectors to a UR robot arm. The user programs the robot arm to bring the Master (robot) and Tool (end-effector) sides together using a 2-position valve. The user supplies 60 to 80 psi to the lock or unlock air port to either latch or unlatch the tool changer. This solution is also fail-safe, which means that, in the case of a power shortage, the tool will remain attached to robot [8] [9].

2.3 Tool changer hardware

It is important to analyze the different available tool changers in the market, one of the most interesting systems in the market is the ” PAL RTC” [10], this is an automatic tool changer used in the pharmaceutical business to change syringes, and can be seen in figure 4.

This system can automatically detect the location and position of a certain syringe allowing the correct picking and placing of this tool, it also assures a correct position of this tool, this 2 features are very important in the pharmaceutical and medical business as most of the production referent to this sectors must be done in a cleanroom preventing the interference of humans [10].

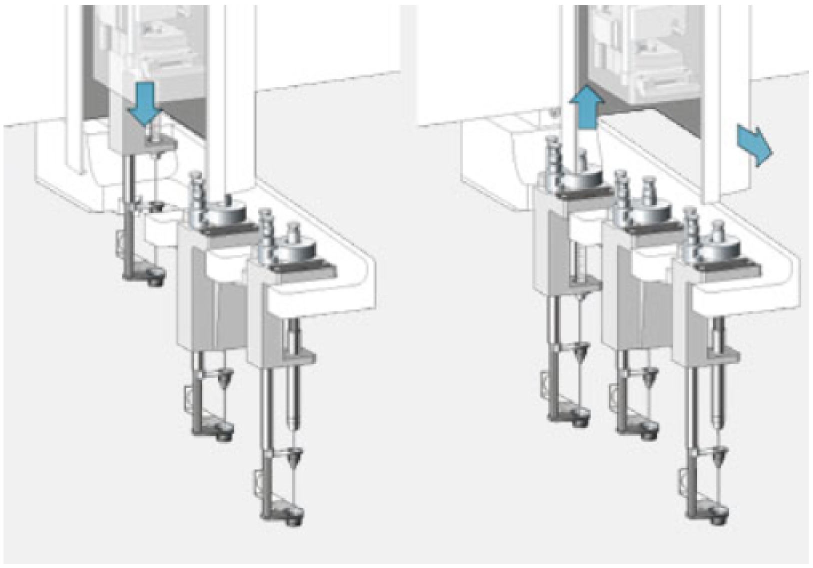


Figure 4: PAL RTC system [10]

It is possible to refer to other hardware tool changer systems, that can be used, this is the *AGI EOAT*, which is catheterized by its high rigidity and accuracy, and the *RSP* which has a very fast locking, however, it can hold up to 1250 kg of mass, which is a fairly high value [11]. It can also be used tool changer systems from *zimmer*, namely the WWR series which has a vast array of options to deal with different tool sizes [12], or tool changers from *Schunk*, namely the SWS series [13].

3 Tool Changer holder

After analyzing the state of the art, and the components available on the market, it is possible to start developing a tool changer that meets the system necessities, to do this it is first necessary to design a holder for all the tools involved in an automatic line.

Many different aspects need to be taken into consideration. Firstly it is necessary to know what kind of robot will be used, this necessity is because different robots have different kinds of movements and different kinds of performance capabilities.

It is also important to know what kind of tools are going to be used, this is due to the different geometry, weight, and center of mass that the different tools can have. A tool may remain balanced in particular holder but may be unbalanced in another, therefor different solutions should be presented.

Due to the variety of tools, there may be many valid solutions depending on which tool is necessary, the best option can even be a combination of solutions.

The tools will be connected with a tool changer, this is a universal system that works like a bridge between the tool and robot, this is, with same tool changer it is possible to connect a variety of tools and a variety of robots. This system greatly improves the line ability to respond to alterations in regards to requirements, increasing its flexibility.

3.1 ABB 2600 20kg 1.65m

The robot that is going to be used is an ABB 2600 20kg 1.65m, this robot has a maximum payload of 20 kg and a maximum reach of 1,65m. It is an articulated robot, which means that the robot moves by rotating its joints. This type of robots can have a minimum of 2 joints and a maximum of 10 joints [14], the ABB 2600 has 6 joints and can be seen in figure 5.

There are different aspects of the robot that can be analyzed, however, the most relevant technical data for this work is the performance parameters which evaluate the ability of the robot to perform a given movement.

Figure 5: ABB 2600 20kg 1.65m [15]

An important parameter is the pose accuracy, which gives information about the ability of the robot to reach the desired pose, expressing the deviation between a command pose and the mean of the attained poses when approaching the command pose from the same direction. It is divided into two parts [16]:

- the difference between a command pose and the barycentre of the cluster of attained points,i.e.positioning accuracy
- the difference between command angular orientation and the average of the attained angular orientation.

$$\Delta L = \sqrt{(\bar{x} - x_c)^2 + (\bar{y} - y_c)^2 + (\bar{z} - z_c)^2} \tag{1}$$

Where:

$$\bar{x} = \frac{1}{n} * \sum_{j=1}^{n} x_j \tag{2}$$

$$\bar{y} = \frac{1}{n} * \sum_{j=1}^{n} y_j \tag{3}$$

$$\bar{z} = \frac{1}{n} * \sum_{j=1}^{n} z_j \tag{4}$$

$$\tag{5}$$

Where:

- x,y,z are the coordinates of the barycentre of the cluster of points obtained after repeating the same pose n times;
- xc, yc, zc are the coordinates of the command pose;
- xj, yj, zj are the coordinates of the j- obtained poses

Therefore this parameter expresses the robot's ability to reach the desired pose, a bad pose accuracy (a high value of this parameter) implicates that the robot won't reach the desired pose.

Another important parameter to analyze is the pose repeatability, this parameter measures the ability of the robot to always reach the same position and expresses the closeness of agreement between the positions and orientations of the attained poses after n repeat visits to the same command pose in the same direction. For a given pose, the repeatability (r) is expressed by[16]:

- the value of r, which is the radius of the sphere whose center is the barycentre and which is calculated as seen in equation 6.
- the spread of the angles $\pm 3Sa, \pm 3Sb, \pm 3Sc$, about the mean values a, b, c, where Sa, Sb, Sc are the standard deviations:

$$r = \bar{x} + 3S_d \tag{6}$$

$$\bar{D} = \frac{1}{n} * \sum_{j=1}^{n} D_j \tag{7}$$

$$S_d = \sqrt{\frac{\sum_{j=1}^{n}(D_j - \bar{D})^2}{n-1}} \tag{8}$$

$$r_a = \pm 3S_a = \pm 3 * \sqrt{\frac{\sum_{j=1}^{n}(a_j - \bar{a})^2}{n-1}} \tag{9}$$

$$r_b = \pm 3S_b = \pm 3 * \sqrt{\frac{\sum_{j=1}^{n}(b_j - \bar{b})^2}{n-1}} \tag{10}$$

$$r_c = \pm 3S_c = \pm 3 * \sqrt{\frac{\sum_{j=1}^{n}(c_j - \bar{c})^2}{n-1}} \tag{11}$$

The procedure is the same as in 1. For each pose r and angular deviations ra,rb and rc are calculated.

A good value of pose repeatability and a bad one of pose accuracy implies that the robot will always reach the same pose, however this pose is not the desired one, on the other hand, a bad value of pose repeatability and a good value of pose accuracy implies that the robot will always be around the desired pose but always in different poses.

It is fairly simple to understand why these parameters are very relevant when analyzing the robot. If the robot reaches the tool that it is supposed to pick in a slightly different position, it may not be able to correctly pick it up, because all the connections that need to be made between said tool and the robot will be slightly off. Such a case may even damage the robot or the tool.

Other parameters can be considered such as the linear path repeatability, which regards the robot's ability to always go through the same linear path. The linear path accuracy, which evaluates the ability of the robot maneuver trough the desired linear path. The pose stabilization time, that measures the time between the robot reaching the desired pose, and the moment the robot stops in that pose. This parameter is relevant to the analysis, for example, a robot with a bad linear path accuracy may damage a tool while trying to take it from a holder that has a linear slot.

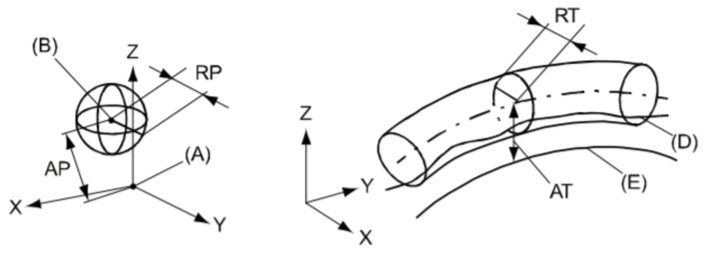
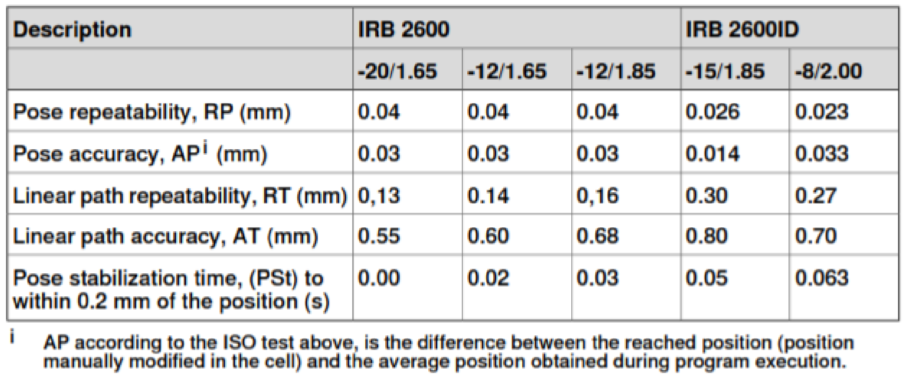


Figure 6: Illustration of the performance parameters [15]

Table 1: Performance parameters [15]

Pos	Description	Pos	Description
A	Programmed position	E	Programmed path
B	Mean position at program execution	D	Actual path at program execution
AP	Mean distance from pro-grammed position	AT	Max deviation from E to average path
RP	Tolerance of position B at re-peated positioning	RT	Tolerance of the path at repeated program execution

Table 2: Values of the performance parameters [15]



Description	IRB 2600			IRB 2600ID	
	-20/1.65	-12/1.65	-12/1.85	-15/1.85	-8/2.00
Pose repeatability, RP (mm)	0.04	0.04	0.04	0.026	0.023
Pose accuracy, AP[i] (mm)	0.03	0.03	0.03	0.014	0.033
Linear path repeatability, RT (mm)	0,13	0.14	0,16	0.30	0.27
Linear path accuracy, AT (mm)	0.55	0.60	0.68	0.80	0.70
Pose stabilization time, (PSt) to within 0.2 mm of the position (s)	0.00	0.02	0.03	0.05	0.063

i AP according to the ISO test above, is the difference between the reached position (position manually modified in the cell) and the average position obtained during program execution.

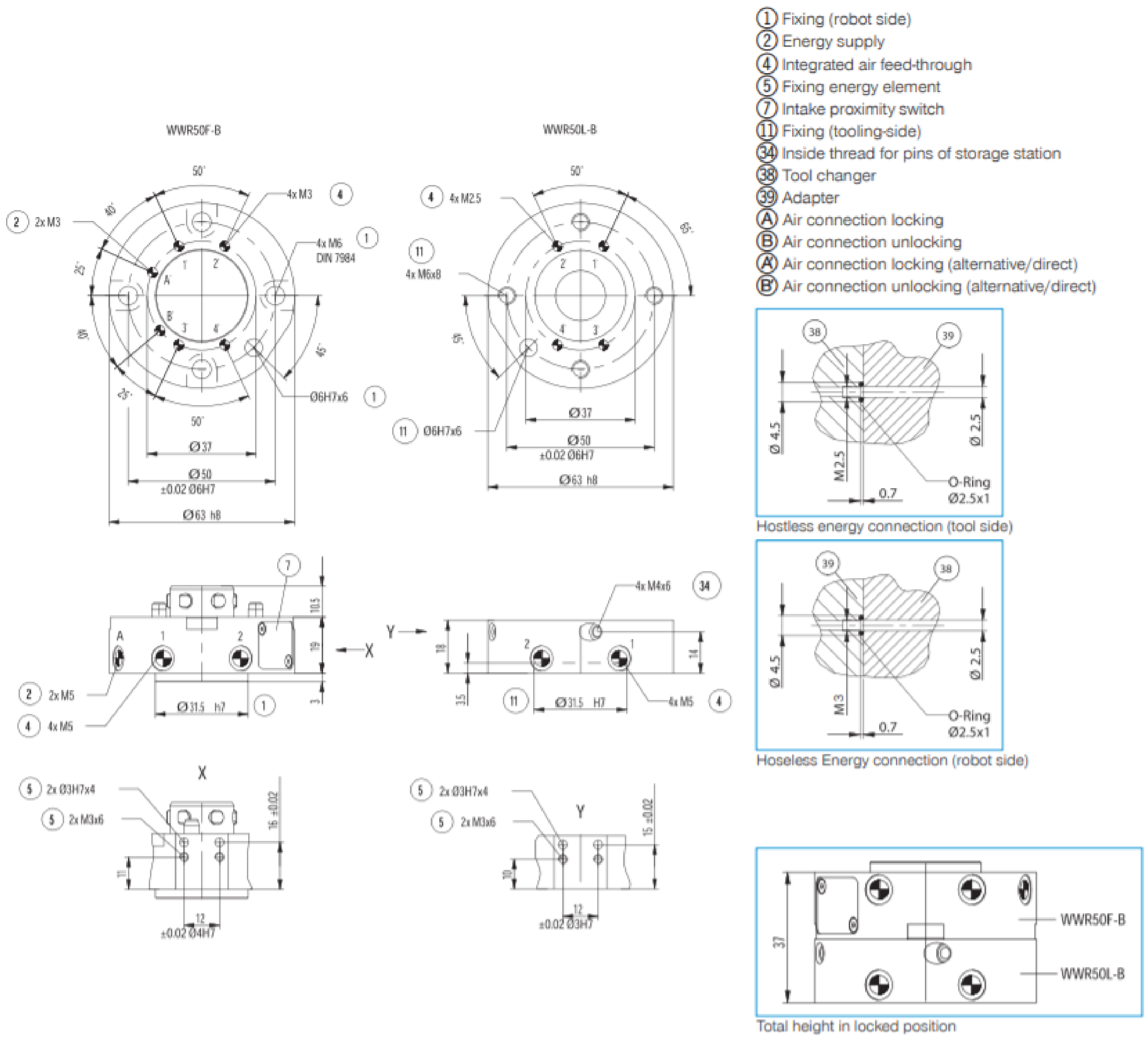
Analyzing table 1, table 2, and figure 6 it is possible to see that the pose accuracy for this robot is of 0.03 mm and the pose repeatability is 0.04mm, therefore, when designing, the holder for the tool changer, any slots or paths that the robot will go to, must have a minimum of 0.04mm of a gap between them, thus assuring that the robot will always succeed in reaching its destination without bumping into anything.

3.2 WWR50

As it was mentioned in the previous section, the tools that are going to be used are coupled with a tool changer, this tool changer is a WWR50 from *Zimmer*, and it is composed by two different parts, the first one is a female tool changer which holds the tool itself, the second is a male tool changer which is connected with the robot.

The female tool changer is connected with the tool using screws, it is possible to see in figure 7 the main dimensions of this tool. This system has a normalized set of holes which are identified in figure 7 by the number 11, which allows a physical non-permanent connection with the tool.

The male tool changer, is the part connected with the fist of the robot through a normalized set of screws identified in figure 7, by the number 1



① Fixing (robot side)
② Energy supply
④ Integrated air feed-through
⑤ Fixing energy element
⑦ Intake proximity switch
⑪ Fixing (tooling-side)
㉞ Inside thread for pins of storage station
㊳ Tool changer
㊴ Adapter
Ⓐ Air connection locking
Ⓑ Air connection unlocking
Ⓐ' Air connection locking (alternative/direct)
Ⓑ' Air connection unlocking (alternative/direct)
WWR50F-B
WWR50L-B
Hostless energy connection (tool side)
Hoseless Energy connection (robot side)
Total height in locked position
O-Ring Ø2.5x1
Ø 4.5
Ø 2.5
M2.5
M3
0.7
2x M3
4x M3
4x M6 DIN 7984
4x M2.5
4x M6x8
Ø6H7x6
Ø37
Ø50 ±0.02 Ø6H7
Ø63 h8
2x M5
4x M5
4x M4x6
Ø31.5 h7
Ø31.5 H7
2x Ø3H7x4
2x M3x6
±0.02 Ø4H7
±0.02 Ø3H7
37
X
Y

Figure 7: Main dimensions of the female tool changer [12]

The tool changer uses pneumatic power to make the connection between both its parts. The male tool changer possesses a special set of spheres, that are used to make the connection between both parts of the tool changer.

When the user wants to connect the female part with the male part, it inserts the flange the male within the female, afterward, a double effect pneumatic cylinder pushes the set of metallic spheres inside especially dimensioned slots in both the male and female parts of the tool changer, allowing a strong grip force between both parts of this component [12].

This tool changer is also fail-safe, this means that in the event of lack of power both sides of the tool changer will remain coupled, this is due to the high tightening strength and a very small gap between the metallic spheres. To separate the 2 parts, the pneumatic cylinder must be moved it in the opposite direction breaking the grip between its parts.

It is possible to see the coupling movement in figures 8, 9, 10, and 11. In figure 8 the male and female tool changer are touching each other but are not connected. In figure 9 the male pneumatic cylinder is activated pushing the metallic spheres inside the female tool changer, in figure 10 the tool changer is fully coupled and ready to use. In the last figure, we see the returning movement of the cylinder and the releasing of the connection.

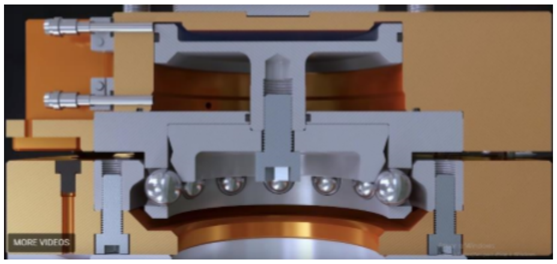


Figure 8: Male and female tool changer before connection

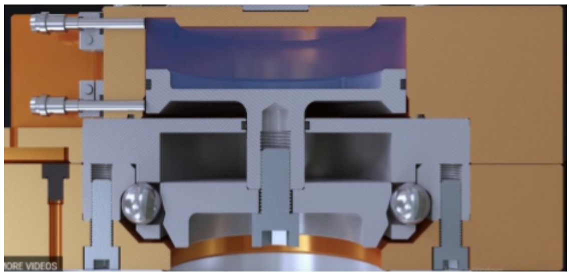


Figure 9: Pneumatic cylinder advancing in to the female tool changer

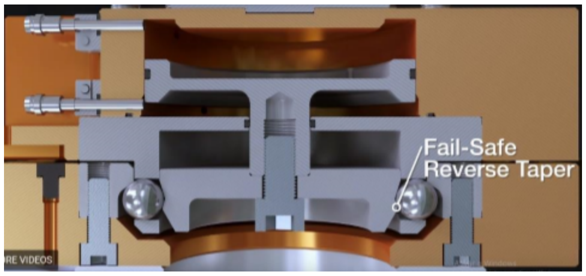


Figure 10: Fail Safe mechanism

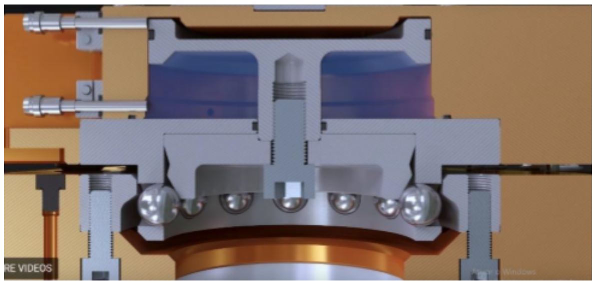


Figure 11: Returning effect

The WWR50 also has set of holes, identified in figure 7 as 34, to insert components that will allow the storage of the combination of the female tool changer and tool, in an appropriate storage station [12].

We can see in figure 12 a 3D representation of the female tool changer with pins attached, this tool changer can then be placed in a holder using these pins.

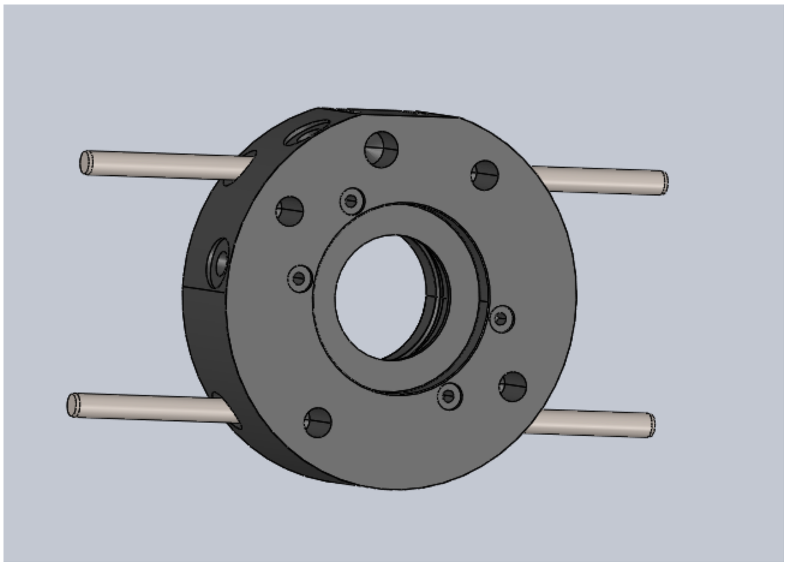


Figure 12: Solidworks view of the female tool changer, with pins attached

This example was generated using *Solidworks*, one of the main tools of this project. We can see that the pins that were used were an example, it is possible to connect different devices to make a more suiting fixation system.

3.3 Tool Changer holder Objectives

Before prototyping a solution for the tool changer holder, it is first necessary to analyze the different objectives of this system and how they should be balanced between each other.

The most relevant aspect to consider is the stability of the tool in the holder, it is necessary to guarantee that, while in the holder, the tool will not have any kind of movement.

It is also important to take into consideration the type of movements that the robot needs to make to manipulate the tool, as well as, the tolerance of these movements. It is preferable smaller movements and systems with a larger dimension tolerance.

The price of the solution is an important parameter to consider when dimensioning the holder. A simpler holder with less and cheaper parts will be preferable when compared to a more complex one.

Another very important concept to consider when designing the holder, is the of space the tool will require inside the industrial area. The space usage must be minimized as much as possible.

3.4 Tool changer holder Concepts

3.4.1 Vertical holder

It is now possible to create an initial design for the holder. First, it was analyzed the proposed holder by *Zimmer*, as standard holder for this kind of tool changers. This holder is a vertical one, which means that once the tool is placed in the holder the working end will be vertically located. This can be illustrated in figure 13.

The holder has 2 grooves parallel to each other, identified in figure 13 by the number 1, and requires the usage of 4 pins attached to the female tool changer. At the end of these grooves that are 2 couplings where the pins will be fixed, this is made of flexible metal and have a stabilization function for the female tool changer. In figure 13 this is identified by the number 2.

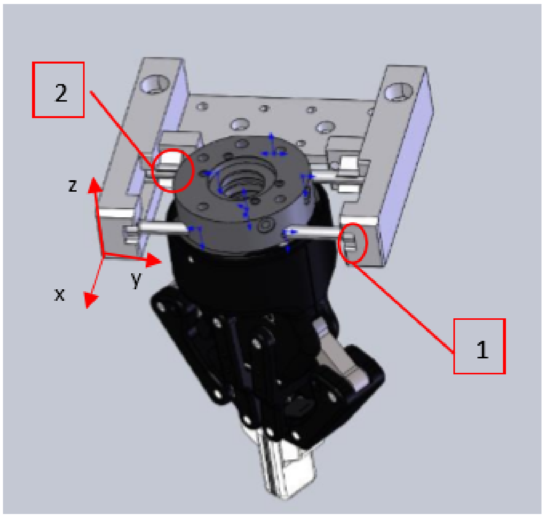
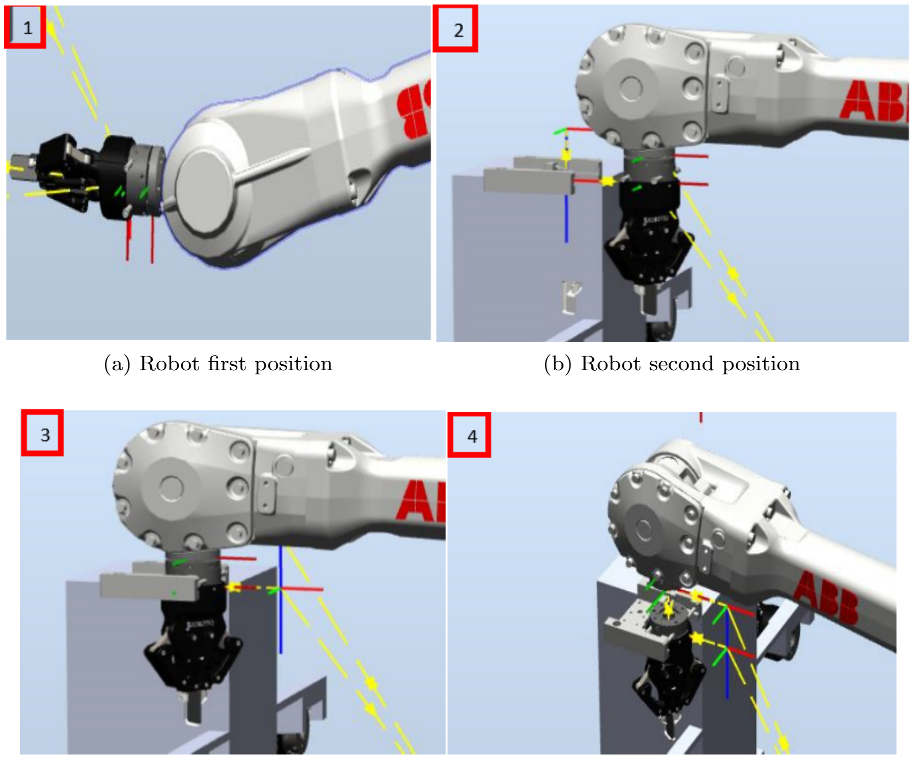


Figure 13: Standard holder

Although the robot will be placed vertically, the movement to a place of the tool in the frame is horizontal, that is, the movement must be made according to the axis x, and not the axis z.

It is possible to see in figure 14 how the robot should move itself to place the tool. In figure 14 a, the robot is just holding the tool, in figure 14. b the robot has faced the tool downwards and prepares to enter the holder. In figure 14. c, the tool has entered the holder, the movement between figures 14. b and 14. c must be linear as the grooves are also linear. In figure 14. d the tool has been placed and the robot is now free to return to is original position.



(a) Robot first position (b) Robot second position

(c) Robot third position (d) Robot fourth position

Figure 14: Movement made to place the tool

Notice that the values of the displacements of the robot were not mentioned, which is due to the variety of possible tools that can be used in a holder. A large tool may require the robot to be placed fairly far from the holder before it initiates is linear movement, this is, the tool in figure 14. b, so it can safely be placed. A smaller tool may not require such additional effort.

It is also important to refer to the linear movement present using this holder. As was said before, certain performance parameters must be respected to obtain a safe usage of the robot, a large linear movement may create some difficulty, however, and due to the fairly good ability of the robot, it should not be problematic.

The removing of the tool from the holder is in all similar to the movement to place the tool, except is made in the opposite direction, it goes from d to a, being the tool picked in the part of the movement referenced by c.

Vertical holder has some advantageous regarding a horizontal one, as it allows a very stable position for the tool due to the existence of 4 holder points equally divided in space. The programming of the robot also does not raise many concerns, as it doesn't require any awkward movement from the robot, and for must tools the spectrum of movement being fairly small. The only exception is when the holder is placed very high within the rack, this disposition will cause some difficulty for the movements of the joints of the robot.

There are some disadvantages regarding an horizontal holder, they mostly lie within the occupation of factory space. Due to the tool being placed vertically the space beneath them can not be utilized to save more tools, as the larger end of the robot may collide with the standing tools.

Using the vertical arrangement every tool must be stored side by side. The space usage could be improved if there is also an horizontal holder to store tools, allowing both storage side by side, as well as, tool beneath tool.

3.4.2 Horizontal solution 1

It was proposed an horizontal solution to allow a more flexible occupation of the space of the multi-tool product line. This solution uses four m4 pins connected with the female tool changer in the holes identified in figure 7 with number 34 to allow the stabilization of the tool. The main frame as a rectangular projection with a slightly smaller gap than the distance between the pins of the female tool changer.

In the main frame there are 4 m6 holes, where screws can be inserted to allow a connection between the holder and an eventual rack, where the solution will be applied.

The tool changer will tightly slide inside this gap. Due to the small difference between the gap in the frame and the gap between the pins the female tool changer will be stuck in the frame and remain incapable of rotating.

In the main frame there are also special couplings at the end of the gap, which are the same used in the vertical holder, and have the function to increase the stability and making difficult an eventual translation of the pins. This structural can be seen in figures 15, 16 and 17

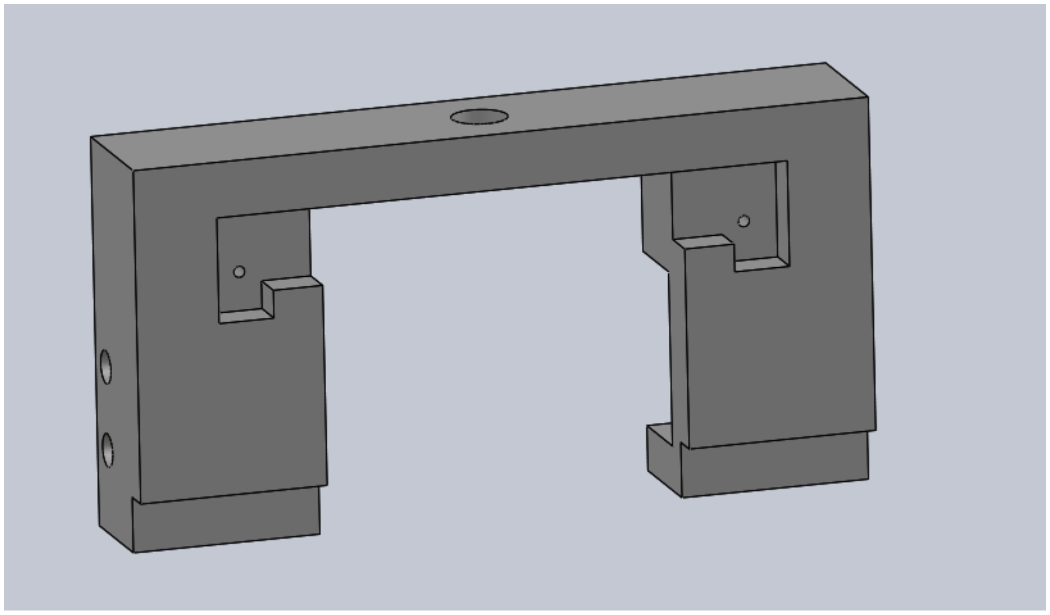


Figure 15: Horizontal Solution without tool

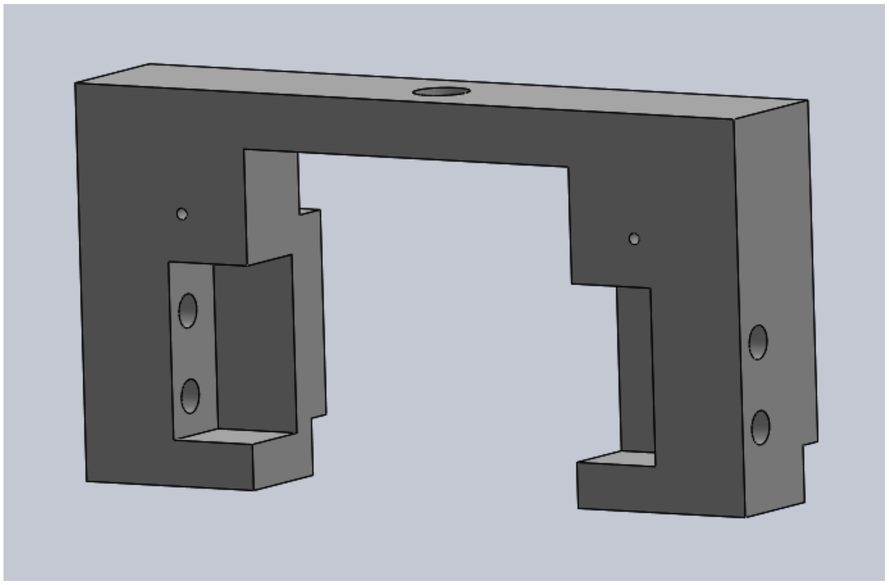


Figure 16: Horizontal Solution without tool 2

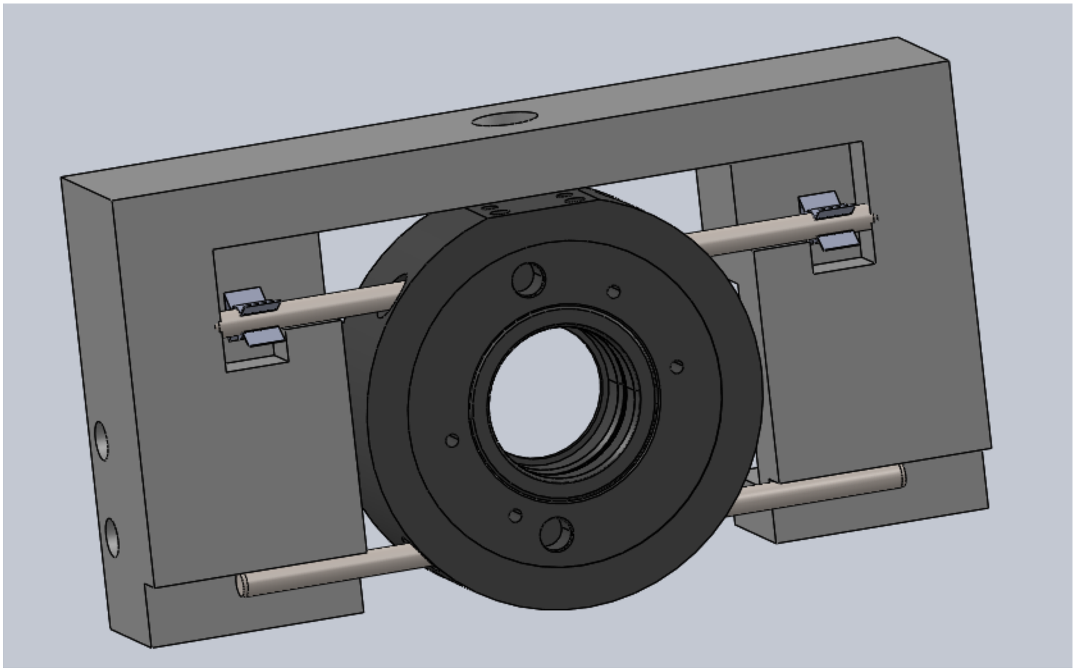


Figure 17: Horizontal Solution with tool

Analyzing figure 17, it is possible to see a small circular hole at the top of the frame, in this hole, there will be inserted a proximity sensor which then is connected with the controller giving information about the existence or absence of a tool in this holder.

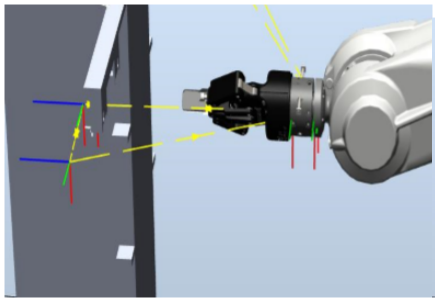
This solution offers an obvious advantage regarding space occupation, especially when combined with the vertical holder, as the tools can be more tightly packed. It also increases the flexibility of the tools as it will allow to a more broad range of option when deciding where should each of the tools be stored.

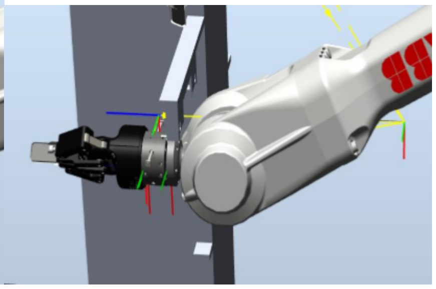
There may, however, be some problems regarding the tool stability when considering a very heavy tool due to the possibility of the tool rotating or sliding when stored in the holder.

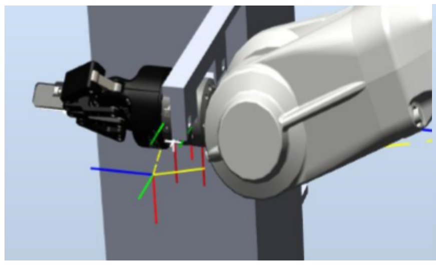
The figure 18 illustrates the movement of the robot to place the tool.

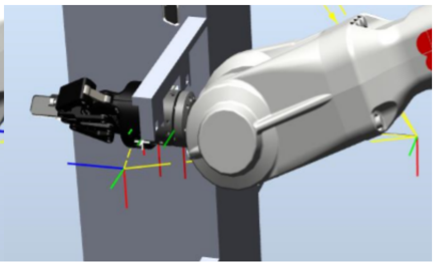
In figure 18. a the robot is just standing in front of the holder, afterward, it dislocates himself to a position slightly more advanced than the location of the holder, this position is illustrated by figure 18. b.

Once the robot reaches this position, it moves upwards to the location where the pins are at exactly the same level as the gap, afterward it moves inside this gap until it reaches the metallic couplings. Once that is done, it releases the tool being free to return to its original position.


(a) Robot first position


(b) Robot second position


(c) Robot third position


(d) Robot fourth position

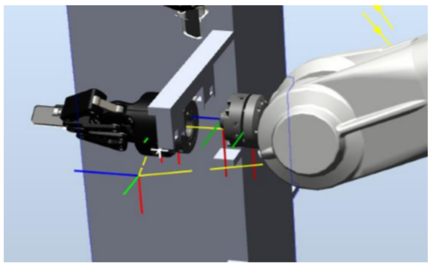

(e) Robot fifth position

Figure 18: Sequence of figures of the robot placing the tool

The retrieving of tools is in all similar to this type of movement except for being made in the opposite direction, as it was for the standard holder.

3.4.3 Horizontal holder solution 2

Both solutions use 4 pins. It may be interesting for the development of a solution that doesn't require such a large number of pins. The interest in using fewer pins is because these tools will be used to manipulate objects in a real workspace, therefore, these pins can bump into objects, making the programming of the movement of the robot more complex.

The second proposed solution has the objective of eliminating the necessity of 4 pins, using instead only 2. Instead of using the 4 previously mentioned pins there were only used two, each of them had, in its further end, a square nut attached. The holder had two special dimensioned slots that were slightly smaller than the side of the square nut. There are also the two couplings at the end of the main frame, again, destined to increase the stability of the tool changer once this has been placed there.

To place the tool in the holder, the pins and square nuts slide inside the slots in the holder, all the way until it reaches the couplings. At this position, the square nuts avoid the rotation of the tool due to the tight gap between them and the slots in the holder, the couplings would again help stabilize the tool and prevent any kind of translation of the system.

It is possible to see in figure 19 this holder.

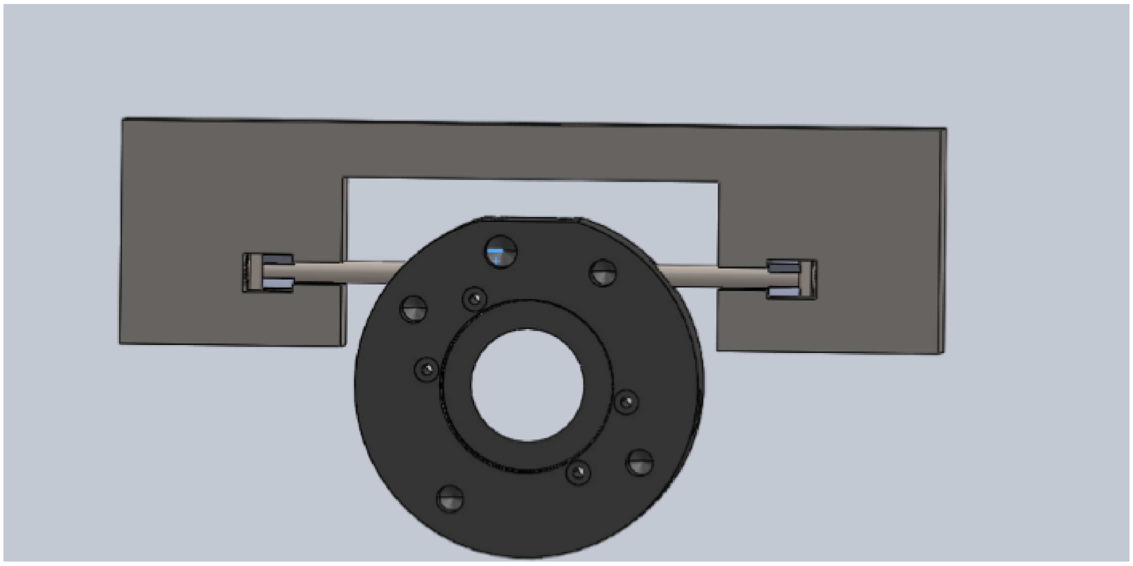


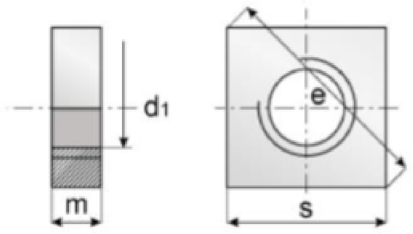
Figure 19: Second horizontal solution with tool

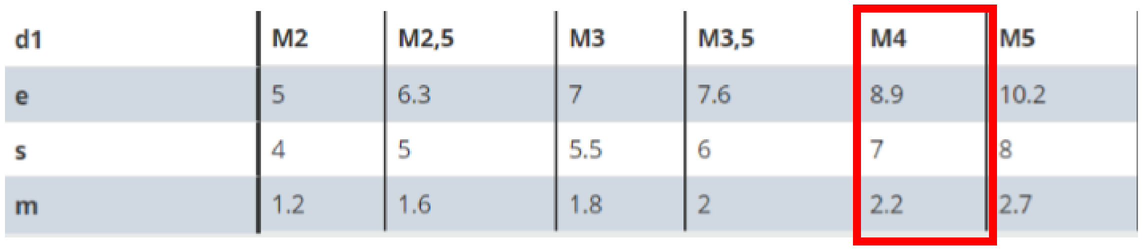
The movement of the robot to pick and place the tool is in all similar to the movement presented in the first solution.

It is now necessary to define what kind of square nut will be used.

Analyzing table 3, which are the nuts normalized according to DIN 562, it is possible to see the existence of only 4 different types of nuts, the choice of the nut that was going to be used was made considering the diameter of the pin, this is of 4mm, therefor the nut will be use is a m4.

Table 3: Square Nut - DIN 562





d1	M2	M2,5	M3	M3,5	M4	M5
e	5	6.3	7	7.6	8.9	10.2
s	4	5	5.5	6	7	8
m	1.2	1.6	1.8	2	2.2	2.7

The minimum size of the gap, corresponds to the value ”s” in this case 7mm, however and due to the performance parameters, it is necessary to have a larger gap to safely place the tool in position. The minimum value of this difference is of 7.04m, however, it is important to have a safety coefficient when dimensioning these slots.

Considering a safety coefficient of 10, the gap between the nut and the slots is 0.4mm, and the total gap is 7.4mm. The difference between the top of the nut and the top of the slot will allow a small rotation of the nut, and therefore, a small rotation of the tool. With this safety coefficient of 10, there will be a rotation of 3.38°.

If it is only used a safety coefficient 2 the rotation angle would be of 0.66°, this kind of rotation should be taken into account when choosing and programming this kind of holder for the tool changer, however, it is not a prohibitive factor when choosing this holder.

3.4.4 Horizontal solution 3

The third solution has similarities with the second solution, the holder has a similar shape and the tool changer uses the combination two pins and square nuts, however, this solution takes one step further in the stabilization and centering of the tool changer.

In this case, instead of using normal square nuts, it was employed as a special designed square nut with a cavity within one of its faces, the one contrary to the center of the tool changer. This cavity has a circular shape, and its design to create a self-centered mechanism.

To do this there are two pins in the holder, that have a complementary shape of the cavity's in the nuts and are connected to the main frame of the holder trough springs, these pins are located at the end of the slot.

The movement of placement of the tool in the holder is in all similar to both the horizontal solutions, it also exists the same coupling at the end of the slot where the pins slide in to.

In this solution, however, when the robot reaches the end of the slot where the special pins are located, there is a connection made between the nuts and the pins. Due to the springs having the same strength from each side the tool changer will be centered in the main frame of the holder. This solution can be seen in figure 20 and figure 21

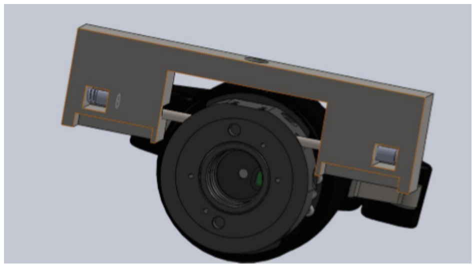


Figure 20: Self centrage view

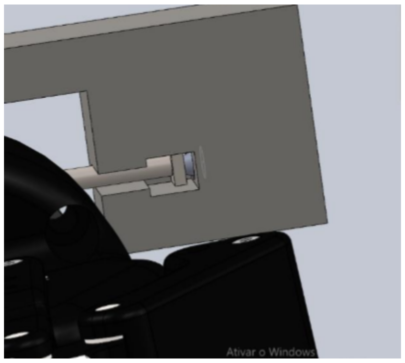


Figure 21: Self centrage view detailed

This solution has higher stability than both the previously mentioned solutions as it has an additional force keeping it in place regarding all axis. That additional strength also helps to prevent any eventual rotation. This solution also has an obvious advantage regarding the positioning of the tool, due to its self centering, even if the robot misplaces the tool changer by more than a few millimeters the tool will return to its place which increases the flexibility and allows this tool changer to be used with a less accurate robot.

The most obvious advantage given by this structure is the self centering mechanism, as it was previously said that the robot has some performance parameters and the position where the tool is left may not always be reached in the same way, therefore having this mechanism will allow the holder to re-position the tool, allowing this to be always in the right location.

These solutions are composed of a high number of different components, most of them not normalized, this fact will severely increase the total price of the system, as they will have to be individually made. If we consider this structure made of aluminum the price of machining will make the holder even more expensive.

3.5 Critical Analysis

It is now important to analyze the different constructive solutions and compare them to each other.

The main guidelines to chose the holder are referenced in the section ”tool changer holder objectives”, it will be considered as the main objectives, the stability of the tool in the holder, the type and complexity of the movements of the robot to reach the holder, the price of the holder and amount of space required when the tool is placed in the holder.

3.5.1 Tool Stabilization

In terms of stabilization of the tool, it is fairly simple that the vertical solution will be the best option, as there is no rotation of the tool in the robot, as well as very good weight distribution in the holder. All of the horizontal solutions score similarly in this aspect, even Although the solution number 3 is slightly better due to the additional stability force given by the springs.

3.5.2 Movements difficulties

The second aspect to analyze is the type and complexity of the movements to pick and place the tools in the holder. The horizontal solution applies a shorter movement range that the vertical solution making these types of solutions better in comparison to the vertical one. The solution number 3, allows a better movement tolerance in comparison to the other horizontal solutions, as the tool will always be centered even if the robot fails to place the tool in the exact position.

3.5.3 Construction Process

The third aspect to consider is the price of the different solutions, different aspects affect this price, but the most important variables are the material of the holder, the number of none normalized parts, and the number and complexity of the machining operations that will be necessary to perform.

There are two main possibilities for the materials of the holder, the first and most obvious is making the holder in aluminum, as it is a cheap and resistant material, that can be machined from a block of material. The second option is to make this holder using an additive process, such as 3D printing.

The first option is significantly better in terms of material resistance and speed of productions, however, the price will be significantly higher. Making the holder in aluminum is, therefore, a good option for a large scale production, the 3D printing is a cheaper process, but the final holder may not be as resistant, however, such affirmation would require a more extensive study and the use of FEM software, the production using 3D printing can also be significantly slower, however, and considering this holders are prototypes, it is recommended to use 3D printing.

It will be considered from now on, that the holders will be 3D printed. This implies that the price of machining won't exist, therefore the price of the solution will be mainly influenced by the amount of none normalized parts required for the holder.

The most expensive solution is, therefore, the horizontal solution number three, to the high number of parts, and the two non-normalized square nuts applied. The two first horizontal solutions are expected to be cheaper options, followed by the vertical solutions, all with a somewhat similar price. Using "*3Dhubs*" website it was assessed the price of both machinings the holder for the horizontal solutions and 3D printing it, the results can be seen in table 4. Notice that this is the price of machining/3D printing one main holder, the final price will be enlarged by the extra parts and will be reduced if it decided to do a larger series.

Table 4: Main holder prices

Solution	3D Printing price(euros)	Machining price(euros)
Horizontal Solution One	78,64	138,89
Horizontal Solution Two	48,43	98,66
Horizontal Solution Three	50.88	102,57

3.5.4 Space Occupation

The final aspect to consider is the occupation of the space that the tools will have when placed at the holders. In this aspect, the vertical solution is not the best option due to the high need for space, however when combined with any of the horizontal solutions, the vertical solution works very well.

In this aspect is simple to understand that the best option is a combination of solution and not a single solution.

4 Tool Changer Programming

4.1 Application Objectives

The objective of a tool changer application is that given the location of a tool, the robot should be able to pick and place it at any time automatically. To do this, the user must first teach the robot where the tools are located moving the robot to the tool position and saving this position in the robotic system.

The developed application should be as universal and scalable as possible, in a way that if a completely different disposition of the holders were used, as well as a higher number of tools, the robot could still pick them up, given a previous calibration of the system.

The program should also allow the user to select the tool that he wants to pick or place at any time.

This application can be used in three separate moments:

- A first moment, where the user teaches the location of all the tools that are going to be used in the project to the robot.
- A second moment, where the movements of the robots are defined, tested, and if needs be, adjusted, to assure a safe movement of the robot.
- A third moment, where the user can pick or place the tools that were previously defined according to the needs of his project.

It is also important to remember the different possibilities for the tool changer holder as they have different kinds of movements to pick and place the tool.

The holder that will be initially considered is the vertical one, however, the movements of the robot can be readjusted inside the program to accommodate most of the holders.

4.2 Tools and Software

To develop a tool changer application there is the necessity to consider different software, firstly we need to consider the language of the robot, then the software that was used to build the HMI and finally the program to simulate the system.

The robot used in this project was an ABB, these kind of robots have a physical console that can work as an HMI to help manipulate and program the robot, which can be seen in figure 22. This company also offers an array of programs to help define applications for this robot.

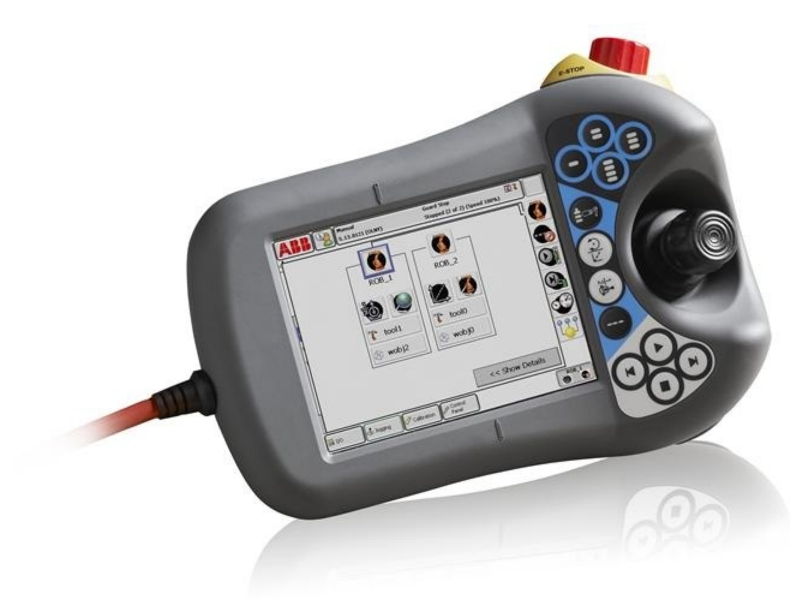


Figure 22: Flex Pendant [5]

The ABB programming language is named *RAPID*, this was the language used to program the robot, which is quite vast and powerful, therefore there is a necessity of explaining some of the concepts inherent to this program.

ABB provides a program named *Robotstudio*, this is where most of the robot applications are programmed, it includes a graphical station where is possible to simulate the real movements of the robot, it also has a series of add-ins that facilitate the full simulation of a real station.

One of the add-ins inside robotstudio is the *screenmaker*, this is a graphical interface, that allows the user to develop applications for the flex pendant in a more intuitive environment.

The creation of a graphical interface for an ABB robot must be done in both the RAPID language and the screenmaker. These two components must communicate between them to share variables and signals, for example, when the user changes a variable or sends a signal through the screenmaker it changes a value in the rapid program, this means the programs are running separately but communicating with one another.

4.3 Rapid

RAPID is the language used in the programming of the different routines that were used in this work, therefore it is important to understand the basic concepts inherent to this programming language, as well as understand how can robotstudio be used to simplify the programming procedure.

The first concept that will be analyzed is the main control mechanisms, which are the different types of routines. It is important to understand the goal of each routine, how they work, and when should they be used.

A major part of the program will be to pick and place tools, such elements are located at different points in space, therefore it will be examined the movement data essential to the RAPID programs.

Lastly, it will be studied the different coordinate systems that RAPID uses to locate and define the different movements of the robot.

4.3.1 Main Control Mechanisms

Procedures The main function type used in rapid is a procedure, this is a piece of code that runs sequentially, and don't return any value [17]. These functions are used to define paths, which have significant importance within any rapid program.

Functions A function is a control mechanism that given an input returns a determined value, calculated always in the same fashion [17]. It is mostly used when there is a particular expression that needs to be repeated several times during the application of the program.

Traps Trap routines are control mechanisms used to deal with interruptions. They are used as a response to a given input, allowing a program to be immediately interrupted to perform a given task. This kind of routines cannot be directly called[17].

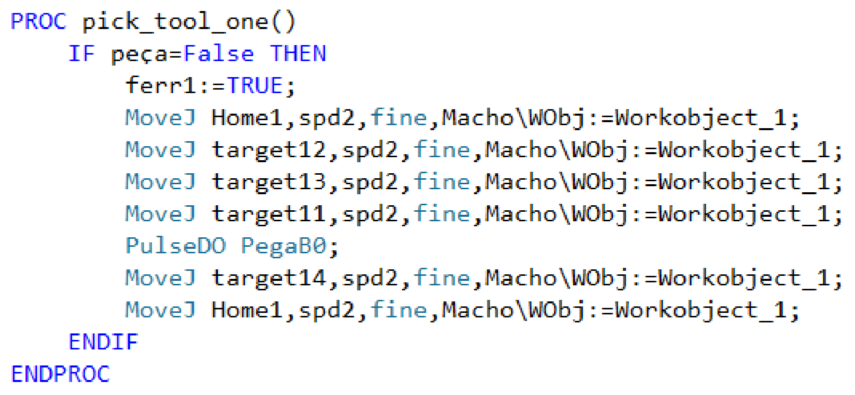
There are 4 different ways to interrupt the rapid program to give way to the trap routine:

- Using signals, which can be digital or analogical, output or inputs;
- From an error, this is, a trap will run when a certain error occurs;
- An interrupt can be fired every few seconds, cyclically;
- A trap routine can be fired every time a variable changes.

4.3.2 Movement Data

Paths A path is a particular type of procedure that defines the way the robot should move between points. It uses "move" instructions to define how the robot should go to a given destination point.

To define a path it is necessary to choose what kind of movement the robot will make. The movements can be either linear (instruction "moveL") where the joints of the robot



```
PROC pick_tool_one()
    IF peça=False THEN
        ferr1:=TRUE;
        MoveJ Home1,spd2,fine,Macho\WObj:=Workobject_1;
        MoveJ target12,spd2,fine,Macho\WObj:=Workobject_1;
        MoveJ target13,spd2,fine,Macho\WObj:=Workobject_1;
        MoveJ target11,spd2,fine,Macho\WObj:=Workobject_1;
        PulseDO PegaB0;
        MoveJ target14,spd2,fine,Macho\WObj:=Workobject_1;
        MoveJ Home1,spd2,fine,Macho\WObj:=Workobject_1;
    ENDIF
ENDPROC
```

Figure 23: Path Example

rotate to allow a linear movement of the fist, or a joint kind of movement, where if the path is linear or not is irrelevant (instruction "moveJ").

It is necessary to characterize the movement speed, this is defined by a "v" with a number in front of it, this number determines the speed of the robot.

Another parameter to specify a path is the zone, this corresponds to the location where the robot should stop moving towards the target inside the current move instruction and initiates the movement towards the target inside the next move instruction. It is defined by a z and a number in front of it, or a "fine" if the user wants the robot to reach the exact destination point.

Finally, it is necessary to characterize the tool that the robot is using and the referential that the coordinates are defined.

In figure 23 it is possible to see the different components of a move instructions. First, it is defined as the type of movement, followed by the location where the robot should move, then the zone, then the tool and the coordinate system.

In a path procedure, some instructions define actions that the robot should make at a given target. It is possible to see in figure 23 an example of this case. After the move instruction to "target11" there is a function labeled "PulseDO PickX0", this instructs the robot to pulse a digital output labeled "Pick0" once it finishes the move instruction to target 11.

Targets The point to where the robot should move is defined by targets. These are data elements that define a given position and a given orientation that the robot can visit. It is determined about a workobject, and also determines the robot configuration.

The position is the spatial location within a coordinate system that the robot should visit. It is defined by the coordinates x, y, and z.

The orientation is the angles of the tool frame when placed in the defined target, it is expressed in quaternions and corresponds to the angles that the fist should have when placed at the target.

The configuration is a confdata type of vector that defines how the joints of the robot should be placed when reaching is a target:

- the first value of the vector represents the quadrant of the first axis;
- the second value of the vector represents the quadrant of the fourth axis;
- the third value of the vector represents the quadrant of the sixth axis;
- the fourth value of the vector represents the quadrant of virtual axes, this is used to represent the center of the robot fist regarding the other axis.

The quadrants start getting enumerated from 0, representing the first positive quadrant, then they are enumerated counter-clockwise this means that the second positive quadrant is represented by the number one, and the third is represented by the number two [17].

The negative quadrants start at minus one and are enumerated clockwise diminishing, so minus one is the first negative quadrant, and minus two is the second negative quadrant, and so on.

So, if, for example, we have a configuration (0,1,2,-1), the first axis is located in the first quadrant (0^o to -90^o angle), the fourth axis is located in the second quadrant(90^o to 180^o angle) and the sixth axis is located in the first negative axis (0^o to -90^o angle).

Most targets can't be reached with every configuration, therefore the user must choose a valid configuration. The targets can be defined using the interface on robotstudio, using this method it is possible to test the different configuration and chose a valid one. Using rapid is not possible to verify the robot configurations automatically.

We can see in figure 24 a defined target, the first part corresponds to the position x, y and z, the second vector corresponds to the orientation expressed in quaternions, the third vector is the target configuration and the last vector defines the external joint axis which is used to define the axis positions of additional axes, positioners, or workpiece manipulators.

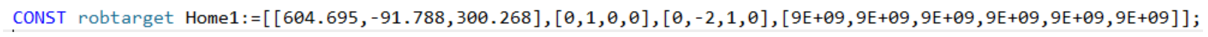

```
CONST robtarget Home1:=[[604.695,-91.788,300.268],[0,1,0,0],[0,-2,1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
```

Figure 24: RobTarget Example

4.3.3 Coordinate Systems Representation

World Coordinate System The world coordinate system (WCS) is the basic reference of the entire work station (robotic cell). This coordinate system is the highest in the hierarchy and all other coordinate systems are referenced with this one.

Workobjects A workobject is a coordinate system used to describe the position of a workpiece and represents a physical location of an object in a workstation [17].

The workobject is composed of two frames: a user frame and an object frame. All programmed positions will be related to the object frame, which is defined according to the user frame, which is specified about the world coordinate system, figure 25 [17].

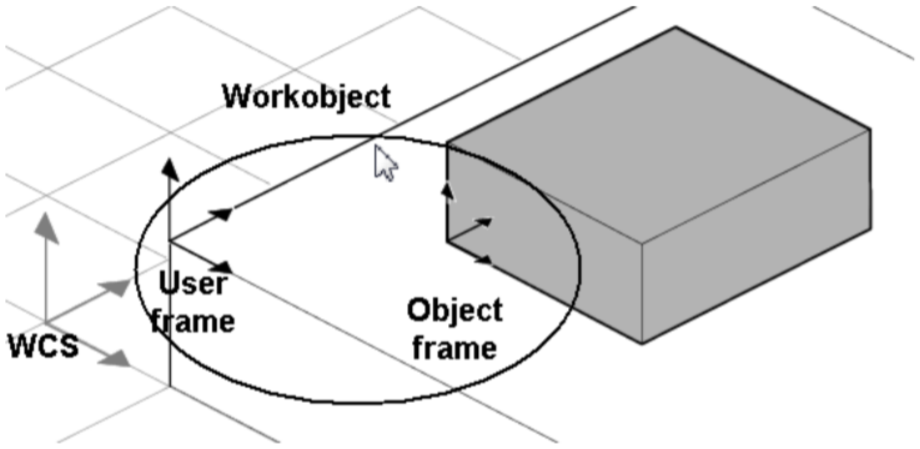


Figure 25: Workobject coordinate system [17]

This coordinate system is particularly useful due to its flexibility, reconfiguration is made easy by using two frames to define a coordinate system. When recalibration is required, it is only necessary to readjust the user frame and all the targets that are defined to that workobject will be adjusted accordingly. A workobject is also useful when the robot should move according to a specified geometry.

Tool Frame A tool frame is a coordinate system associated with a given tool, it has both an orientation and a location, like a target, and is used to facilitate the programming procedure of any motion where this tool is involved.

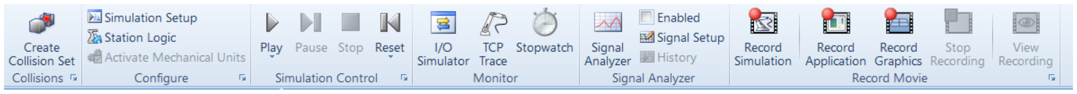
This coordinate system is mostly defined in the working end of the tool, for example in a welding torch, the tool frame is defined at the end of the torch. with the orientation of z pointing to the welding location.

4.4 ABB RobotStudio

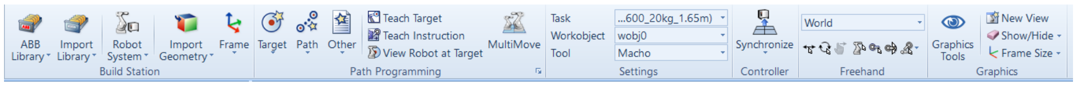
The ABB robotstudio is a program used to simulate real ABB robotic stations, it has a series of libraries where the user can pick a given robot, program it and simulate a real station.

This program has a graphical interface that simulates a real robot performing different tasks, it is a very appealing system because it allows the user to see different programmed routines working in the station the same it would work in the real robot.

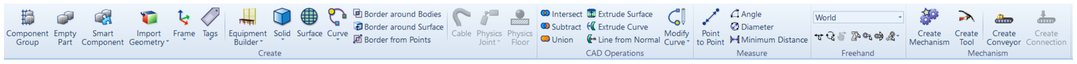
This software also allows for a more direct type of programming, as it has a series of buttons that facilitate the creation of targets, paths, workobjects, tools, etc. It is possible to see in figure 26, a series of buttons that allow the creation of different visual elements.



(a) Open/Close Screens



(b) Signal options



(c) Rapid Data

Figure 26: Different functions of the buttons

Smart components In many different station and simulations, there is the necessity of using smart components, these components are in fact properties associated with a particular tool or object, and are used to simulate a certain function

For example, to simulate the picking and placing of components there can be used attachers which connect two objects in the simulation, making them move has one, and dethatchers that cancel the attacher propriety.

Smart components can also simulate sensors, to detect objects inside the simulation, and can simulate constant movements to simulate for example a conveyor.

Parts Using robotstudio it is possible to import geometries that were built using CAD software. These geometries may have different purposes and they can, for example, simulate a product that the robot should pick, or an obstacle that the robot can not touch.

These parts are also the bases for simulating tools, it is possible to define a given geometry as a tool that the robot can use, it is even possible to define movement inside this tools such as closing and opening clamps of a gripping tool.

4.5 Screenmaker

The process of creating this flex pendant application using *screenmaker* is simple to understand. On the main page, there is an interface where the user drags different controls to the work area, which can be seen in figure 27, this is the area that will later be exported to the flex pendant. The user must then define what each of the objects inside the work area will do using a different interface, seen in figure 28.

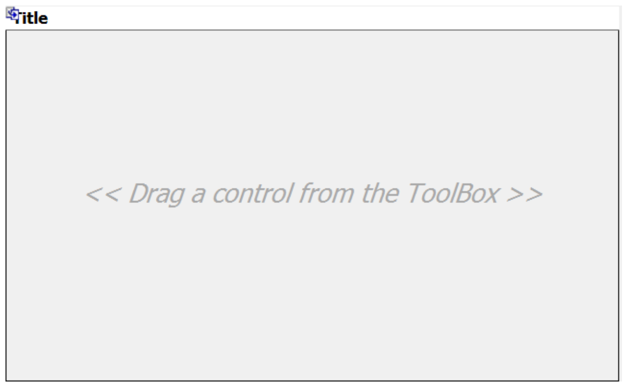
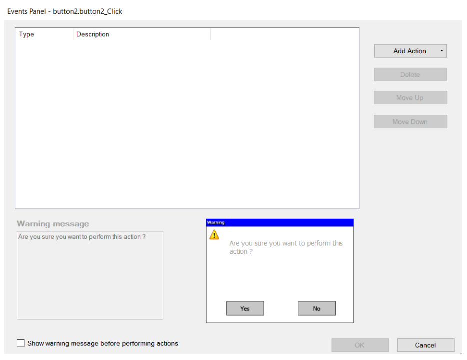


Figure 27: Empty Work Area


Events Panel - button2.button2_Click
Type
Description
Add Action
Delete
Move Up
Move Down
Warning message
Warning
Yes
No
Show warning message before performing actions
OK
Cancel

Are you sure you want to perform this action ?

Figure 28: Definition Screen

It is possible to see in figure 29 the most common controls used in the screenmaker applications. These controls can have different functions, and constitute the main tools that the programmer uses to define the HMI.

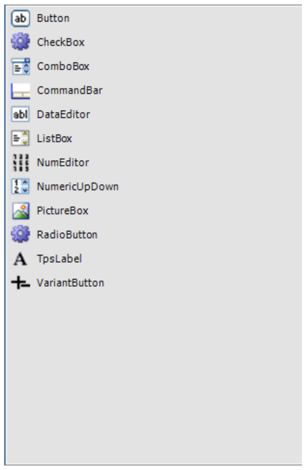
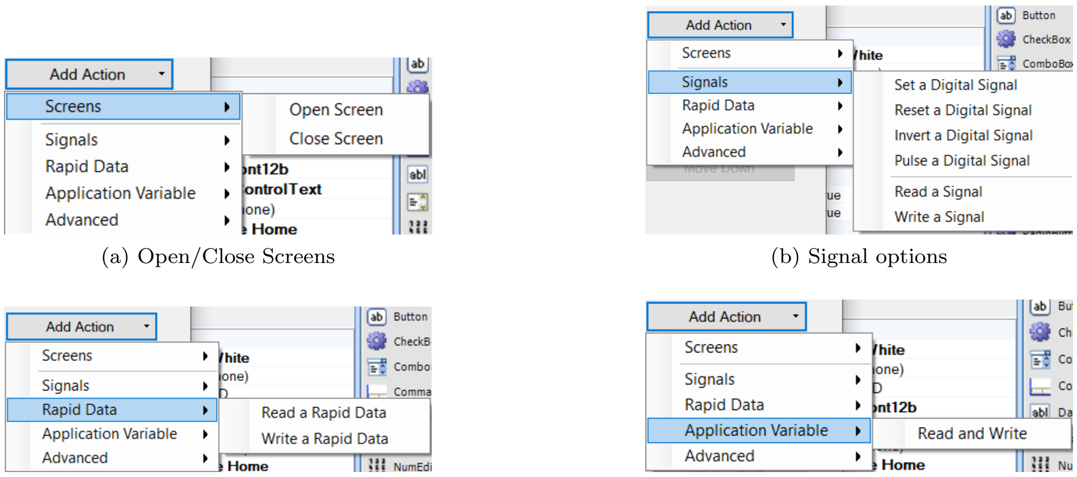


Figure 29: Most common controls

The most basic object used in the screenmaker is the button. To place a button in the HMI the programmer must drag this component to the work area, it can then define its properties and functions by clicking on the button.

There are different functions that this controller can perform, it can open and close a screen, it can read, write, set, pulse, invert or reset a signal, it can read or write a Rapid data, which, is a variable within the rapid part of the project, or it can read or write a variable within the screenmaker application. This functions can be seen in figure 30.



(a) Open/Close Screens (b) Signal options (c) Rapid Data (d) application variables

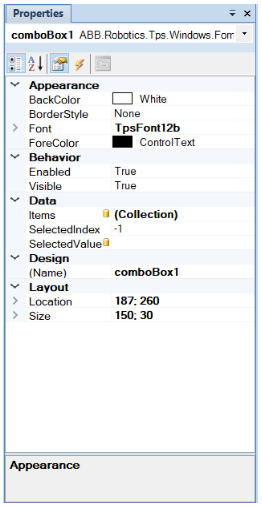
Figure 30: Different functions of the buttons

Another very common control used in this program is the combo box. These elements are used when the programmer needs to define a series of values from which the user can select one, for example, the user may be asked to select a tool from the array of available tools.

The values in this combo boxes are defined in the properties menu under data, they are bound to an array of options implemented further in the RAPID program. In the previous example, the combo box would be bounded to an array where all the available tools were declared.

The value inside the combo box can also be bonded to a single persistent variable inside the rapid program, this implies that when the user selects a value inside the combo box the persistent variable to which the combo box is bonded will also change accordingly.

The bonding of variables is made using the properties menu that can be seen in figure 31.



Appearance	
BackColor	White
BorderStyle	None
Font	**TpsFont12b**
ForeColor	ControlText
Behavior	
Enabled	True
Visible	True
Data	
Items	**(Collection)**
SelectedIndex	-1
SelectedValue	
Design	
(Name)	**comboBox1**
Layout	
Location	**187; 260**
Size	**150; 30**

Figure 31: Properties menu

An element that can be used to give feedback to the user is the led, this is a visual element that will light up under certain conditions, and can be bonded with both a signal or a variable. For example, a led can be added to inform the user if a certain tool is available or not.

Another useful elements are the labels, this are boxes with a given text, and can be used for giving basic instructions or identifying a certain tool. To define a label the user must drag this command from the control boxes and then write, the information he wants to display there.

There are also text and number boxes, which are elements where the user can freely define numbers and text respectively. These elements are useful for example to define coordinates because this element is not bonded to an array, this means the user can define any value for those coordinates.

4.6 Station

The first thing to do in this system is to create the station that will simulate the station, which is done in Robot studio. This simulation is both graphical due to a visual representation of the workstation and true to reality, as it could be directly applied to a station without the necessity of a great deal of further work.

As it was previously mentioned the robot is an ABB 2600 20kg 1.65m, which is contained in the libraries provided in robotstudio, therefore it is only necessary to select the said library and an appropriate controller to create a base station, which can be seen in figure 32

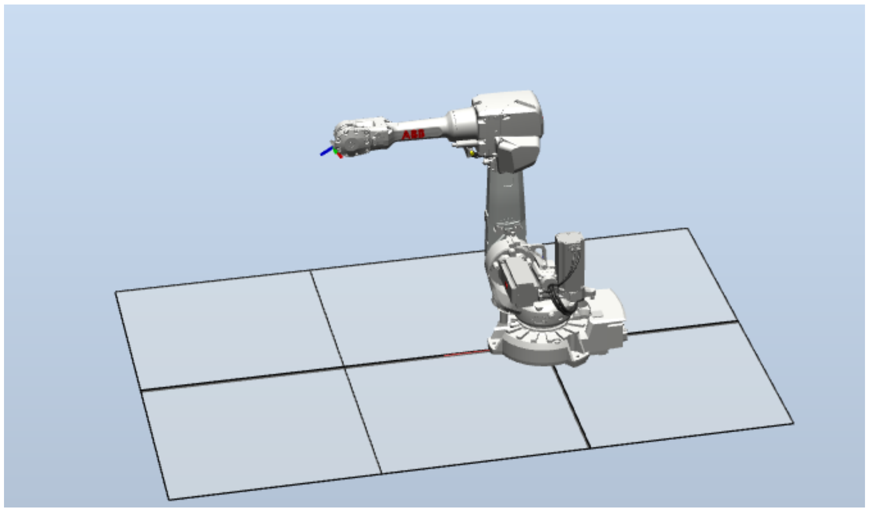


Figure 32: Empty Station

Once a station has been defined is important to import, create and attach the tool that is going to be used to the robot, in this case, the tool is the male tool changer.

To build this component, it is first necessary to get a 3D representation of the tool, this task is simplified because such a file is made available for download by *Zimmer*. Once the model has been downloaded it can be directly imported to the station, with an appropriate file type, which is a .SAT file. It is possible to see in figure 33 the male tool changer in the robot studio station.

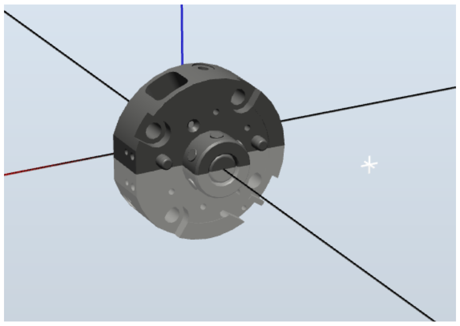


Figure 33: Imported male tool changer

With the tool imported, it is necessary to define a frame that will later be the tool center point. In the male tool changer, the tool center point corresponds to the center of the flange that goes inside the female tool changer to perform the connection between both parts. The orientation of this tool should be with z parallel to the flange.

Once both the geometry of the tool has been imported and the tool center point has been defined, it is possible to compile the tool.

In the robotstudio it was selected the create tool option, and used the imported geometry has the tool part and the previously created frame as the tool center point. It was also defined as the mass and center of gravity of this tool, for further calibration.

Once both the station and the tool were properly defined, it was necessary to import the rack, holders, and tools that were going to be used. This process is in all similar to the one used to make the geometry for the tool, a 3D model was generated and then imported to the robot studio using an appropriate file type.

The imported tool rack had merely a testing purpose and simulation requirements, it was not a defined part of the project at this point. This rack was designed to be as common and simple as possible, it was meant to simulate a possible tool rack. The geometry used can be seen in figure 34, again it could be used a different geometry.

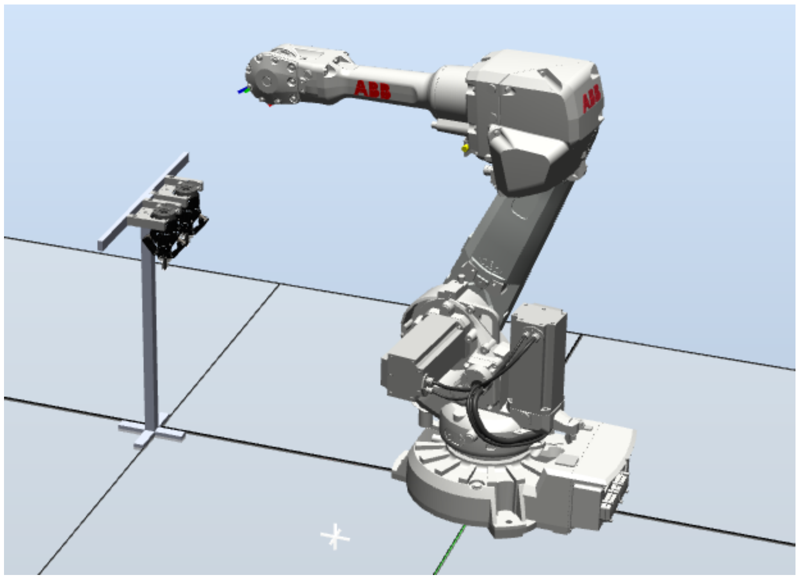


Figure 34: Station with tools

There was also the need to simulate sensors associated with the rack, in this case, retroreflective sensors, that would allow the robot to detect the presence of tools in a determined holder. These sensors can be simulated using smart components, it was used one for each of the holders.

Some tools may require an input to work, a gripper needs indication to close or open is fingertips, a welding tool needs information regarding when to weld, the male tool changer will also need to receive a signal to know when it can activate the pneumatic sensor to connect himself with the female tool changer. Smart components can again be used to simulate these effects.

To define the smart component it is necessary to first create an empty smart component and then define is function. To simulate the picking of tools it was used as an attacher, this component "glues" 2 parts together, as once active they move as one. The attacher is composed by two parts, the first one is the tool that will pick the object, in this case, the male part of the tool changer, and the second is the component that is going to be picked, which is the tool. To separate the components were dethatchers, which are defined similarly but have the opposite function.

This smart component will only work with the particular combination male tool changer and tool defined, this means it is not possible to have a universal smart component to pick or place any tool. To simulate multiple tools it is necessary to create multiple smart attachers and dethatchers, one for each of the tools.

4.7 Program Structure

The built station has the function to help develop and simulate the application is not associated with the final program. This final program can be divided into three hierarchy stages:

- The first is the calibration stage, at this moments all the paths, targets and workobjects needed to the tool changer program are generated, those are then saved to be used in subsequent stages
- The second stage is a manual testing moment, basically is where all the procedures required for the tool changer are defined and stored. It is also used to test if the tools can be safely picked and placed.
- Finally, the application stage, this moment is not truly imbibed in the created program, it corresponds to the user part of the program, is where he can call the created routines to pick and place the tools, which are stored in the manual testing moment.

The two first stages of the program are constituted by two other parts. A screenmaker application which is an HMI, and it is used to simplify the program use, and a RAPID program which controls the robot.

These two parts are not entirely separate, as they must communicate between them to share variables and orders.

In figure 35, it is possible to see an explanatory diagram of the communications between the entire program.

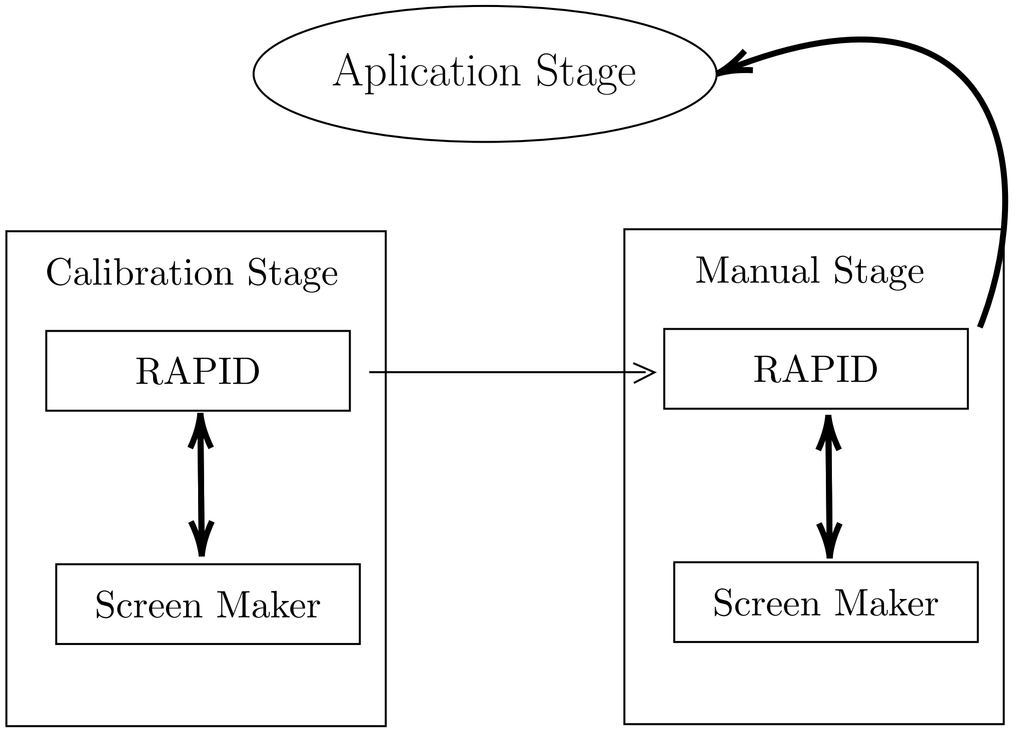


Figure 35: Communications between all the parts of the program

The communication between the RAPID programs in the calibration stage and the manual testing stage and the communication between the RAPID program in the manual stage and the application stage, is very straight forward, as they mostly share variables.

The communication between the screenmaker and the RAPID is a little more complicated, yet is somewhat systematic. In this section, there will be a more synthetic explanation of how some of these communications are made and a mapping of the different shared variables.

One of the most used mechanisms in this program is trap routines. They are used because some of the routines present in the program must be done when the user presses a button.

The different ways to interrupt the program have already been analyzed. It makes little sense to interrupt the routines from an error message or in a cyclical fashion.

It would be possible to either interrupt the routines from a signal or from a persistent variable. From the user point of view, it would significantly simpler to use persistent variables instead of signals, because, the former implies the creation of all the signals manually in each of the stations. Using the variable method, the information can be stored in the tool changer application.

The way this trap routines will be used is:

- The user presses a button, in the screen maker application.

- The application changes a variable to true inside the RAPID program
- These changes initiate the trap routine.
- Once this trap routine is finished the value is changed back to false

This behavior can be summarized using a grafcet, seen in figure 36

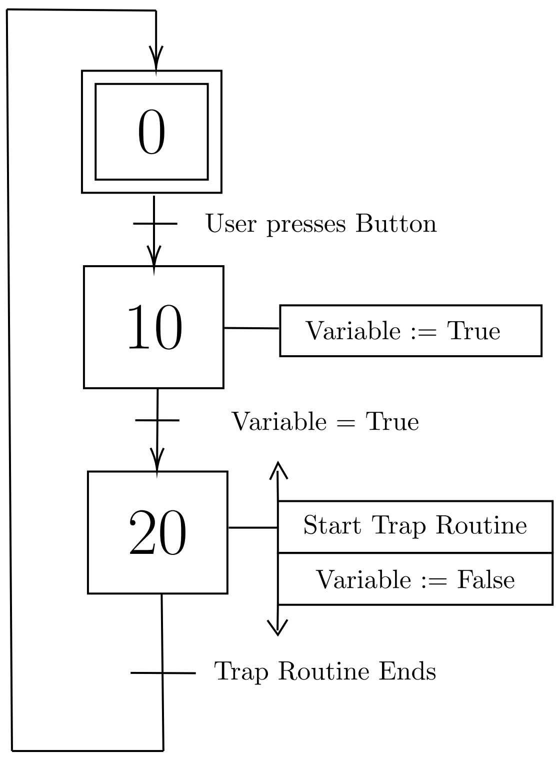


Figure 36: Trap Routine Grafcet

The shared variables, routines, and buttons can be seen in table 5. In the following section, the function of each routine will explain in detail.

Table 5: Traps Mapping

Trap	Variable	Button
TC_calibrate	TC_I_Calibrate	New Tool
TC_Pick	TC_I_Pick	Pick Tool
TC_Place	TC_I_Place	Place Tool
TC_Home_Trap	TC_I_HomeTrap	Define Home
TC_Changespeed	TC_I_Changespeed	Change Speed

Another very used element in this program are combo boxes, these elements are used to choose a value from an array of options. To do this, an array is created inside the RAPID program, this defines the values in the combo box inside the screenmaker, which are the ones the user can select. This behaviour can be explain by figure 37.

The value the user selects inside this combo box defines a persistent variable inside the RAPID program, that can then be used freely in other parts of the program.

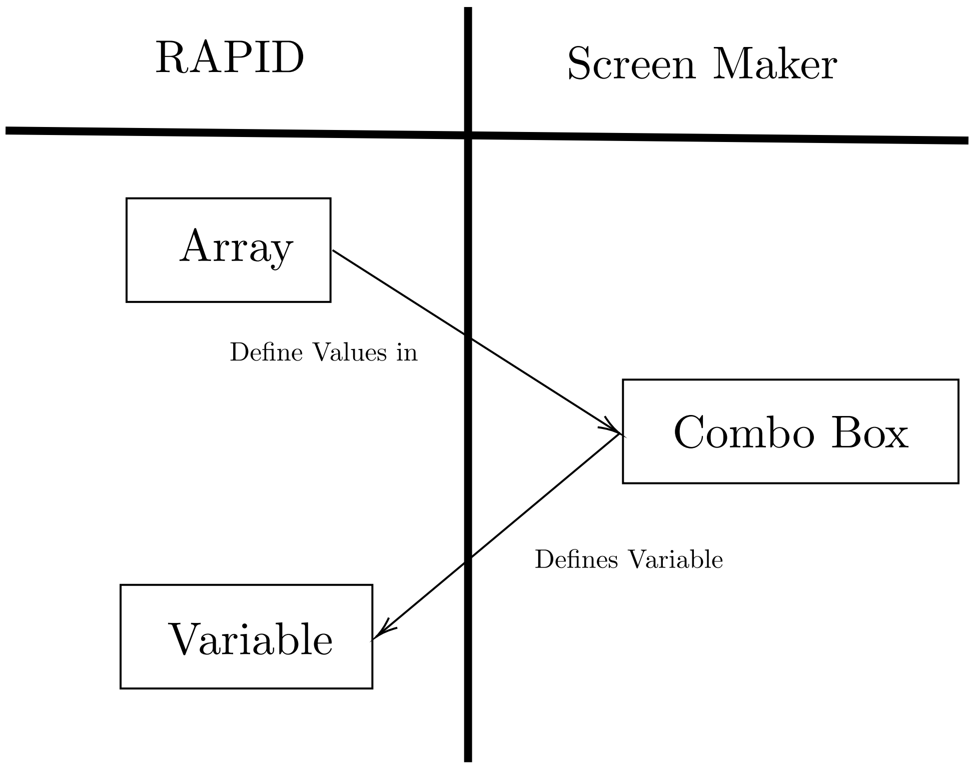


Figure 37: Combo box structure

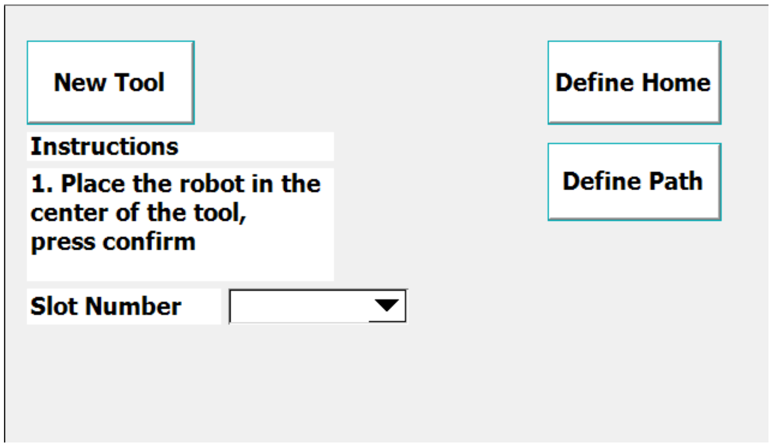
Table 6: Combo box variables

Array	Combo Box	Variable
TC_Slot	Tool Number	TCnumber
TC_AvailebelSpeed	Change Speed	TC_spd1

4.8 Calibration Stage

4.8.1 Screenmaker

It was proposed an initial solution for the HMI, to allow the user to quickly calibrate the tool changer system, this HMI can be seen in figure 38.



Instructions

1. Place the robot in the center of the tool, press confirm

Figure 38: HMI calibration system

In the presented solution the user must define a number using the combo box, which will be associated with the tool that it pretends to calibrate. Afterward, the user takes the robot to the tool location and presses confirm, this will interrupt the rapid main program and will initiate the process of building the targets, paths, and workobjects required to define the tool. Once this is complete the user can define the next target using the same method.

The button "define home" allows the user to define a new home position. When this button is pressed, a variable inside the rapid is changed and the trap routine starts.

The robot will define the home position as the position that he is currently in, so the user must first jog the robot to the desired location, guaranteeing that the fist of the robot is with the correct orientation.

Not all tools have the same dimensions, a certain tool may be wider and another be larger, this means that not all paths to pick or place tools should be equal, a path to pick a larger tool should have a larger movement range than the movement to pick a smaller tool. This application should also be able to be applied to different holders that require different paths.

The "define path" button, open a screen where the dimension of the movements can be changed, this new screen can be seen in figure 39

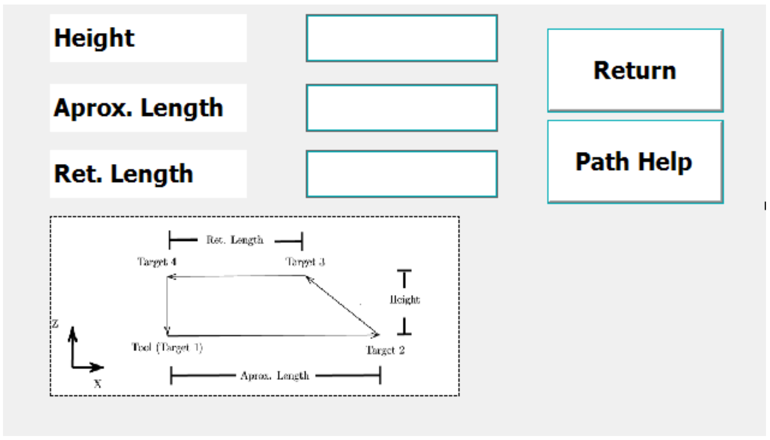


Figure 39: Define Path Screen

The values inside "height", "Aprox. Length" and "Ret.Length", are the values of the main dimensions for the paths, editing a number box in front of the label will change the corespondent dimension of the path for the next defined tool, it does not change the dimension of any of the previously defined tools.

This behavior is obtained by bonding this number boxes to variables inside the rapid program, this will later be used to define the different targets.

We can see in figure 40 the spectrum of movements executed for a path, this figure can be augmented by pressing the button "path help".

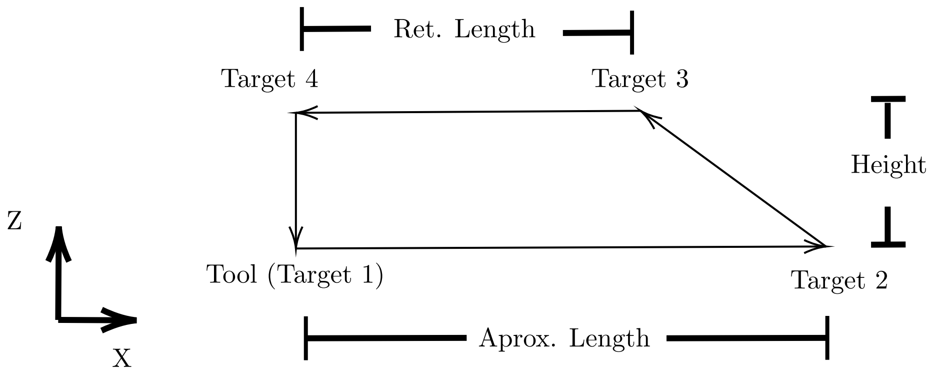
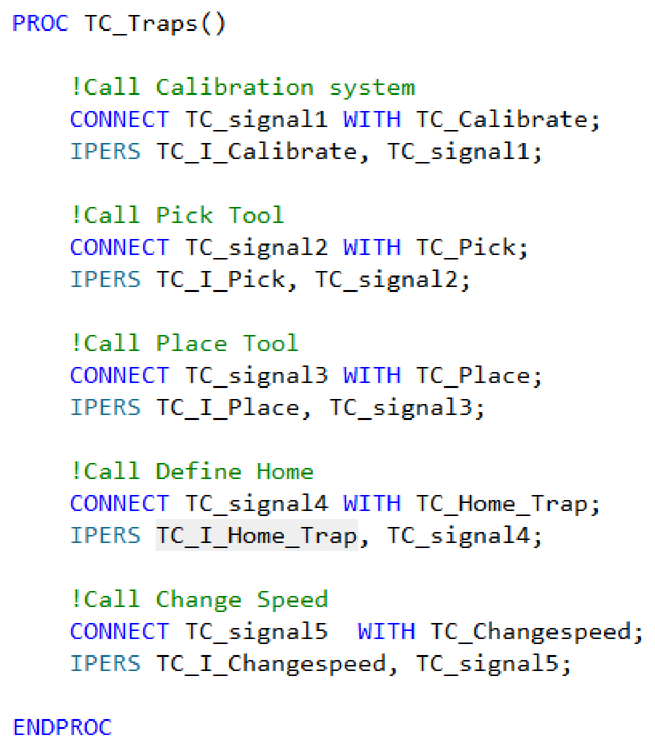


Figure 40: Path spectrum of movements

4.8.2 Rapid

It is now necessary to analyze how the RAPID program was built. In the first module (file) of the program all of the trap routines are connected, as in, the combination variable, button, trap is created.

The trap routines are located in different modules of the project, to connect them it is first necessary to create an intnum variable that is activated by the variable exposed in the "program structure" section and connect it with the trap routine that should be associated with that variable. It is possible to see in figure 41 the code that connects the different routines.



```
PROC TC_Traps()

    !Call Calibration system
    CONNECT TC_signal1 WITH TC_Calibrate;
    IPERS TC_I_Calibrate, TC_signal1;

    !Call Pick Tool
    CONNECT TC_signal2 WITH TC_Pick;
    IPERS TC_I_Pick, TC_signal2;

    !Call Place Tool
    CONNECT TC_signal3 WITH TC_Place;
    IPERS TC_I_Place, TC_signal3;

    !Call Define Home
    CONNECT TC_signal4 WITH TC_Home_Trap;
    IPERS TC_I_Home_Trap, TC_signal4;

    !Call Change Speed
    CONNECT TC_signal5  WITH TC_Changespeed;
    IPERS TC_I_Changespeed, TC_signal5;

ENDPROC
```

Figure 41: Trap Rotuines

So taking the first trap as an example, the "TC_signal1" is connected with the trap "TC_Calibrate", and when this variable changes to 1 the trap is activated. The "TC_I_Calibrate" is the variable responsible to changes the value of the "TC_signal1" to 1.

All of the traps necessary for the calibration stage are stored in a module called "calibration", and will now be analyzed separately.

The new tool button is associated with the TC_Calibrate routine, which is divided in 2 procedures:

- A procedure to generate the workobjects associated with that tool;

- A procedure to generate the targets necessary to pick and place that tool.

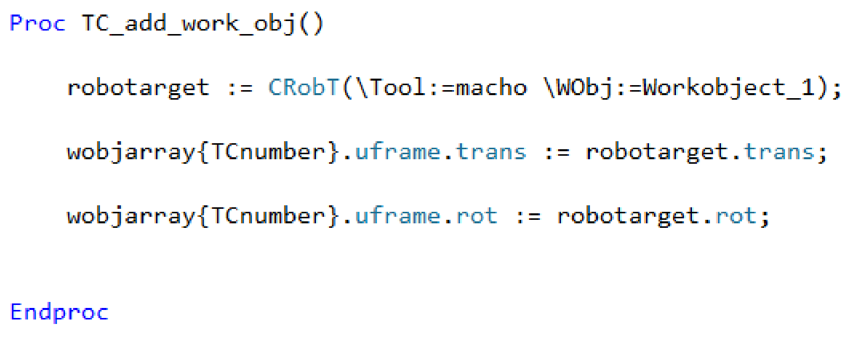
The best and most automatic way to define all the targets required for the path is, to first define a workobject for each of the tools, and then define all the targets for that tool about the created workobject. So, first, a procedure to generate the workobjects is called.

As it was said before a workobject is represented by two frames, being the most important the user frame, this is the frame that needs to be changed to later define all of the necessary targets. The user frame is defined in relation to a fixed referential that is universal, this frame could be, for example, the world coordinate system.

It was defined a workobject at the superior edge of the rack, all the workobject used in paths shall be created in the relation to this referential, this was labeled as "Workobject 1".

The definition of a universal workobject makes it easier to recalibrate the real system, as all the paths and targets will be characterized in relation to this one, so, if there is problem in the real system due to bad calibration, it is not necessary to re-calibrate all the targets and workobjects, it is only necessary to recalibrate that single workobject.

Once the routine to generate the workobjects as been called, the RAPID records the orientation and the position of the robot in that present moment and associates it with a workobject present in an array. The position in that array is defined by the "TC_number", this variable is bonded to the combo box that was previously created in the screenmaker application, the routine can be seen in figure 42.



```
Proc TC_add_work_obj()

    robotarget := CRobT(\Tool:=macho \WObj:=Workobject_1);

    wobjarray{TCnumber}.uframe.trans := robotarget.trans;

    wobjarray{TCnumber}.uframe.rot := robotarget.rot;


Endproc
```

Figure 42: Create workobject routine

In the RAPID programming language, every workobject must be declared and characterized previously to any usage, this kind of data doesn't allow the creation of an empty array with an undefined size. This means every workobject must be explicitly written down in the program. This fact poses a limitation to the number of tools that can be implemented in the station if there are 3 workobjects declared there can only be 3 tools, without changing the rapid program directly.

To solve this situation a large number of workobject should be declared to allow a more broad spectrum of tools. In this station, it was decided to consider 16 workobjects. The workobjects were saved inside an array, labeled "TC_wobjarray".

Once all of the workobjects have been generated is necessary to define the targets that are going to be necessary for the paths. The procedure responsible for creating the targets is called immediately after the "TC_add_wor_obj" is competed.

To generate all the targets it is necessary to consider the holder that is going to be used because it conditioned the movements that should be made. For an initial supposition the vertical holder shall be considered, this can, however, be changed by the user.

Analyzing the figure 14 it is possible to define 4 different target, the first target corresponds to the inside of the tool, where the connection between the male and female tool changer is made, the second target corresponds to a linear position outside the gap that guides the tool to the clippers, target number 4 is above the tool, this targets can be seen in figure 40

The routine to generate all the targets is called immediately after the workobjects have been generated. At this point the flange of the male tool changer is still located at the center of the female tool changer where this should be picked, analyzing figure 43 it is possible to conclude that this location can therefore be recorded as a target 1.

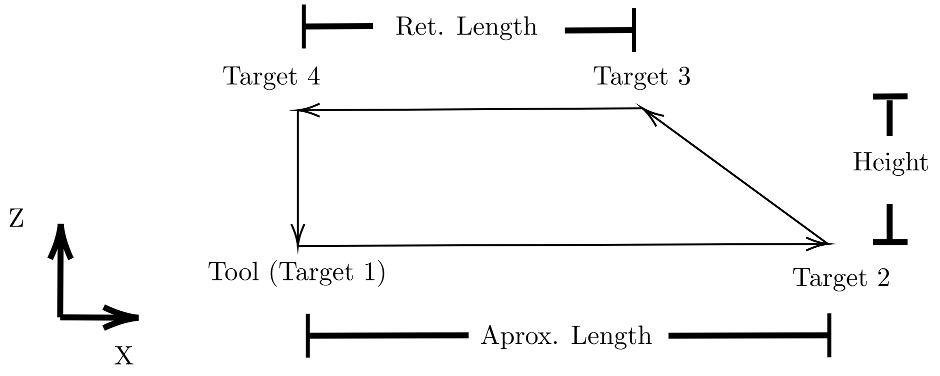
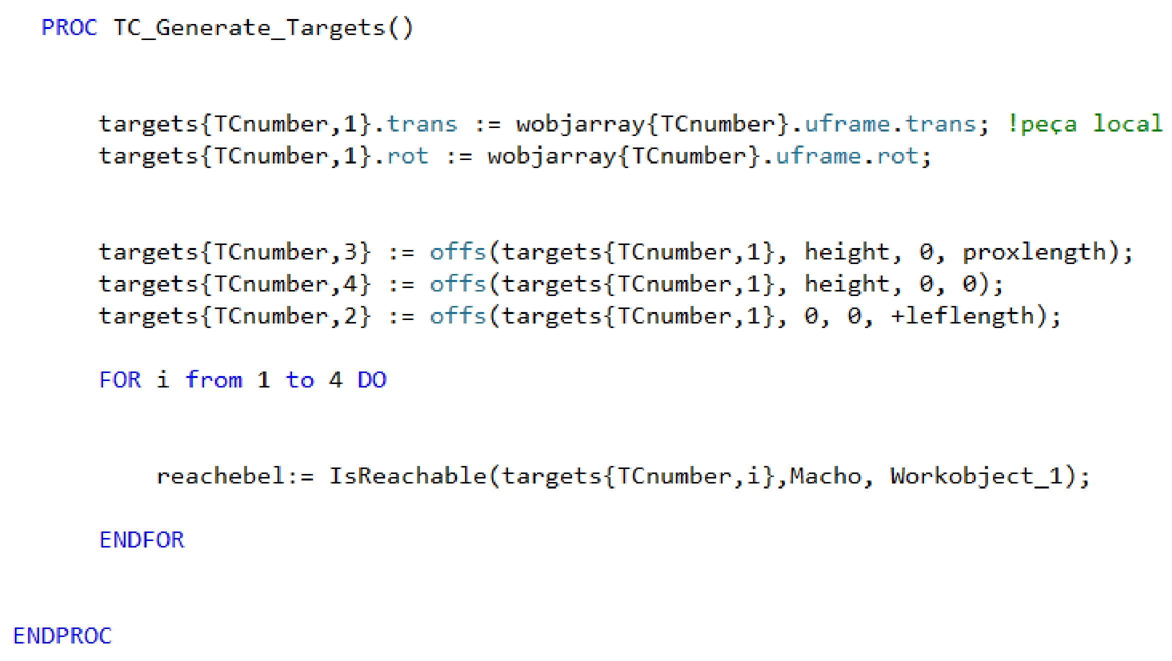


Figure 43: Path spectrum of movements

In all the targets implemented, the fist of the robot is facing downward which is a signicant advantage in the programming of the robot because all the targets are offsets from each other, this is valid for all the developed holder, however, it may not be valid for all the holders. Considering this, it is only necessary to determine the distance that needs to be implemented from each target, for example, if the user wants that target 2 to be 300mm from the target 1 all it is necessary to do is to create the new target with that extra distance.

To create this targets the offset function within rapid is used. This function creates a new target with a certain distance from a previously defined one. In table 6 we can see the base offsets implemented in the targets, and in figure 44 it is possible to see the implemented routine.



```
 PROC TC_Generate_Targets()


     targets{TCnumber,1}.trans := wobjarray{TCnumber}.uframe.trans; !peça local
     targets{TCnumber,1}.rot := wobjarray{TCnumber}.uframe.rot;


     targets{TCnumber,3} := offs(targets{TCnumber,1}, height, 0, proxlength);
     targets{TCnumber,4} := offs(targets{TCnumber,1}, height, 0, 0);
     targets{TCnumber,2} := offs(targets{TCnumber,1}, 0, 0, +leflength);

     FOR i from 1 to 4 DO


         reachebel:= IsReachable(targets{TCnumber,i},Macho, Workobject_1);

     ENDFOR


ENDPROC
```

Figure 44: Generate targets

Table 7: Recommended Settings

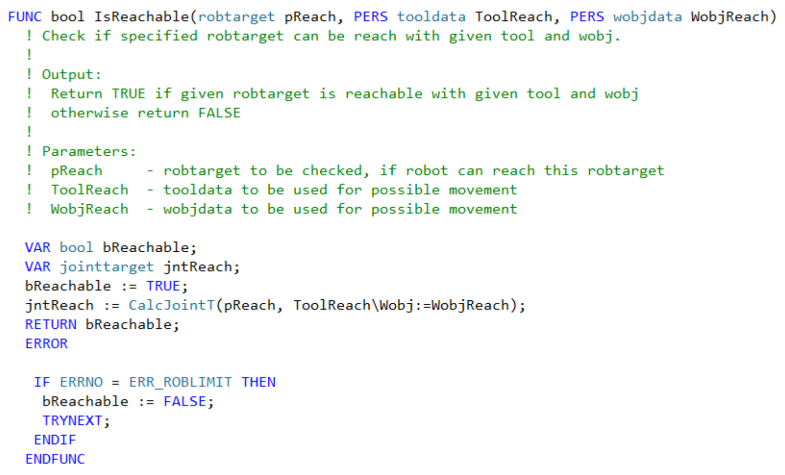
0.5

target	x	y	z
1	0	0	0
2	-61	0	+107
3	-61	0	0
4	0	0	307

These are just the recommended settings and may not work for every tool, however, the user can define is own distances inside the number boxes seen in figure 34. These number boxes are bonded to variables ”height”, ”proxlength” and ”leftlength”, and are responsible to define the offset.

All of the generated targets are recorded inside an empty matrix, in this case the paths have 4 targets so we need a matrix with 16 vectors of 4 targets, each vector contains the target for its workobject, which means it has the targets for one of the tools, so vector one has the targets for tool one, the second vector has the targets for the second tool, and so on.

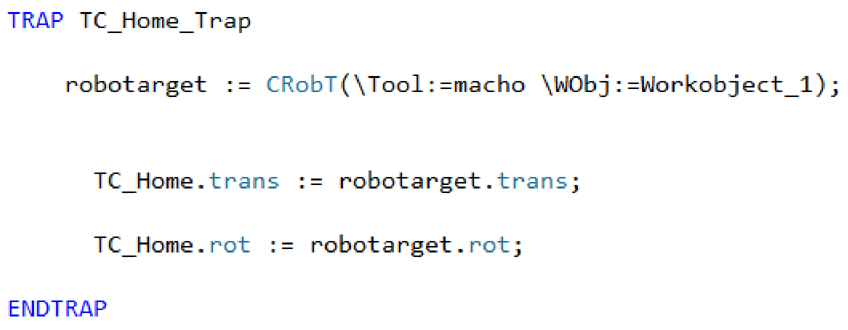
It was also added a function within the routine that generates the targets to verify if all the targets are within the robot range, doing this verification in the calibration routine is important because it prevents errors during the execution phase. This function can be seen in figure 45.



```
FUNC bool IsReachable(robtarget pReach, PERS tooldata ToolReach, PERS wobjdata WobjReach)
  ! Check if specified robtarget can be reach with given tool and wobj.
  !
  ! Output:
  !  Return TRUE if given robtarget is reachable with given tool and wobj
  !  otherwise return FALSE
  !
  ! Parameters:
  !  pReach     - robtarget to be checked, if robot can reach this robtarget
  !  ToolReach  - tooldata to be used for possible movement
  !  WobjReach  - wobjdata to be used for possible movement

  VAR bool bReachable;
  VAR jointtarget jntReach;
  bReachable := TRUE;
  jntReach := CalcJointT(pReach, ToolReach\Wobj:=WobjReach);
  RETURN bReachable;
  ERROR

   IF ERRNO = ERR_ROBLIMIT THEN
    bReachable := FALSE;
    TRYNEXT;
   ENDIF
  ENDFUNC
```

Figure 45: Is reachable function

The define home button defines the home position of the program, there is a location where the robot should go after or before picking the tool. This location is defined using a target, and to calibrate it the robot must be placed in the home position that is desired that location is stored in a TC_Home target, this routine can be seen in figure 46.



```
TRAP TC_Home_Trap

    robotarget := CRobT(\Tool:=macho \WObj:=Workobject_1);


      TC_Home.trans := robotarget.trans;

      TC_Home.rot := robotarget.rot;

ENDTRAP
```

Figure 46: Home Trap

Once all the targets have been generated the user can now pass to the next part of this project, which consists of the usage stage of the tool changer system, this is the picking and placing of tools.

4.9 Manual and Testing Stage

4.9.1 Screenmaker

The second stage of this application it's used to test the routines to pick and place the tools.

It was proposed an initial solution for the HMI that can be seen in figure 47. In this solution it is possible to see a combo box labeled ”Tool number” where all the defined tools are identified and can be selected, this the combo box is directly bonded to a variable inside the rapid program that will define which tool will be picked.

The user must test the picking routines and the placing routines, to do this there are 2 buttons in the HMI, the ”pick tool”, which is used to test the pick routine, and the ”place tools” which isused to test the place the tool routine. These buttons trigger the traps identified at the ”program structure” section.

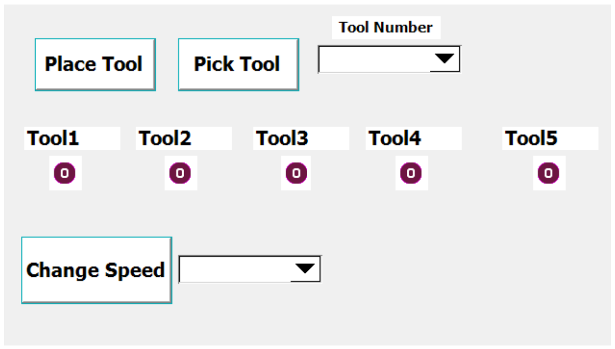


Figure 47: HMI Manual testing stage

Analyzing figure 47 it is possible to see 5 LEDs, the value of these LEDs is related to the smart components generated has retroreflective sensors, and are destined to determine if there is a tool in the holder identified by the number on the label.

The button ”change speed” allows the user to define the speed with which the robot will perform its tasks, this is bonded to an array with the different available speeds, and can be seen in figure 48. There is also an option named ”test speed” this is the slowest possible speed to implement in the robot, it is used to test if in any part of the work the tool or robot hits any of the static elements. The user selects this speed and then runs the program to pick or place for a given tool, the movement is slow enough for the user to evaluate if the tool collides with the holder at any moment.

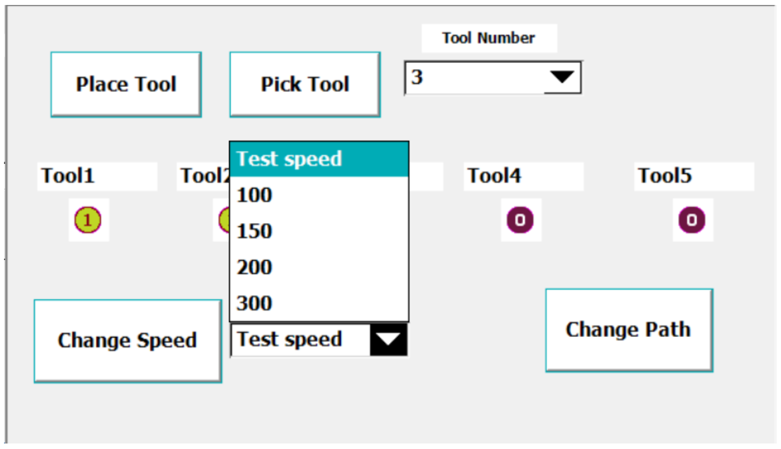
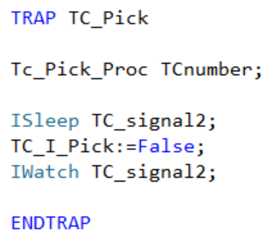


Figure 48: Available Speeds

4.9.2 Rapid

The routine named ”TC_Pick” is responsible for the picking of the tool and the routine named ”TC_Place” is responsible for the placing of the tools, and are triggered by the button identified in the table 5. The same way every workobject must be declared to function, so must every ”move” instruction and therefore every path, this creates a big problem when trying to escalate the solution, declaring, for example, 200 path and writing it down individually is inefficient and troublesome.

As it was said before the trap routines use internal variables that fire the routine every time its value is changed, however, the screenmaker buttons can only change the value to either true or false, this means, that the value must be changer back to his original value once the routine is finished.

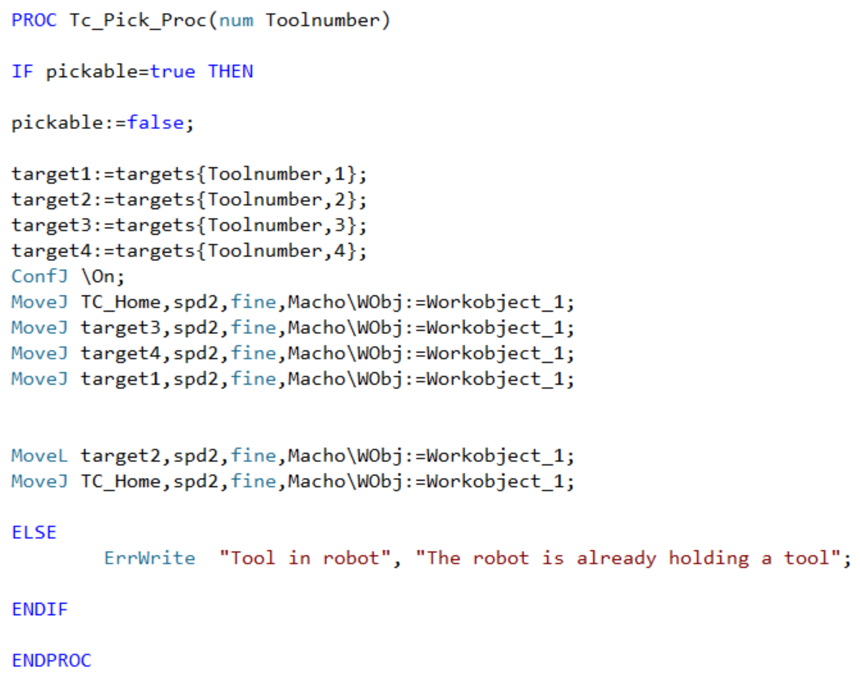


```
TRAP TC_Pick

Tc_Pick_Proc TCnumber;

ISleep TC_signal2;
TC_I_Pick:=False;
IWatch TC_signal2;

ENDTRAP
```

Figure 49: Pick Routine one

Notice the functions ”ISleep” and ”IWatch” this function prevents the trap routine to be triggered when the routines change back to his original value, these functions can be seen in figure 49.

So at the beginning of the procedure, the targets inside the move instruction will be replaced by the targets of the vector defined by the combo box. This allows the existence of only one path, and the targets inside this one path varying according to the combo box present in screenmaker application, allowing a more broad program with less computational necessities. This can be seen in figure 50.



```
PROC Tc_Pick_Proc(num Toolnumber)

IF pickable=true THEN

pickable:=false;

target1:=targets{Toolnumber,1};
target2:=targets{Toolnumber,2};
target3:=targets{Toolnumber,3};
target4:=targets{Toolnumber,4};
ConfJ \On;
MoveJ TC_Home,spd2,fine,Macho\WObj:=Workobject_1;
MoveJ target3,spd2,fine,Macho\WObj:=Workobject_1;
MoveJ target4,spd2,fine,Macho\WObj:=Workobject_1;
MoveJ target1,spd2,fine,Macho\WObj:=Workobject_1;


MoveL target2,spd2,fine,Macho\WObj:=Workobject_1;
MoveJ TC_Home,spd2,fine,Macho\WObj:=Workobject_1;

ELSE
        ErrWrite  "Tool in robot", "The robot is already holding a tool";

ENDIF

ENDPROC
```

Figure 50: Procedure to pick tool

Most of the ”move” instructions are made according to a joint type of motion, this allows more flexibility of position has the robot can move more freely not needing to assure a linear path.

The motion towards the target 2 which consists of the taking of the tool from the holder, is linear due to the existence of a linear holder guide used in the vertical holder.

Examining figure 50 it is possible to see that there is a variable named pickable if this variable is false, the routine to pick the tool won’t start. This ensures that the robot can’t pick a tool when it already has one picked, has this would create a robot malfunction.

If the user still presses the button to pick a tool when the robot is already holding a tool, the flex pendant will display an error message, informing the user that already exists a tool in the robot, this error can be seen in figure 51.

This variable becomes false once the robot performs the pick instruction, and becomes true, allowing the picking of a new tool once the robot performs the place instruction.

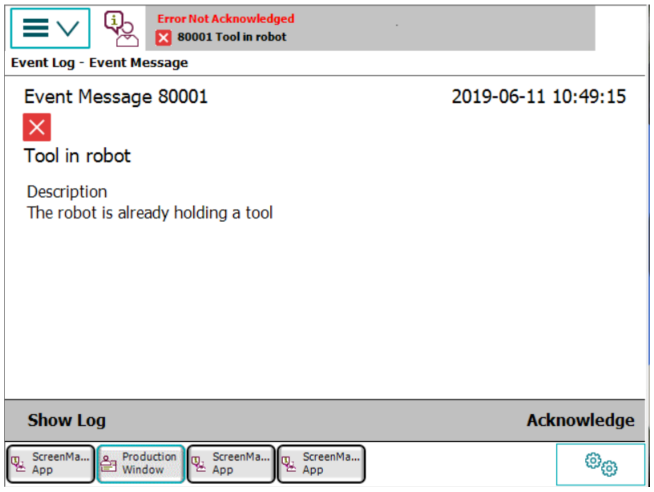

Error Not Acknowledged
80001 Tool in robot
Event Log - Event Message
Event Message 80001 2019-06-11 10:49:15
Tool in robot
Description
The robot is already holding a tool

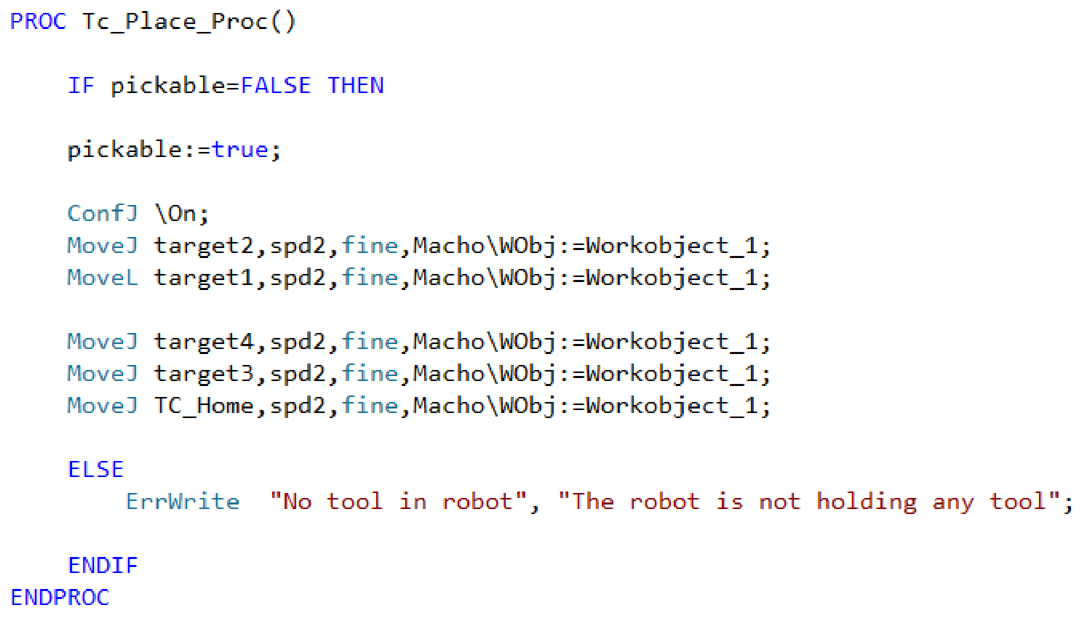
Figure 51: Error of tool in robot

There is also a noteworthy function named ”confJ”, this is a function inherent to RAPID, it allows a robot’s joint configuration to be automatically defined(if the value is on) or not(if the value is off).

For this application the value is on because the robot should always get to define target being the configuration with which it reaches there less important, this allows smoother work as the user will never receive error message associated with an invalid configuration once the calibration has been made, it has, however, a disadvantage regarding the movements of the robot, as sometimes this can be awkward and not the most effective one.

The trap routine responsible for placing the tools can be seen in figure 52, it is somewhat similar to the pick instruction, because they share the same targets and types of movements, however for this routine the program won’t replace the previously generated targets, this is due to the fact that the robot should always place the tool in the same location, ergo the targets that robot will go to place the tool will remain the same regardless of any further interaction of the user with the combo box.

If instead the robot was required to place a tool in a user-defined location, the targets would need to be replaced, for example, for the targets inside a vector identified by a new combo box.



```
PROC Tc_Place_Proc()

    IF pickable=FALSE THEN

    pickable:=true;

    ConfJ \On;
    MoveJ target2,spd2,fine,Macho\WObj:=Workobject_1;
    MoveL target1,spd2,fine,Macho\WObj:=Workobject_1;

    MoveJ target4,spd2,fine,Macho\WObj:=Workobject_1;
    MoveJ target3,spd2,fine,Macho\WObj:=Workobject_1;
    MoveJ TC_Home,spd2,fine,Macho\WObj:=Workobject_1;

    ELSE
        ErrWrite  "No tool in robot", "The robot is not holding any tool";

    ENDIF
ENDPROC
```

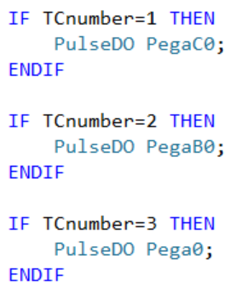
Figure 52: Place Routine

In both routines it can be implemented a function named ”PULSEDO” when this function is called a digital output is pulsed, this represents a signal that is sent from the robot which can have different functions. This output represents the signal that should be sent to the tool when it is in position to place or pick the tool, in this simulation, this can be represented using smart components, the attacther (to pick) and the detacher(to place).

This kind of components can only be associated with a child part, this means that it is not possible to make the robot pick any tool at any moment, we can not teach the robot to pick an arbitrary tool, it can only pick the defined tools in the smart components.

For simulation purposes, and considering 3 tools, it would be needed a definition of 6 smart components, an attacher and a dethatcher for each of the tools.

Once all of the smart components have been created, the tools would need to be defined and stored in determined slots, for example, slots 1 2 and 3. The rapid program would then need to be manually changed so that the robot knew that when doing the movements referent to these slots it should pulse the signals that would activate the respective smart components. The code that will perform these tasks can be seen in figure 53



```
IF TCnumber=1 THEN
    PulseDO PegaC0;
ENDIF

IF TCnumber=2 THEN
    PulseDO PegaB0;
ENDIF

IF TCnumber=3 THEN
    PulseDO Pega0;
ENDIF
```

Figure 53: Simulation of the attatchers and dettatchers

As it was said this is merely a hypothetical scenario if a different set of tools would be implemented all the smart components would need to be replaced.

The real robot, however, would not require such a large definition, since this sends a real signal to the tool, and in this case, the tool is a male tool changer that can pick any of the tools that are going to be used for the multi-product line.

The program would still require to send a signal to the tool, however, this signal would always be the same, and would give of the order of advance for the pneumatic cylinder inside the male tool changer.

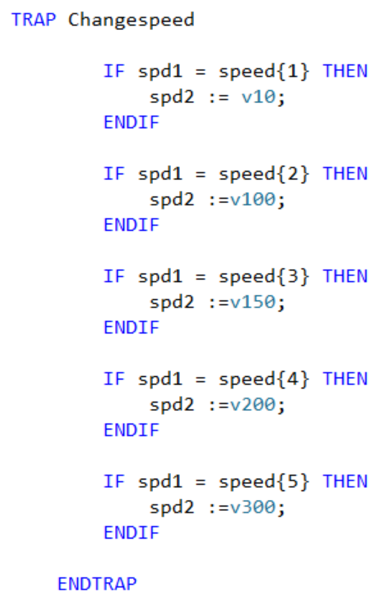
The change speed is the button used to change the speed of the robot, this element changes the variable "spd2" inside the move instructions which is responsible for the definition of the speed in that instruction.

The speed that will be defined is stored in an array labeled "TC_speeds" which should be a speed data persistent array, however, speed data can not be bonded directly with a combo box inside the screen maker.

To solve this problem, the array was made of strings, with each of the strings having a determined name associated with the correspondent speed, so the speed v100 would be associated with the string "100". A second string variable was created and connected with the value inside the screen maker combo box, so this value would change according to the value inside the combo box, which is defined by the user.

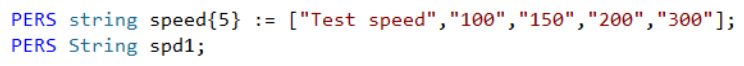
Once both the array and the variable have been created it is possible to make a simple program that associates the speed data with the value that should be implemented and was defined in the combo box.

This program should be called when the user presses the change speed button, therefor there is trap named "Tc_changespeed". This routines can be seen in figure 54 and figure 55



```
TRAP Changespeed

        IF spd1 = speed{1} THEN
            spd2 := v10;
        ENDIF

        IF spd1 = speed{2} THEN
            spd2 :=v100;
        ENDIF

        IF spd1 = speed{3} THEN
            spd2 :=v150;
        ENDIF

        IF spd1 = speed{4} THEN
            spd2 :=v200;
        ENDIF

        IF spd1 = speed{5} THEN
            spd2 :=v300;
        ENDIF

    ENDTRAP
```

Figure 54: Change Speed



```
PERS string speed{5} := ["Test speed","100","150","200","300"];
PERS String spd1;
```

Figure 55: Values inside the array

4.10 Tool Changer Implementation

This section is a basic explanation of how the user can download, implement, and use this application in his station. Notice that there are a few limitations on where this application can be used, even Although it can be adapted by the user to better fulfill his needs:

- This application can only be used in ABB Robots which are the only type of robots compatible with the programming language used.
- It can only be used with the defined tool changer without changing the calibration data of the tool. This can, however, be changed in the calibdata module.

The first thing the user must do is to download the application, which is available in a website built for this single purpose (https://abbtoolchanger.wordpress.com/). The website is at a very early stage, however, it fulfills its objectives. The main page can be seen in figure 56. Once the application has been download the user must extract the files from the zipped file to his computer.

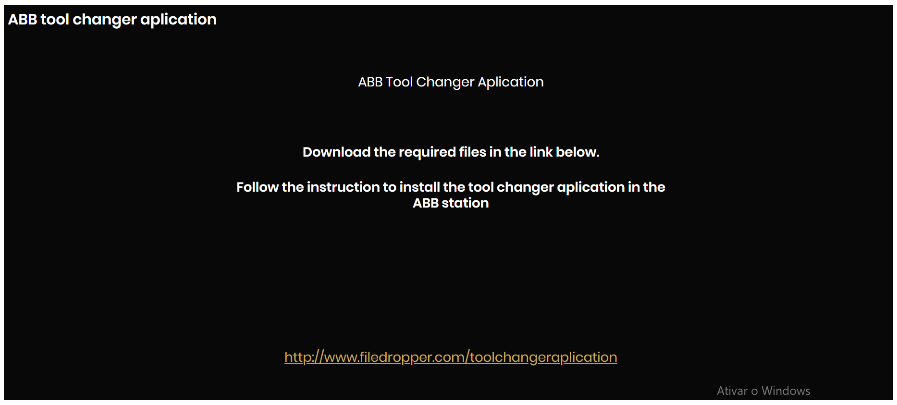


ABB tool changer aplication

ABB Tool Changer Aplication

Download the required files in the link below.

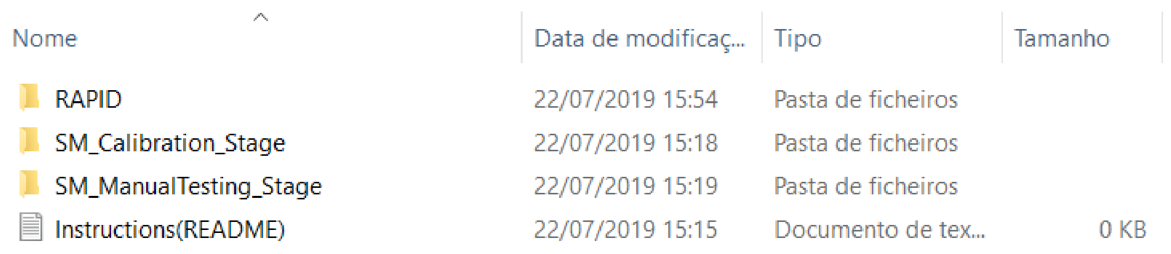
Follow the instruction to install the tool changer aplication in the ABB station

http://www.filedropper.com/toolchangeraplication

Ativar o Windows

Figure 56: Main page of website

In the folder that has the program (which can be seen in figure 57), there are 3 other folders, one of them has all the RAPID code necessary to run this application. To implement this code, in the robotstudio application the user must go to the ”rapid” tab and load all of the modules to the correspondent controller.



Nome	Data de modificaç...	Tipo	Tamanho
RAPID	22/07/2019 15:54	Pasta de ficheiros	
SM_Calibration_Stage	22/07/2019 15:18	Pasta de ficheiros	
SM_ManualTesting_Stage	22/07/2019 15:19	Pasta de ficheiros	
Instructions(README)	22/07/2019 15:15	Documento de tex...	0 KB

Figure 57: Folder containing Tool Changer Files

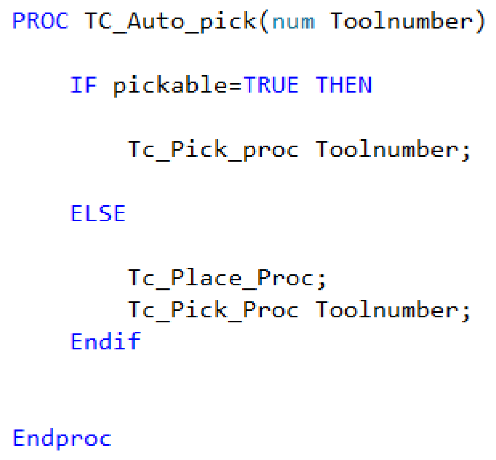
There are 2 other folders in this application, these contain both the screenmaker applications and can be directly loaded onto the robotstudio station. To do this the user goes to the screenmaker tab, chooses loaded, selects the folder correspondent to the screenmaker application, and then open the .smk file.

To call the routines that pick and place the tools in the user's program, the user must first activate the trap routines by calling the "ActivateTraps" procedure. This will activate all the trap routines. Once this has been done, the user can use both the screenmaker applications to calibrate every tool without further work.

Once this procedure is done the application is in the station, however, the user must first calibrate the "TC_workobject1" to the corresponding users tool rack.

The user is now also capable of using the pick and place instruction by simply writing in his code "Tc_pick_proc(num)" where "num" represents the number of the tool he wants to pick. To place the tool he must write "TC_place".

There is another function developed for a faster application of the routines, it is called "TC_Auto_Pick", it is used to pick a tool regardless of the current tool being held. If the robot is holding a tool that his not the correct one, calling this routine will automatically make the robot place the current tool and pick the right one. This function can be seen in figure 58.



```
PROC TC_Auto_pick(num Toolnumber)

    IF pickable=TRUE THEN

        Tc_Pick_proc Toolnumber;

    ELSE

        Tc_Place_Proc;
        Tc_Pick_Proc Toolnumber;
    Endif


Endproc
```

Figure 58: Autopick function

5 Dual Gripper Adapter

5.1 Dual Gripper Adapter Objectives

One of the main goals of this dissertation was to build a flexible tool changer system. To add flexibility to a system it is important to analyze the process of developing a new tool that can be used with this tool changer.

It was therefore analyzed a real problem of a real production line, in this case, it was necessary a tool that could pick both an individual part, as well as a blister.

A blister is a system that holds a variety of parts within them, it is used to easily pack and transport an array of parts These are usually made out of plastic, and the one that was used was fairly light-weighted.

Inspecting the dimensions of the blisters it is possible to consider some of the complications of balancing it, especially considering the use of only one robot with a single arm to maneuver it, furthermore each part must be able to be picked individually. To solve this problem it was developed a special tool with 2 grippers that had a 90^{o} angle between them to pick the blister and different parts on it.

The angle between the grippers would allow some distance between the fingertips of the grippers, providing balance for the blister once it was being held in the air, this angle would also allow good handling for the parts that were going to be picked individually.

The grippers that were going to be used had already been acquired, they were a Robotiq 2F-85, therefore it was only necessary to build a holder to hold both the gripers in place with the defined angle and a distance.

5.2 Dual Gripper Adapter Connections

5.2.1 Robotiq 2F-85

To build the adapter it was first necessary to understand and define how the different components would be coupled with each other, this is, how would the connection between the grippers and the adapter, and the adapter with the robot, be made.

Examining the catalog provided by *robotiq*, it is possible to see that are two main aspects to consider when dimensioning the connection with the adapter. The first aspect is the gripper, which consists in the mechanic tool that will open and close to grab the different objects, the second is the coupling, this is where the gripper will be mounted. The coupling will then be connected with the robot, or, in this case, to the female tool changer.

The coupling also has a pig tail cable that does the connection with the device cable to assure communication with the robot. This connection is serial RS-485, and consists of a connection to pass information such as orders to open or close the gripper [18].

The different components of the gripper can be seen in figure 59

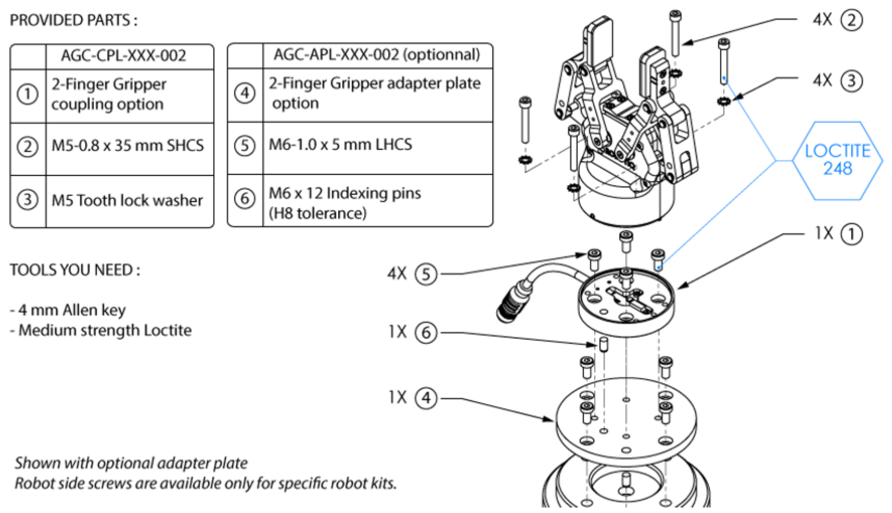

PROVIDED PARTS :
AGC-CPL-XXX-002
① 2-Finger Gripper coupling option
② M5-0.8 x 35 mm SHCS
③ M5 Tooth lock washer
AGC-APL-XXX-002 (optionnal)
④ 2-Finger Gripper adapter plate option
⑤ M6-1.0 x 5 mm LHCS
⑥ M6 x 12 Indexing pins (H8 tolerance)
TOOLS YOU NEED :
- 4 mm Allen key
- Medium strength Loctite
Shown with optional adapter plate
Robot side screws are available only for specific robot kits.
4X ②
4X ③
LOCTITE 248
1X ①
4X ⑤
1X ⑥
1X ④

Figure 59: Assembly of the gripper [18]

We can see in figure 60 the representation of the coupling.

The blue area represents the area that can be edited to allow a connection between the gripper and the adapter, in this blue area there can be made holes with any depth, where screws or bolts can be inserted to allow the connection with the adapter. In the grey area, holes can be made up to 3mm in depth, not being a good option for connection [18].

The manufacturer suggests and produces couplings that already have a type of connection made, which are made according to different norms. The gripper that was acquired had a coupling made according to ISO 9409-1-50-4-M6. This type of connection secures the gripper to the coupling using 4 m6 screws that are spaced with a 90 degree angle between them and are located 25mm away from the center of the coupling.

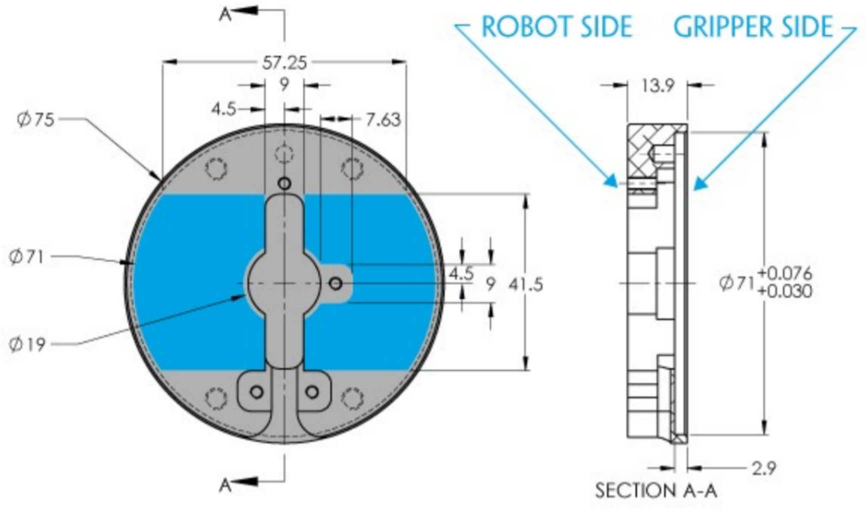


Figure 60: Editable part of the gripper [18]

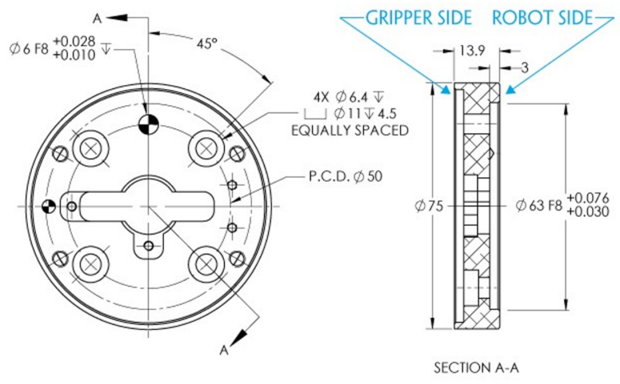


Figure 61: Coupling according with ISO 9409-1-50-4-M6 [18]

The other factor associated with the gripper that needs to be considered is the main dimension of this tool.

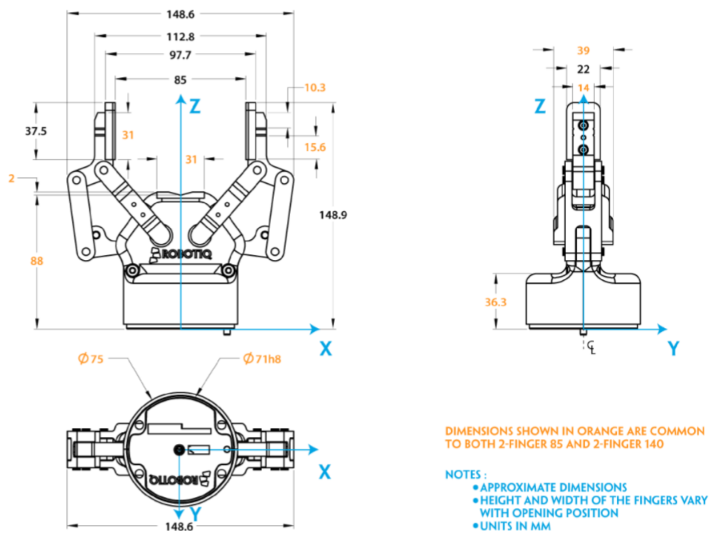

148.6
112.8
97.7
85
10.3
37.5
31
31
15.6
2
88
148.9
Z
X
Ø75
Ø71h8
X
Y
148.6
39
22
14
Z
36.3
Y
DIMENSIONS SHOWN IN ORANGE ARE COMMON TO BOTH 2-FINGER 85 AND 2-FINGER 140
NOTES :
• APPROXIMATE DIMENSIONS
• HEIGHT AND WIDTH OF THE FINGERS VARY WITH OPENING POSITION
• UNITS IN MM

Figure 62: Main dimensions of the gripper with finger opened [18]

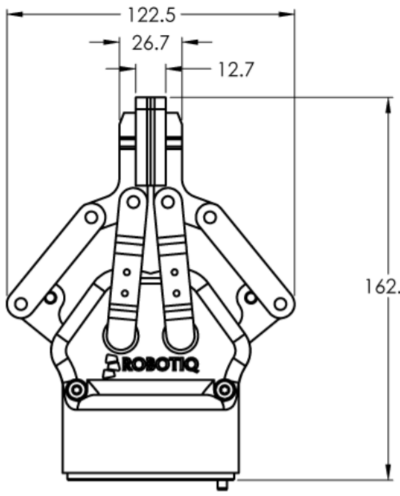


Figure 63: Main dimensions of the gripper with finger closed [18]

Analyzing the schematics presented in figures 61, 62, 63, it is possible to see that the height of the coupling is 13.9mm with a diameter of 75mm, the height of the griper is of 148.9mm with is fingertips closed, and 162.8 with is fingertips fully open. The gripper can open up to 85mm.

5.2.2 Female Tool Changer

After understanding how the connection between the gripper and the adapter is made, it is now necessary to define the connection between the adapter and the robot, or in this case the female tool changer and the robot. This analysis was previously made in the second chapter, section "WWR50".

5.3 Blister

After analyzing the connections aspects as well as the main dimensions of the gripper it's important to examine the blister itself which is displayed in figure 64.

This blister his made of plastic, which makes it fairly light-weighted. It holds within plastic parts, it has a rectangular shape with 457mm and 719mm of side, which makes it a fairly large product to pick.

One of the goals of this tool is to pick up the blister, which has specific plastic projections where the gripers can cling on to in an easier way.

There are two different kinds of projections within the blister, the first projections are located along the centerline, there are rectangular and have 24mm by 48mm of side, a height of 20mm, and are inclined of 41 degrees. There are 3 of these projections in the centerline and are separated between them by 120mm.

The other plastic projections are located 219mm away from the lengthier wall, they are also rectangular, but have 43mm by 20mm on side and an inclination of 75 degrees, and have a height of 20mm. These projections are separated between them by 120.5mm.

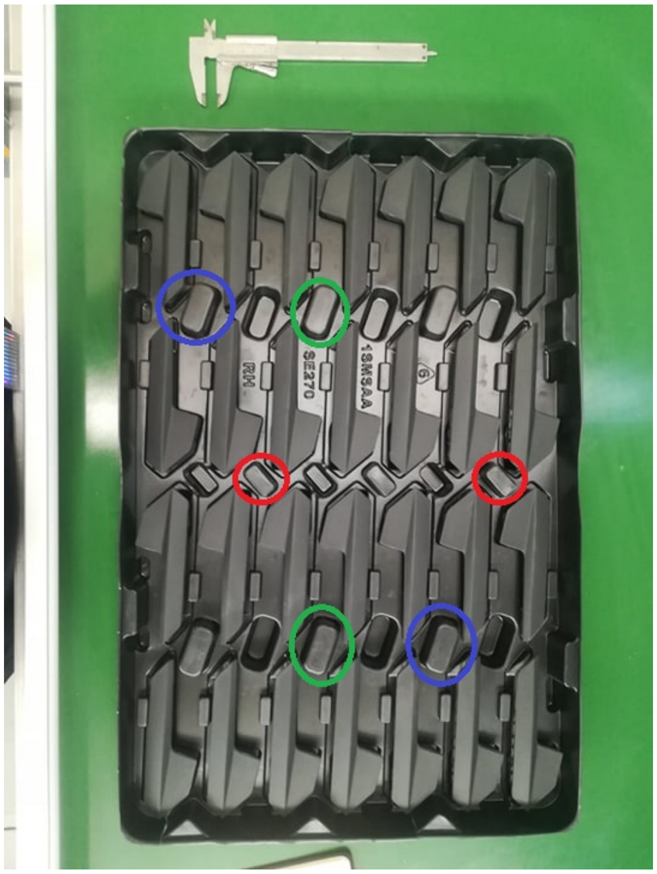


Figure 64: Blister with grabbing locations highlighted

There are several possible positions for the gripper to hold the blister. By experimenting it was considered the optimal position to be along the middle of the blister, being picked up in the locations highlighted by the red circles, in figure 64.

It was also considered to be possible to pick it up in the places highlighted by the blue and green circles, these determinations were made by experimentation.

5.4 Adapter

5.4.1 Adapter Considerations

As it was previously mentioned, the adapter has the function of connecting the grippers with the robot. Each gripper has a 90 degree angle between them. There are different aspects to consider when designing this adapter such as what kind of material will be used to make it, how it will be made, the configuration that is going to be used, and other constructive aspects.

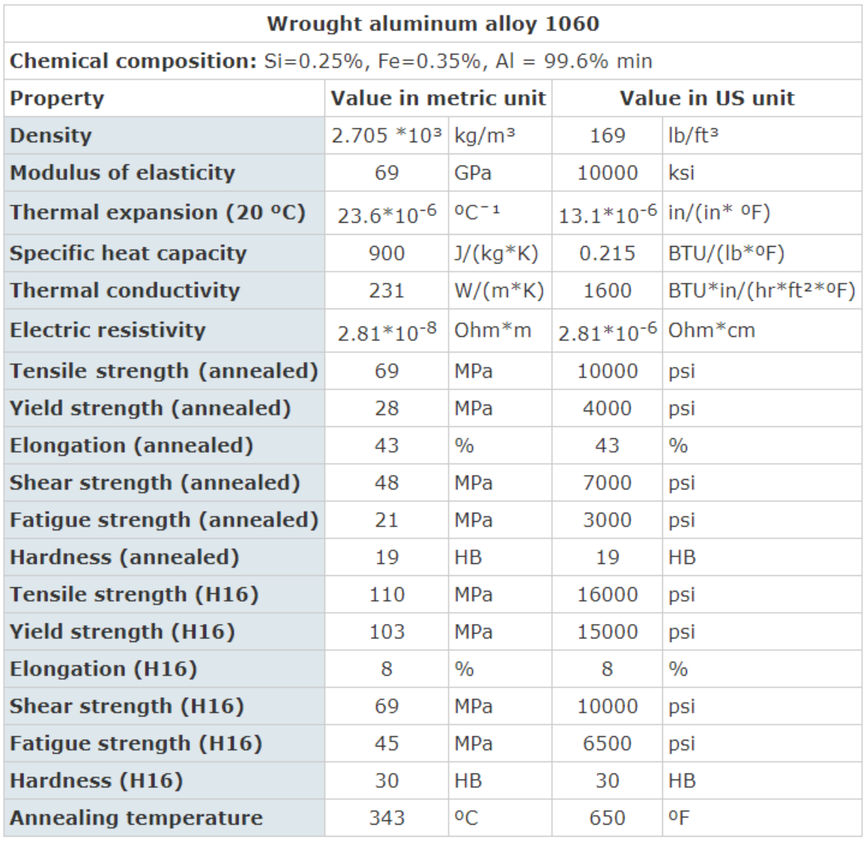
It was initially considered to divide the adapter into 4 different parts:

- the first would be the main structure of the adapter, that is, the structure that will connect the robot to the gripper.
- the second part is a disc that connects the main frame with the adapter structure
- the last 2 parts are discs that connect the grippers to the main frame.

The idea of doing these parts separately instead of just producing one single part is to save money with eventual machining operations as this is, in most cases, far less costly. Machining different parts also add flexibility to the model, as it is possible to combine different main frames with different discs for different grippers.

It was decided that the adapter would be built in aluminum, this choice was heavily influenced by the lightweight of this material, as well the relatively cheap cost. It is noteworthy to mention that it also has good mechanical properties such as high ductility and resistance to corrosion, however, it is a hard material to weld, therefore when idealizing the structure or eventual connections, the welding difficulty, must be taken in consideration. It is possible to see in table 8, the main properties of the aluminum alloy used [19].

Table 8: Aluminium alloy proprieties



Wrought aluminum alloy 1060				
Chemical composition: Si=0.25%, Fe=0.35%, Al = 99.6% min				
Property	**Value in metric unit**		**Value in US unit**	
Density	$2.705 *10^3$	kg/m^3	169	lb/ft^3
Modulus of elasticity	69	GPa	10000	ksi
Thermal expansion (20 ºC)	$23.6*10^{-6}$	ºC^{-1}	$13.1*10^{-6}$	in/(in* ºF)
Specific heat capacity	900	J/(kg*K)	0.215	BTU/(lb*ºF)
Thermal conductivity	231	W/(m*K)	1600	BTU*in/(hr*ft^2*ºF)
Electric resistivity	$2.81*10^{-8}$	Ohm*m	$2.81*10^{-6}$	Ohm*cm
Tensile strength (annealed)	69	MPa	10000	psi
Yield strength (annealed)	28	MPa	4000	psi
Elongation (annealed)	43	%	43	%
Shear strength (annealed)	48	MPa	7000	psi
Fatigue strength (annealed)	21	MPa	3000	psi
Hardness (annealed)	19	HB	19	HB
Tensile strength (H16)	110	MPa	16000	psi
Yield strength (H16)	103	MPa	15000	psi
Elongation (H16)	8	%	8	%
Shear strength (H16)	69	MPa	10000	psi
Fatigue strength (H16)	45	MPa	6500	psi
Hardness (H16)	30	HB	30	HB
Annealing temperature	343	ºC	650	ºF

The adapter shouldn't have gaps between any of its components, and if it has they should be kept to a minimum. This is due to the tool being applied to a robot and requiring a calibration to be used, which means that the tool in the robot is initial calibrated so that the center of gravity and moments of inertia are known for control purposes. Any change in these parameters can drastically change the robot's performance, as it can impact final positions, speed, etc. which is to be avoided.

Keeping in mind the previously mentioned topics, the discs of the adapter will most likely need to be machined as this is the most economical method to build them.

The main frame, however, can be machined or it can be made using normalized profiles that afterwards are connected by dismountable connections. The later solution has advantages in price and longevity but can lift significant problems when designing the frame due to the lack of existent profiles that can be used for this project.

5.4.2 Mainframe

The main frame of the adapter can be considerd has rectangle triangle for dimension purposes(figure 65).

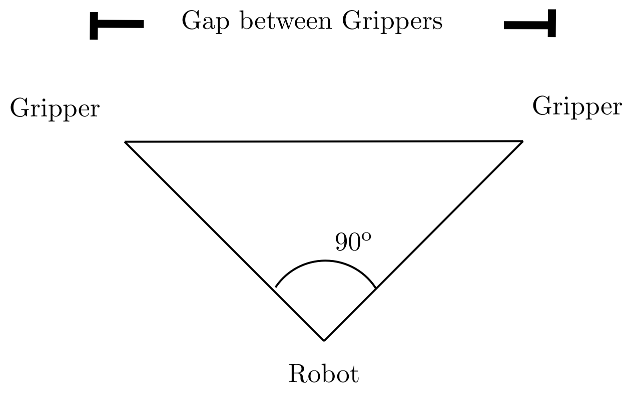


Figure 65: Schematic drawing of the grippers

To determine the minimum possible gap of the gripper, we need to consider all of the components and respective dimension that are relevant, from the base of the adapter to the tips of the gripper:

This procedure is done to assure that it is possible to build an adapter, that can hold the blister, in the desired locations.

- medium length of the grippers:

$$L = \frac{fingertipsclosed + fingertipsopen}{2}; \tag{12}$$

$$L = \frac{148.9 + 162.8}{2} = 155.85mm \tag{13}$$

- Thickness of the coupling = 13.9mm
- Thickness of the disc that connects the coupling with the adapter = 15mm

The full length of the triangle's cathetus is 184,76mm, and, using the Pythagorean theorem, we can determine the hypotenuse of the triangle which represents the minimum gap (G) between the grippers:

$$x^2 + x^2 = z^2; \tag{14}$$

$$G = \sqrt{184.7^2 + 184.7^2}; \tag{15}$$

$$G = 261.2mm; \tag{16}$$

There is no necessity of considering the thickness of the discs that connect the female tool changer and the adapter when determining the hypotenuse of the triangle, has the system will be connected there. There is however a necessity for a small gap between them so the discs don't collide.

To determine this gap it would be first necessary to determine the diameter of the discs which is not yet possible to be determined, as this will depend on several factors most importantly how the adapter would be built, however, an initial supposition can be made for this value.

The minimum possible radius for the disc would be around 30mm, due to the holes of the adapters being located 25mm from the center and have a 3mm radius. It shall be considered an initial radius of 37.5mm or a diameter of 75mm, as this is the diameter of the coupling.

The angle between the discs will again be of 90°, so the gap between the discs can again be calculated using the Pythagoras theorem:

$$G = \sqrt{37.5^2 + 37.5^2} \tag{17}$$

$$G = 53mm \tag{18}$$

Adding the gap caused by the size of the grippers, and the gap caused by the minimum distance between the discs, we can determine that the minimum possible gap between the fingertips of the grippers when this is coupled to the adapter is 314.2mm.

As it was said before, the red circles represent the ideal location for the gripper to grip in to, however, these 2 places have a gap of 262.8mm between them, therefore it is not possible to hold the blister in the said locations.

The second best location, again, determined by experiment, is highlighted by the blue circles. The gap between those locations is approximately 420 mm, which makes this location ideal for the gripper to hold on to.

5.4.3 Adapter Simple Solution

The initial case for this solution was meant to be a starting point for the next options. It was intended to be rather simple, all it needed to be guaranteed is that the grippers grip into the right place with the 90^o angle. This solution won't have any kind of requirements regarding occupational space or maneuverability.

The first part of the adapter to design should be the disc that connects the main frame with the robot, this should be made as simple as possible.

In the previous chapter it was considered that the diameter of the disc should be of around 75mm, and the thickness of the adapter 15 mm, this value will remain for the design.

It is now necessary to consider the connection method between the tool disc and the adapter as well as the robot and the adapter. Starting with the first situation we consider a simple connection with a single screw in the center of the disc, this screw should be a low cap screw and a small cavity should be machined in the interior of the disc, to prevent any contact of this with the flange of the male tool changer.

The connection method between the main frame and the adapter has already been determined by the female tool changer, it should be 4 m6 screws set apart by a 90 degree angle between each other set on a radius of 25mm. It is possible to see in figure 66 an initial solution for this disc.

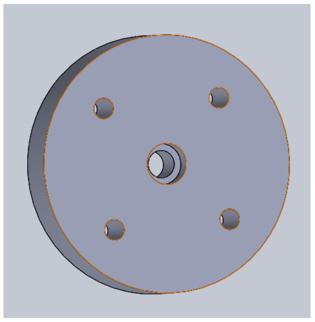


Figure 66: Adapter with hollow body

The second part of designing the adapter consists on the dimensioning of its main frame, as it was previously said, the gap that is desired is 420mm, so it necessary to determine the size of the main frame that will allow such gap

Considering that the length of the plastic projection in the blue circles(20mm), it is possible to consider that the gripper will have to be opened 20mm as well.

The gripper in the fully open position has a gap between its fingertips of 85mm, and when closed it has a 0mm gap between his fingertips. Using these values it is possible to determine the percentage of opening of the grippers when the gap between his fingertips is of 20mm

$$openingpercentage = \frac{20}{85} * 100 \tag{19}$$

$$openingpercentage = 23(\%) \tag{20}$$

This means that the grippers will be 77% opened when grabbing the blister, so it is possible to acknowledge that the grippers will be at 77% of its maximum height when grabbing the blister.

$$heightofthegrippers = ((Height_(max) - Height_min * height_percentage + Height_min)mm \tag{21}$$

$$heightofthegrippers = ((162.8 - 148.9) * (1 - 0.23) + 148.9)mm \tag{22}$$

$$heightofthegrippers = 159.6mm \tag{23}$$

It was determined that the total height of the grippers, coupling and the discs is 225.7mm, and so the gap between them has the values of 266.8mm.

The gap that the mainframe must place the gripper is:

$$Grippersrequireddistance = Distancecausedbytheadapter + distancecausedbythegrippers \tag{24}$$

$$Distancecausebytheadapter = 421 - 266.8 \tag{25}$$

$$Distancecausebytheadapter = 182.2mm \tag{26}$$

There are different ways of designing the main frame now that the necessary gap is known, it is possible to just build a very long parallelepiped with his tips cut in a 45º angle. It is also possible to propose more complicated structures.

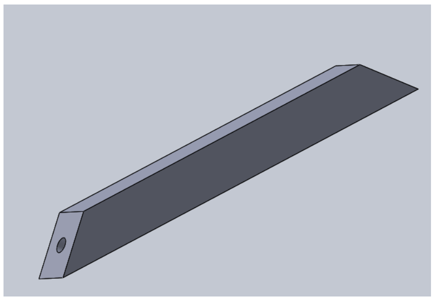
It is necessary to specify a thickness of the main frame, and determine how this will be secured with discs responsible for making the connection between the main frame and the coupling of the grippers. The connection method between the connection disc that couples the main frame with the robot was already determined.

A very simple method of bonding the connecting discs that connect the gripper with the adapter was considered, just a single m8 screw in the center of the discs.

The thickness of the adapter needs to be small enough to assure that the connection can be made, this is the frame does not cover any places where there will be connections, but large enough to guarantee the structural integrity of this adapter.

The distance between the center of the holes that will allow the placement of the screws to make the connection between the gripper and the main frame, and the main frame with the robot, is 29mm so it will be considered a thickness for the adapter of 25mm.

It was proposed two different solutions that can be seen in figures 67 and 68.


(a) Frame one, first view

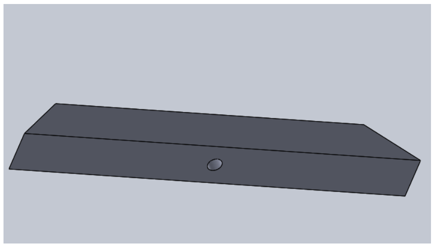
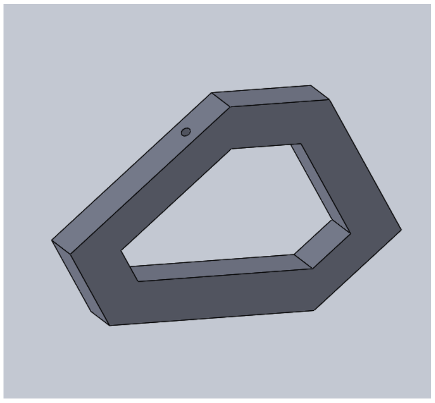

(b) Frame one, second view

Figure 67: Main Frame one.


(a) Frame two, first view

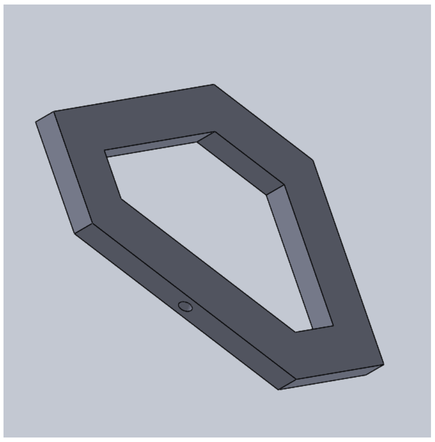

(b) Frame two, second view

Figure 68: Main Frame two

The first solution represents the parallelepiped solution, it can be made using profiles or machined. The second structure is more complex and was design to have larger structural integrity. This solution can still be built using profiles or machined.

Notice that the second structure was built using a hollow body, this was done to keep the possibility of making this structure using profiles. If the structure is machined, it is recommended to use a full-body, as the price would be smaller, and the structural integrity would be higher.

To decide which solution can be applied a small study regarding the structural integrity when holding the grippers was done, however, the connection discs between the coupling and the main frame need to be dimensioned.

It is now necessary to define the connection discs between the grippers and the main structure, this is again made to be as simple as possible. Analyzing the coupling we can see the necessity of the 4 m6 holes with the centers distanced 25mm from the center of the coupling, there is also a single m8 hole made in the adapters. The discs can be seen in figure 69.

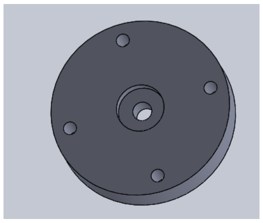
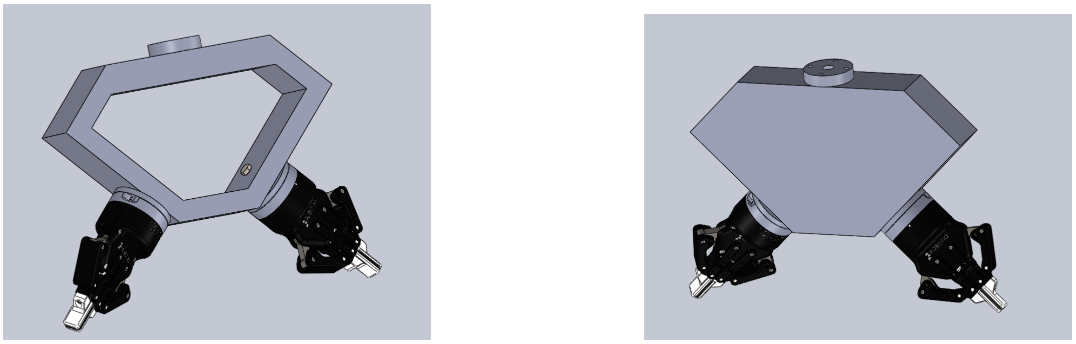


Figure 69: Adapter with hollow body

With all the different components defined it is now possible to make an assembly of the dual gripper adapters, using both the proposed frames, which can be seen in figure 70 and figure 71


(a) Hollow Body Frame.

(b) Full Body frame

Figure 70: More complex structure frame

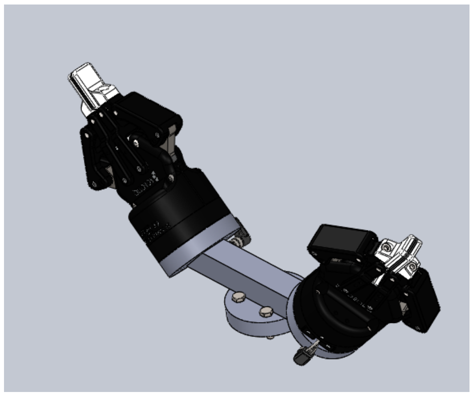
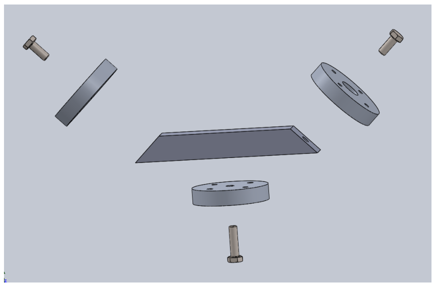
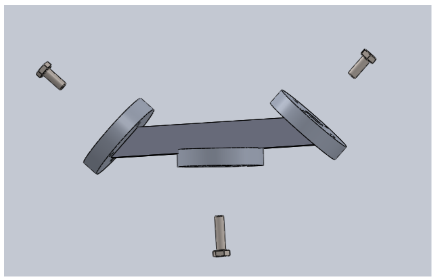


Figure 71: Simple Adapter

It is quite important to analyze how the assembly of the adapter will be done, the first part is to fix the different disc to the main frame of the adapter, followed by connecting the gripper to the main frame, lastly it is necessary to connect the now assembled tool with either the robot or the female tool changer, according to the intention of the operator.


(a) Step one


(b) Step two

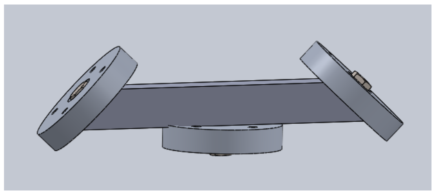

(c) Step three

Figure 72: Simple adapter assembly process

In the figure 72, it is possible to see the assembly of the parallelepiped main frame solution. The assembly of the solution with the more complex geometry is similar to this one.

5.5 Main Frame Comparison

The next part of the work is to consider the different aspects of the adapter and decide which is the best option for the current work. there are many different aspects to consider, the price is the main concern. Both disc adapters use the same disc to make the connection with the robot and the grippers, so this analysis will be focused on the main frame. If the main frame is done using profiles the price is not significantly different between both main frames, if, however, it is decided to machine the structure, the price of the more complex geometry is higher.

In the maneuverability and stability of the structure, the simplest solution is advantageous as it has a smaller area and smaller weight therefor it is significantly simpler to maneuver inside the robot.

5.5.1 Comparative Structural Analysis

The last parameter to analyze is the structural integrity of the adapter. This parameter, although important, is not the most relevant because the adapter will be used in a very light weighted blister, however, it is important to study the different forces applied to the structure of the adapter, without being necessary to make a to exhaustive investigation regarding the subject.

The first part of the test of this subject is to define what kind and where should the loads be applied, as well as determining what kind of software is going to be used to make the required tests.

As this is not a very thorough test it was used the *solidworks* tool to test the statically forces. It was also considered a significantly disadvantageous scenario for both the adapters. It was decided to run two tests, in the first one the robot will be holding a blister with 20 kg, and the force will be evenly distributed between the grippers, the second one will consider that one of the grippers is holding a product weighting the same 20 kg.

It is possible to see in figures 73 and 74 a representative figure of the forces applied in both tests.

It was decided to use this weight (20 kg) due to being the maximum payload of the robot and also 4 times superior to the heaviest object that the gripper could hold, which gives a good assessment of the dual gripper adapter structural integrity.

It is also important to assess the boundary condition of the adapter, in this case, the adapter is encastreted in the robot, that means that all the rotation and translations are blocked in fixation location. The connection disc adapter between the fist of the robot and the adapter is the fixation location.

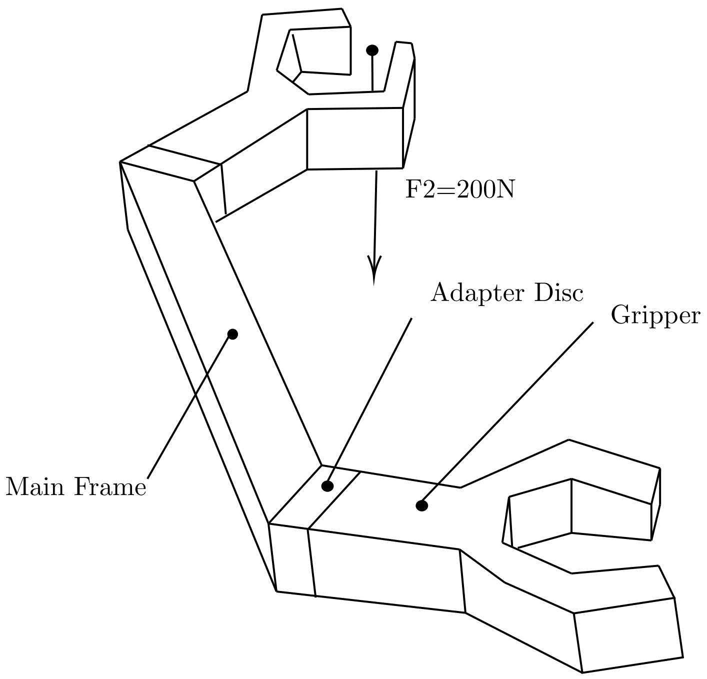


Figure 73: Structural analysis for blister

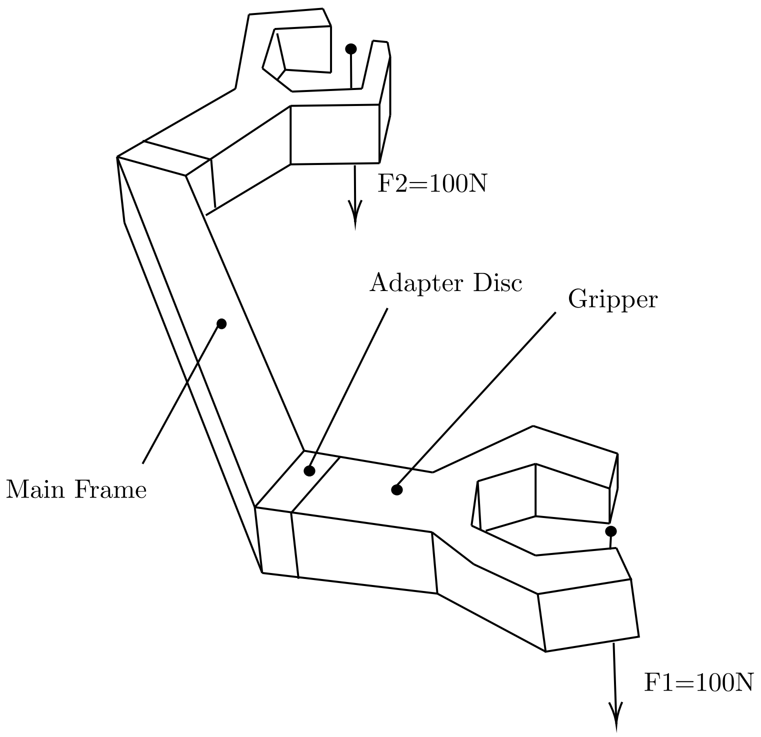


Figure 74: Structural analysis for single part

Using this program we first need to define the material of the adapter.

All the components that were design were made in aluminum, however, there is no information about all the materials that were used to build the grippers, therefor the simulation could be compromised.

It is possible to transport the forces applied to the fingertips directly to the main frame, by just considering the same force and the momentum caused by the force applied at the fingertips. It is also possible to simulate the geometry of the gripper in aluminum and then applied the forces directly there, however, this case consumes far more computational power and it can prove an impossible task for the computer, thus, it was decided to test using equivalent forces.

As it was previously said the gripper is not designed to hold weights superior to 5 kg, so it would never be able to hold the 20 kg that we are going to test it with, however, and solely for the purpose of this simulation, it will be considered that it is capable.

Like this if it was considered a force applied in the fingertips of 200N which roughly corresponds to a weight of 20 kg and we also considered the distance between the mainframe and the gripper, with is fingertips closed (the worst possible case scenario), which corresponds to a value of 191.7mm it is possible to calculate the equivalent momentum in the base of the adapter. The forces representation can be seen in figure 75.

$$T = L * F \tag{27}$$

$$T = 0.1917 * 200 \tag{28}$$

$$T = 38.34Nm \tag{29}$$

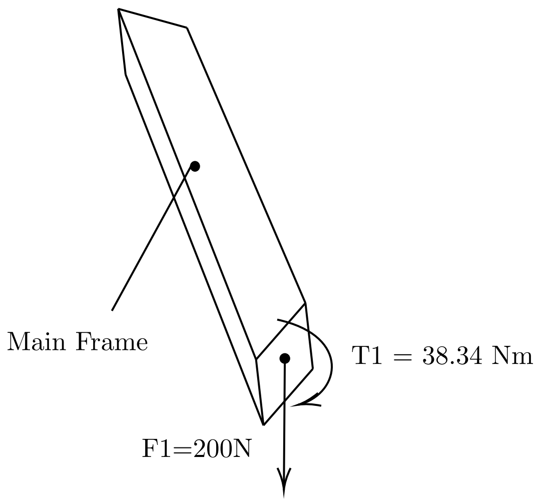


Figure 75: Equivalent Forces

The equivalent forces applied to the structure using 2 parts instead of one, could be determined by the same method, and, as the applied forces in the second case are half as big as in the first case, the resultant force will be half as big as well.

In a static system it is necessary to define the boundary conditions, in this case, the system is built in the robot, this means that there is no rotation or translation regarding any of the axis, and the forces resultant of this blockage are applied in the back of the connection disc between the robot and the adapter. Once, all the settings have been defined it is possible to start the simulation.

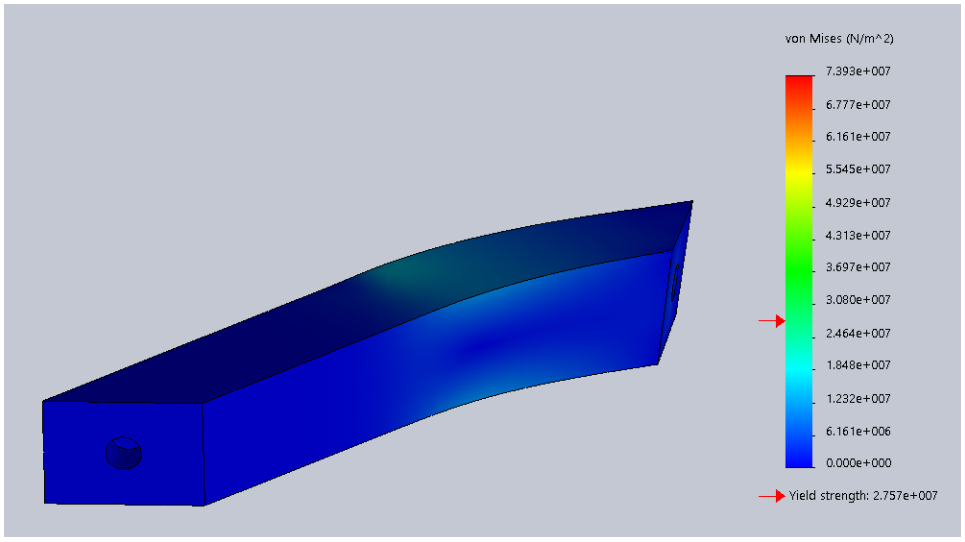


Figure 76: Static test, 200N, main frame one, first view

Analyzing figure 76, which corresponds to 200N applied on the right side of the adapter, with the force facing downwards, it is possible to see that the main frame won't hold the imposed weight, the yield strengths of the material is inferior in determined areas namely the hole where the screws will be holding the main frame to the adapter.

There are different ways of reducing this problem, the first one consist of increasing the thickness of the adapter, this, however, is not possible due to the way the adapter is assembled.

The second solution would be to add another screw to the disc that connects the main frame with the robot, which would allow a better distribution of the forces instead of concentrating on only one point.

This test was repeated for a weight of 20 kg, divided by both grippers. Analyzing the second test, represented in figure 77 and 78 made to the main frame, it is possible to see that this time the adapter is being stressed underneath the yield strength, therefore this adapter can be considered.

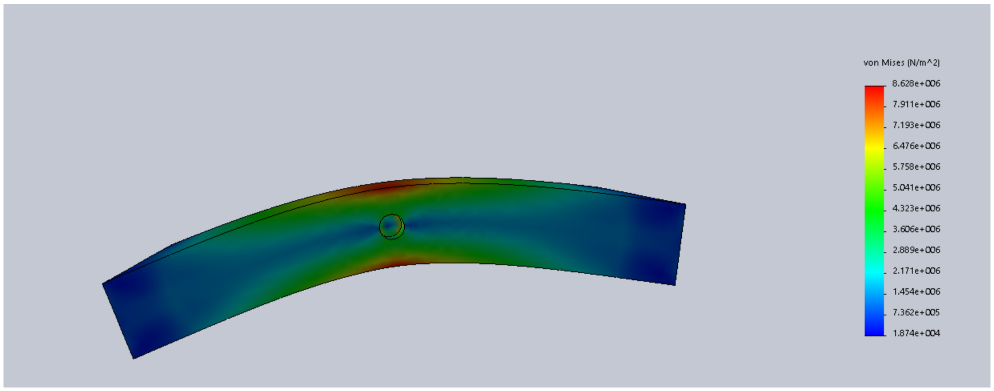


Figure 77: Static test, 100N, main frame one, first view

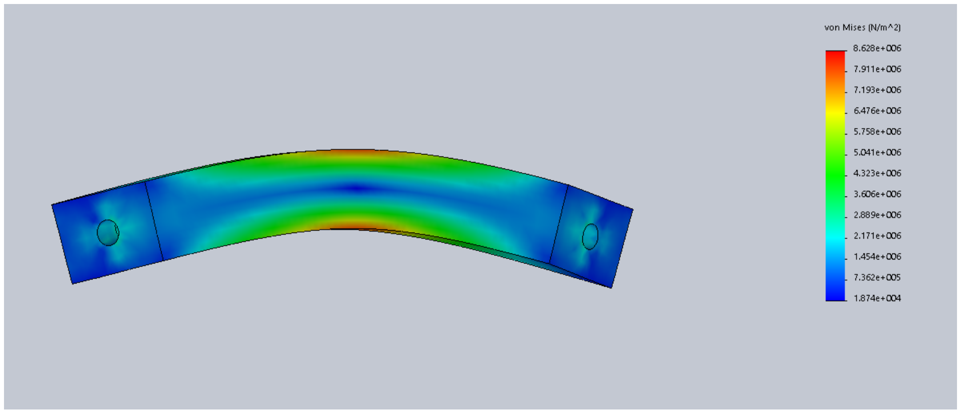


Figure 78: Static test, 100N, main frame one, second view

As it was expected with more evenly distributed forces the main frame will have a better response to the solicitation.

Figure 79 we see the second main frame response when subjected to the 20 kg divided by both his grippers. It is possible to see that this frame can also hold the divided weight, and has a better response in comparison to the simpler frame.

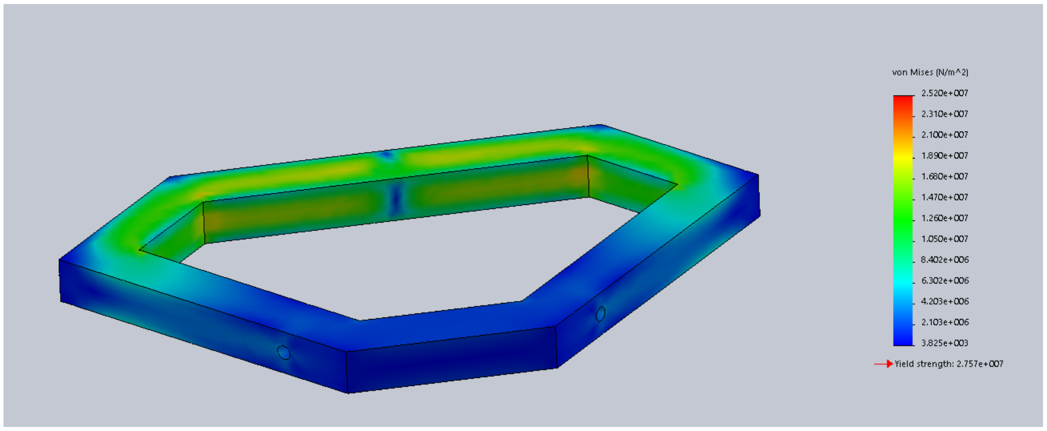


Figure 79: Static test, 100N, main frame two

Testing the structure with the concentrated charges, reach a similar result that the one that was obtained using a more parallelepiped adapter, seen in figure 80. The main frame, in this case, is less stressed than using the simpler frame, however, there are still points above the yield strength.

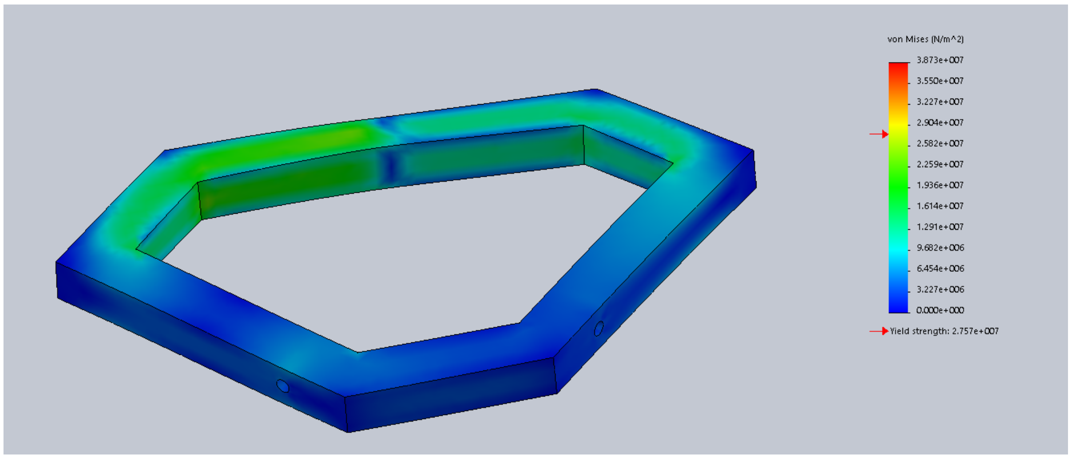


Figure 80: Static test, 200N, main frame two

It was then decided to test the second solution with a full-body, to asses if this frame could withstand the 20kg charges. Analyzing figure 81 it possible to conclude that it still cannot hold the weight, this is due to the high amount of tension concentration in the hole that holders the main frame. This problem is small if we considered that this adapter will be built to lift very small weights, however, and keeping in mind a flexible approach, it was decided to propose an alternative solution to the adapter disc.

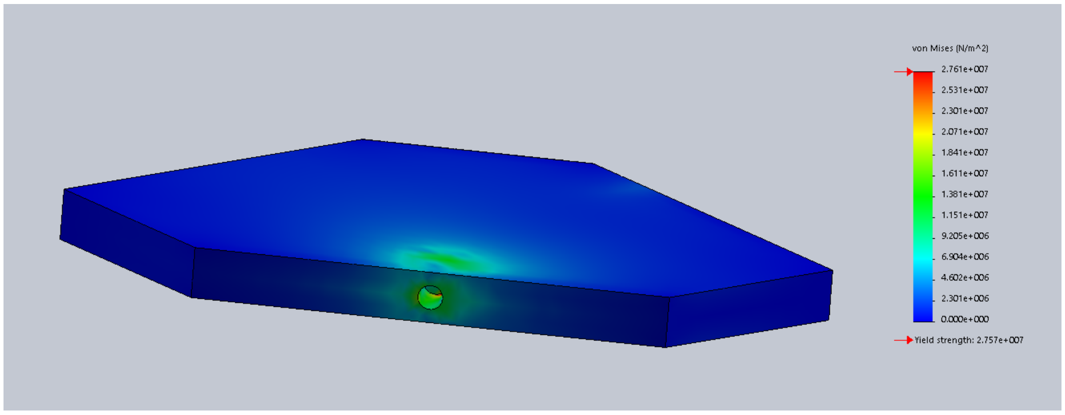


Figure 81: Static test, 200N, main frame two, full body

The second disc that was proposed is in all similar to the previous option in regards of the connection between the disc and the robot, however it was proposed a better distribution of charges by simply adding a two more screws, this way the stress will be more evenly distributed, this disc can be seen in figure 82.

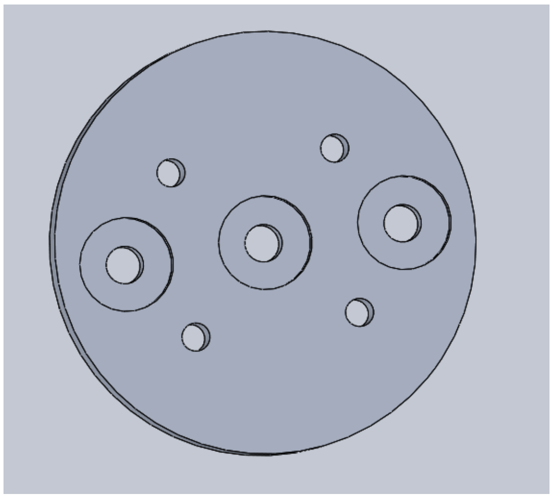


Figure 82: 3 holes disc

The main frame will now have three different holes where the screws will be attached, there will significantly better strength distributions, as can be seen in figure 83 and 84.

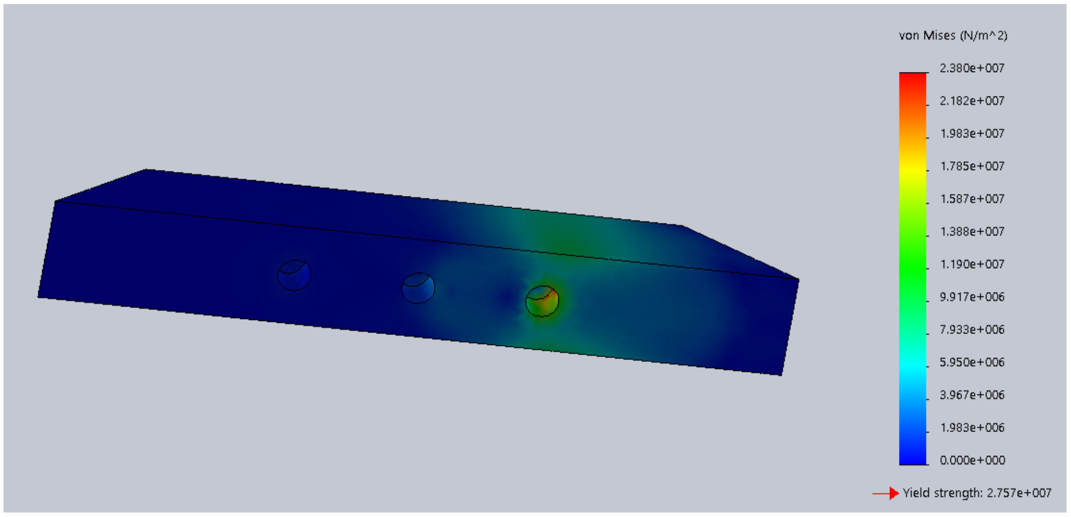


Figure 83: 3 holes disc, 200N,static test, view one

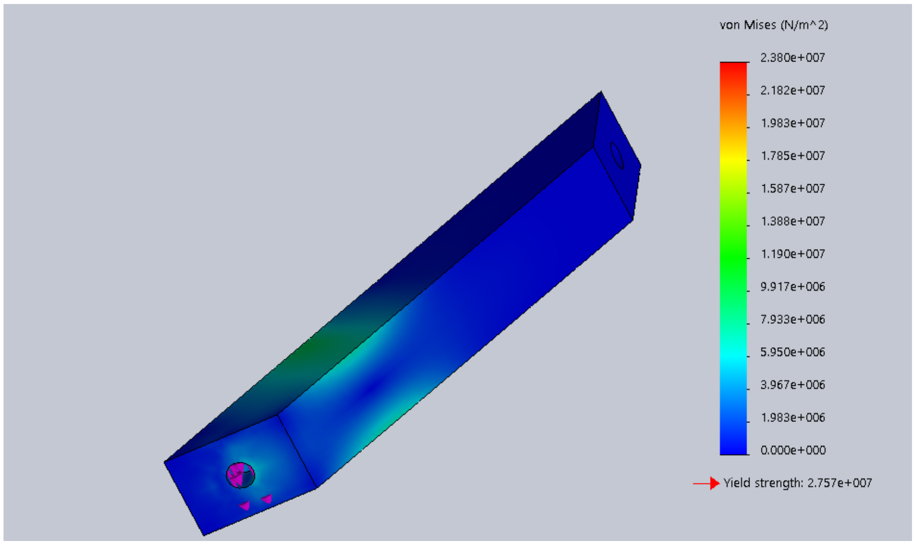


Figure 84: 3 holes disc, 200N,static test, view two

Analyzing the different parameters within the solution is not quite as simple as it appears to decide which is the best main frame for the adapter, the price and structural stability are the two main factors that need to be considered when choosing the adapter.

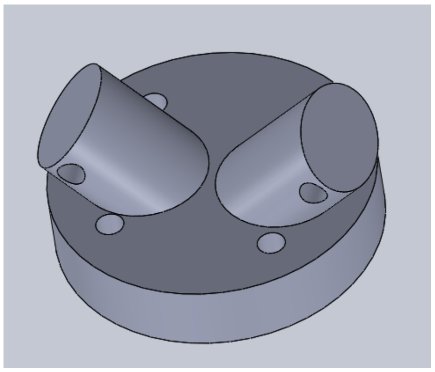
The less expensive adapter that can be built is parallipipied one, the one with the best structural integrity, on the other hand, is the one with the more complex geometry and a full-body, however using a different kind of disc it is possible to have very good mechanical properties in the parallipiped adapter, this change in the disc adapter, however, will increase the price of machining this disc, therefore it may no longer be a viable alternative to a full-body structure. Keeping this in mind, and reconsidering the low weight the structure of the adapter will be submitted to, it was decided to go with the more simple frame, and the more complex disc, due to being a good compromise between structural integrity and price, it was also decided to build this solution using profiles.

5.6 Telescopic Adapter

It was then proposed a second solution, which aimed for higher flexibility, it was desired an adapter with a higher range of dimensions, this is an adapter that would allow placement of the grippers at different distances, allowing an of array dimensions for the gap between them. The adapter would keep the same main objectives, this is, the adapter still needs to allow a 90º between the grippers and would also need to be able to pick each part individually.

Considering the different factors involved, it was proposed a solution built like a telescope, a solution that had a fixed cylinder or a set of tubes that would allow them to slide inside each other and be fixed within certain points. This kind of mechanism would require several parts to be implemented.

The consideration previously made for the adapters is still valid, therefore the first part of designing this adapter is to dimension the disc that will perform the connections of the adapter. This disc will necessarily be machined so the price of machining operations needs to be taken into consideration. Therefore it was proposed a disc with two solid cylinders projected with a 90 degree angle between them, the disc could be machined individually and the cylinders purchased individually. The discs can be seen in figures 85.


(a) Conection disc View one

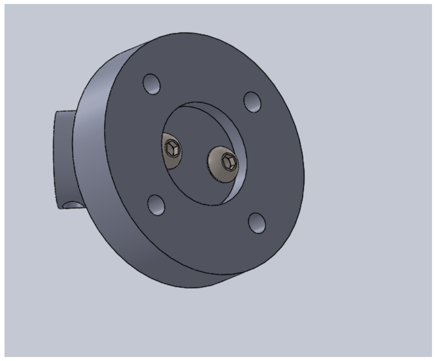

(b) connection disc View two

Figure 85: Connection discs

In the next part of the project, it was defined a hollow tube, which is normalized and it is coupled with the cylinder presented in the base via screws and nuts, in this hollow tube it was made three holes, that will later determine the expansion that the adapter will have. The assembly of this tube in the connection disc can be seen in figure 86.

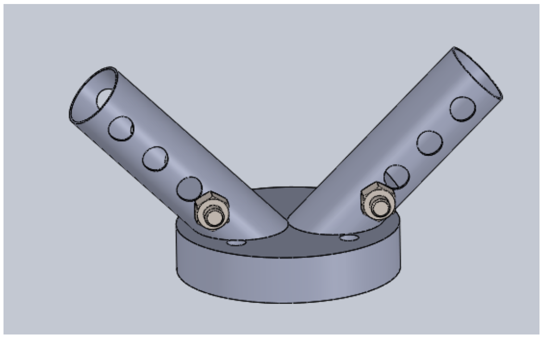


Figure 86: Base with hollow tubes

It will then be mounted in this the hallow tube a cylinder (figure 87), which can either move freely or be fixed in position, depending on if the screw is fastened or not. The fixation of the tube will be responsible for giving the adapter is flexibility, as this will determine the size of the gap of the adapter.

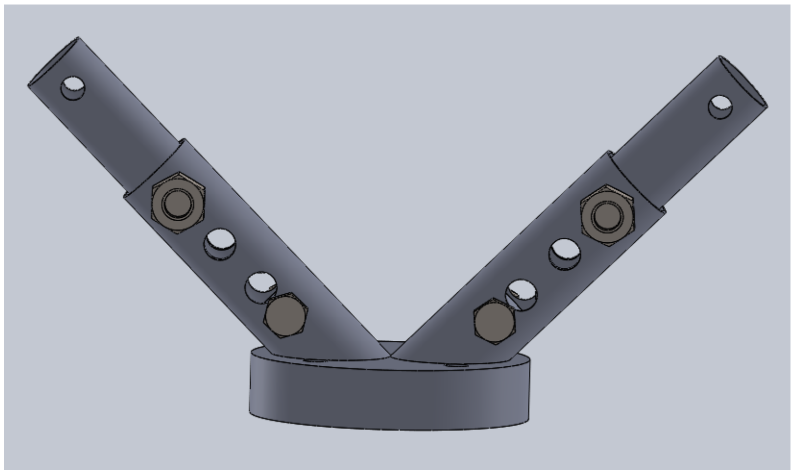
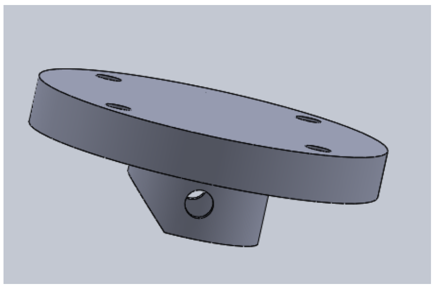


Figure 87: Telescopic adapter without grippers on

The last part of the telescopic adapter will be the disc that will connect the gripper with the main frame, or in this case, the tubes that will constitute the main frame. This disc is in all similar to the previously designed ones, except the projection within the disc, which will allow the connection with the hollow tube.

This projection will be a hollow cylinder and will be connected with the tube by a screw, it will have part of its end cut, which will allow the disc to be directly connected with the base to creating the minimum possible gap, that this adapter can have. This connection disc will also be machined, and can be seen in figure 88. Due to this metal projection will be the most expensive component in the adapter, as it will require more material and more machining operations.


(a) Adapter Disc

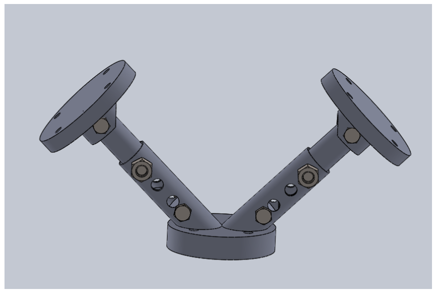
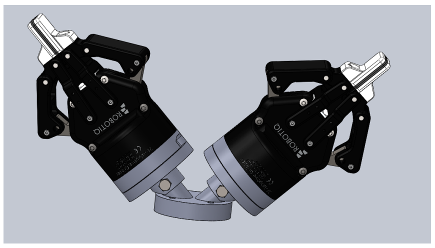

(b) Telescopic adapter with adapter disc

Figure 88: Telescopic adapter with adapter disc

With all the components dimensioned it is now possible to do a final assembly as the grippers and the coupling will remain the same, there exists a vast number of configurations possible for this adapter, and there could be even more with an increase of tube size or change in any of the locations of the holes. With the current configuration, the farthest position has a gap of 435mm, which is well over the required gap, in the mid-position a gap of 421mm is obtained. This is the gap that it is necessary to pick up the blister in the desired locations.

Notice that the holes made in the fixed tube were somewhat arbitrary, it is possible to consider a tube where the only pre-defined configuration is the one with the middle hole, any further configuration could be defined by the user by simply drilling the tube in the location that would allow the desired gap, which could be calculated as it was previously saw.

The tubes can also be removed from the adapter(figure 89. a), and the connection could be made directly between the disc and the base of the adapter, this configuration corresponds to the minimum possible gap. The maximum possible gap can be seen in figure 89. b


(a) telescopic adapter with the least possible gap

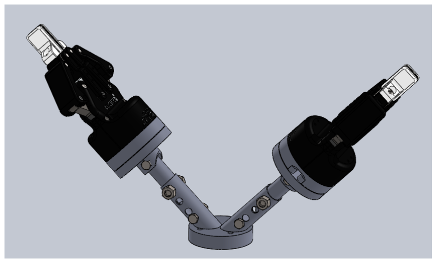

(b) telescopic adapter with the larger possible gap

Figure 89: Different gaps for the telescopic adapter

This mechanism significantly improves the flexibility of the adapter because instead of having a fixed gap between the grippers, the gap can be changed to better respond to different needs, furthermore, the blaster itself is quite light and therefore the adapter doesn't require a very significant mechanical resistance.

The price of the telescopic adapter should not be much higher than the previously saw solutions, with the exception of the discs, all the parts or normalized and therefore are easy to acquire, the coupling disc between the grippers and the main frame will require more machining and its price should be higher, but not in a very significant way.

This adapter, however, will have a large problem with the high number of gaps that exists in the main frame, as it was previously said a robot is a very precise instrument and requires a very accurate calibration, with such a high number of parts, all connected by dismountable part, the adapter will have a high range of small unwanted changes in is structure, this fact could make the usage of this adapter impossible, it will depend on the robot and the disposition of the user in recalibrate the system more often to validate the solution.

6 Production Line Simulation

6.1 Production Line Definition

The last part of this work was to build a station that would simulate and validate the different elements developed throughout this dissertation.

To test the usefulness of this application it was proposed a terminal part of a production line, where different products would be sorted out by an ABB robot. To separate these parts the robot would need to use different tools. In figure 90 can be seen a representative figure of the product line.

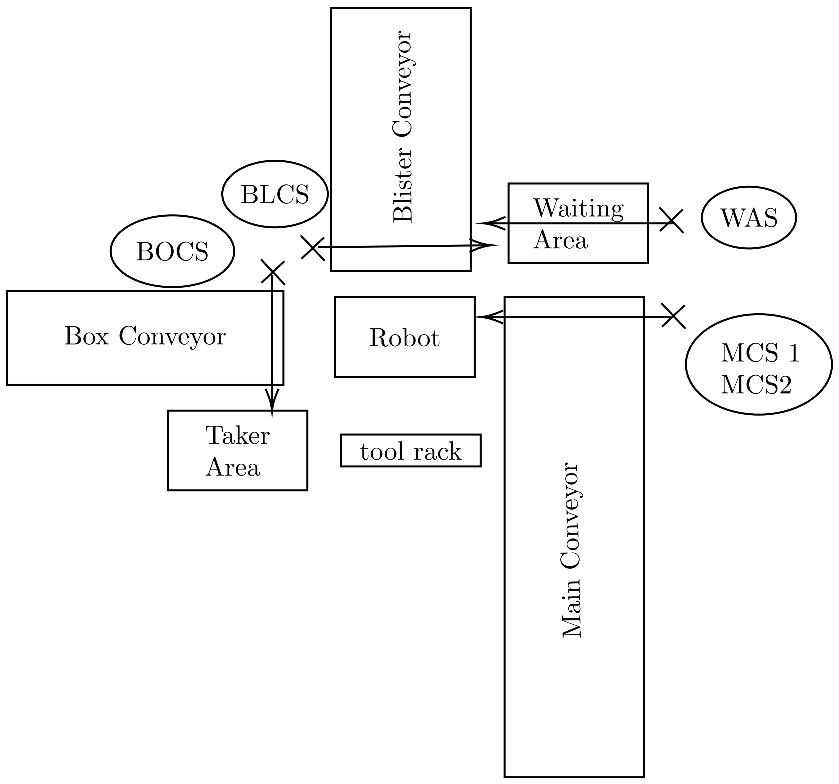


Figure 90: Line Schematics

It was considered two different types of products, the first one was the blister that was analyzed in the previous chapter, this blister should be picked with the tool developed for this purpose, namely the telescopic one. the second product was a simple flat box that needed to be picked using a suction gripper. This product would reach the robot working area through a conveyor.

Once the product had reached the robot it would be identified through sensors to decide which tool should be used. If the robot wasn't holding the correct tool, it would place the tool that it was currently being held and pick the tool that was necessary to sort out the product, if the robot wasn't holding any tool it would just pick the tool and then sort out the product, and if the robot was holding the correct tool it would just pick the product. This behavior can be summarized by grafcets represented in figure 91 and figure 92.

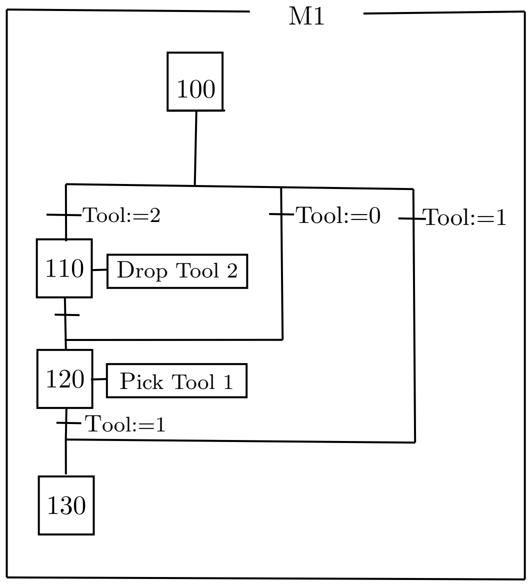


Figure 91: Macro M1

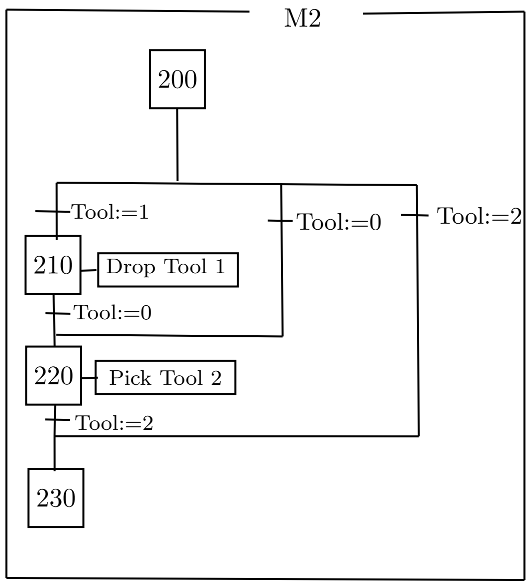


Figure 92: Macro M2

Once the robot had picked the correct product it would place it on a convoy that would take away the product to a different line, depending on the type of product, the robot could place the product in one of two conveyors one for the blister and one for the box. If for any reason, there was already a product on the beginning convoy, the robot would place the picked product in a waiting area and continue it's normal work until the line has been cleared. Once the line has been cleared, and the robot was not in use, the product in the waiting area would be sorted in the right line. If a new product would need to be sorted out and the correspondent line was in the use, and the waiting area was full, the product would be placed in an area where a worker would come to pick it up. This functionality can be summarized in the grafcet in figure 93.

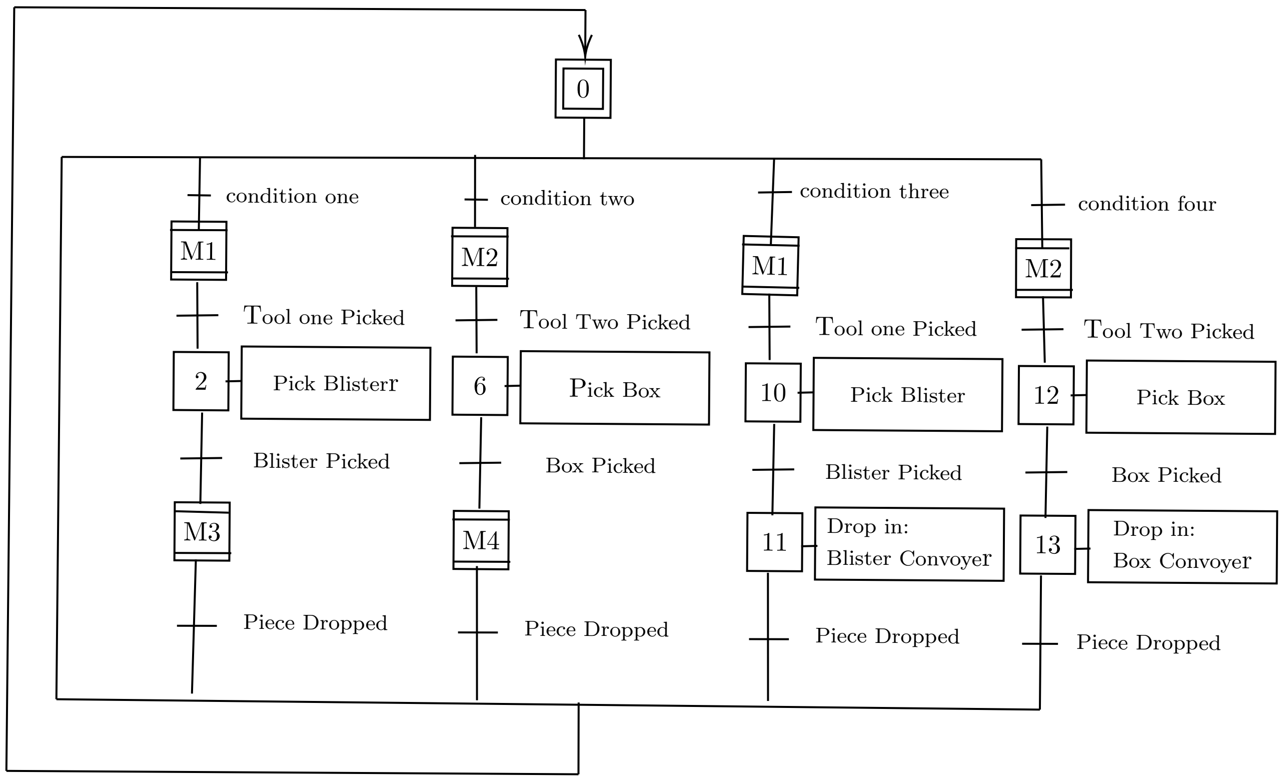


Figure 93: Main Grafcet

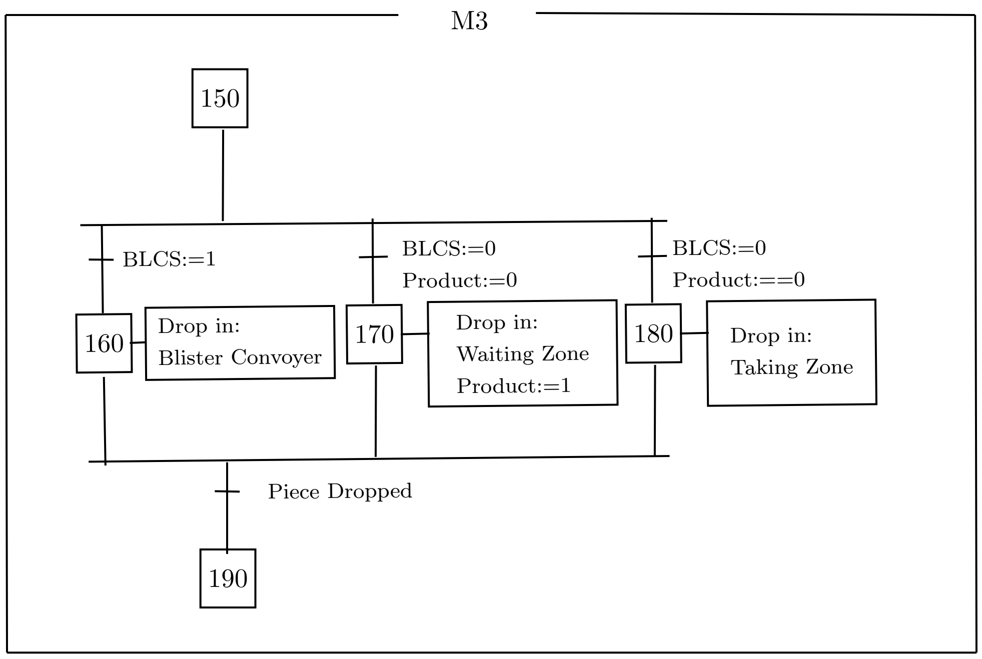


Figure 94: Macro M3

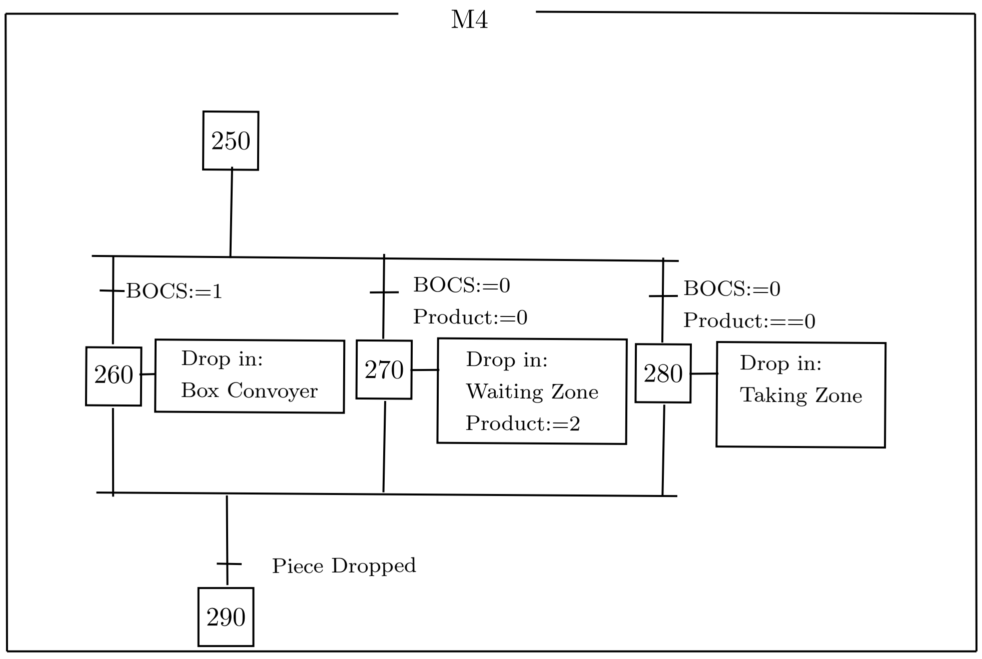


Figure 95: Macro M4

Where condition one is MCS1:=1, MCS2:=1, condition two is MCS1:=1, MCS2:=0, condition3 is MCS:=0, MCS2:=0, product:=1, condition three is MCS:=0, MCS2:=0, product:=2.

To make the line have the correct behavior it was necessary the implementation of different actuators, sensors, and areas that can be seen and are identified in figure 90.

To identify the product it was used 2 different retroreflective sensors at the end of the main conveyor, these are placed in the same plane, but at different heights, with the main conveyor sensor 1(MCS1) below the main conveyor sensor 2(MCS2). If both sensors are activated, which means their value would be 1, the product would be a blister, if only the MCS1 sensor was activated the product would be a box, if no sensor has been activated then there was no product in that area.

There is also a retroreflective sensor at the start of the blister conveyor, identified by the initials "BLCS" (Blister Conveyor Sensor), to determine if there is a product in the location where the robot should drop the blister. There is a similar sensor in the box conveyor that has the same functionality but for the box product.

Finally, there is a retroreflective sensor at the waiting area to determine if there is or not a product there, this sensor is identified by the initials "WAS" (Waiting Area Sensor).

The architecture of the line was simulated in robotstudio, it is possible to see in figure 96 a 3D representation of the line.

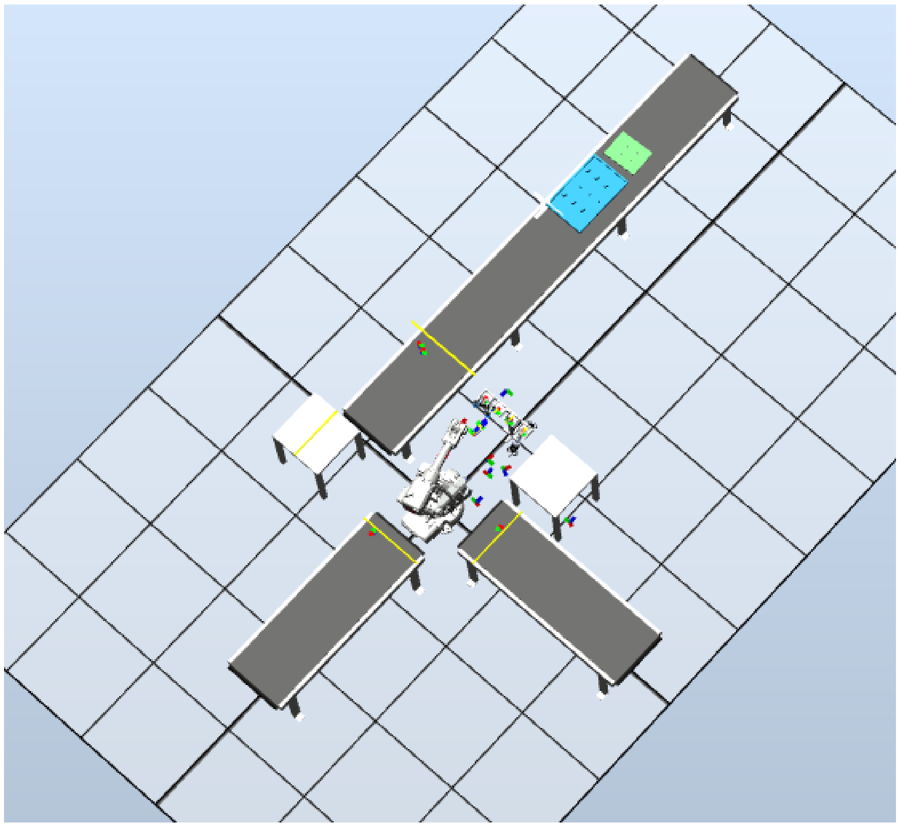


Figure 96: Station

All the sensors can be seen in table 9.

Table 9: Sensors

Sensor	Name
WAS	Waiting Area Sensor
MCS1	Main Conveyor Sensor 1
MCS1	Main Conveyor Sensor 2
BLCS	Blister Conveyor Sensor
BOCS	Box Conveyor Sensor

6.2 Production Line Implementation

The behavior that was necessary for the line to work was analyzed in the previous section, some elements were added, including conveyors and retroreflective sensors. The conveyors were imported from a library within robotstudio and simulated using smart components. The main conveyor was by default activated and was deactivated when a robot was on the robot picking area. The box and Blister conveyor were always activated.

The retroreflective sensors were also simulated using smart components.

The tool rack was built and imported using solidworks, it only has 2 different tools, the dual telescopic gripper, placed in the second position, which corresponds to the position ideal to pick the blisters, and a suction cup developed by *Robotiq*.

The application to implement the tool changer system was installed using the procedures identified in the subsection ”Tool Changer Implementation”. To apply the behavior described in the grafctes, and analyzing the procedures that were identified in chapter 6, it was first necessary to teach the different targets to the robot, this was done manually using the jog function of the robot.

It was defined 5 different targets, the first two targets correspond to the location and orientation that the robot should visit to pick the blister or the box. The third and fourth targets were the location inside the box conveyor and the blister conveyor, where the robot should place the product. The fourth corresponds to the location where the robot should drop the product in the waiting zone, and the last target corresponds to the location where the robot should drop the product in the taker zone.

These are the targets that are external to the tool changer application, afterward, the robot was jogged to the location of the tools and using the calibration stage screenmaker application these tools were defined. Once the tools have been identified they were tested and adjustments were made to assure that the tools could be safely removed from the holder.

Using the grafcets saw in the previous section it is possible to directly implement the code in robotstudio, and test to see how well it is working.

7 Conclusions

7.1 Contributions

There was one main objective for this dissertation, which as the development of an effective automatic tool changer that could be used in ABB robot, this single objective was divided into different parts to create a flexible and robust system. To validate the system was simulated a simple production line.

It is possible to conclude that the were proposed different holder systems for the tool to be held in, allowing a more broad range of tools to be used in a production line, and accumulating different requirements for the different lines. It was also proposed a different position for the tools to be holdered in two and a vast array of options and prices assuring to better suit the need of the user.

It was also developed an application to facilitate the programming of the routines to pick and place the tools, this includes an HMI that simplifies the process of creating the path to pick or place the tools in position. Furthermore, the developed application is not limited by the holders proposed in the first chapter, allowing the user to develop and use is own holders that better fulfill his needs, and use the application to still ease the programming of the necessary routines.

It was analyzed the building process of a tool to use with this tool changer system, and the different steps that need to be taken to develop another tool to use in a production line.

Finally, the application was tested, and it was possible to confirm a significant simplification in the programming part of the work, as the user doesn't need to be familiarized with the ABB programming language to use the HMI that was created, and the routines created can be implemented with a lot of ease.

7.2 Future works and recommendations

The next step in this project should be the manufacturing of the holder and the adapters to test the developed solutions in a real-life case, it is also necessary to implement the ABB application in a real robot.

The website built to share the developed application could also be improved, increasing the information about this system.

The biggest and most challenging step regarding this system is the implementation of this system using the algorithms and methods used to develop this system and expanding to different types of robots.

8 References

[1] I. E. Commission, “White paper - factory of the future,” 2015. [Online]. Available: https://www.iec.ch/whitepaper/futurefactory/

[2] Yaskawa, “Yaskawa robot software,” Yaskawa Electric Corporation, Tech. Rep., 2014. [Online]. Available: https://www.yaskawa.eu.com/fileadmin/Products/Software/Flyer/Flyer_YaskawaRobotSoftware_E_06.2014.pdf

[3] Fanuc, “Fanuc robotics ipendant,” Fanuc Robotics, Tech. Rep., 2007. [Online]. Available: http://www.fanucrobotics.com.mx/Productos/Controladores/data_sheet/iPendant.pdf

[4] F. May, “How to create fanuc hmi panels using the panel wizard,” 2015. [Online]. Available: https://www.dmcinfo.com/latest-thinking/blog/id/9123/how-to-create-fanuc-hmi-panels-using-the-panel-wizard

[5] ABB, “Application manual screenmaker,” ABB Robotics, Tech. Rep., 2009, document ID:3HAC035956-001.

[6] A. Owen-Hill, “The 5 minute guide to use any end effector with robodk,” 2018. [Online]. Available: https://robodk.com/blog/robot-end-effector-guide/

[7] A. Owen-Hill., “How to use tool changers with robodk,” 2019. [Online]. Available: https://robodk.com/blog/tool-changers-with-robodk/

[8] U. Robots, “Ati qc-11 automatic tool changer.” [Online]. Available: https://www.universal-robots.com/plus/accessories/ati-qc-11-automatic-tool-changer/

[9] RSP, “Rsp products.” [Online]. Available: https://www.robotsystemproducts.com/en/products/tool-changers

[10] PAL systems, “Pal rsi and pal rtc sample prep and injection,” PAL systems, Tech. Rep., 2014.

[11] AGI, “Robotic tool changer,” 2016. [Online]. Available: http://www.agi-automation.com/product-category/robotic-tool-changer/

[12] Zimmer Group, “Tool changers installation size wwr50,” Zimmer Group, Tech. Rep., 2015.

[13] Schunk, “Sws,” 2019. [Online]. Available: https://schunk.com/de_en/gripping-systems/series/sws/

[14] O. Safety and H. Administration, *OSHA Technical Manual.*

[15] ABB, “Product specification irb 2600,” 2019, document ID:3HAC035959-001.

[16] D. Taslakova, “Positioning accuracy and repeatability of a class of technological robots,” *PROBLEMS OF ENGINEERING CYBERNETICS AND ROBOTICS*, pp. 99–105, 1997.

[17] P. Abreu, “Manual de utilização robotstudio,” *FEUP*, 2018.

[18] Robotiq, “2f-85 2f-140 for e-series universal robots,” Universal Robots, Tech. Rep., 2019.

[19] M. J. F. Gándara, “Aluminium: The metal of choice,” *Materials and technology*, pp. 261–265, 2012.