ASSISTED ROBOTICS FOR FEEDING INDIVIDUALS WITH UPPER LIMB DISABILITIES

Santiago García Arango
Elkin Javier Guerra Galeano

EIA UNIVERSITY
MECHATRONICS ENGINEERING
ENVIGADO, COLOMBIA
2021

ASSISTED ROBOTICS FOR FEEDING INDIVIDUALS WITH UPPER LIMB DISABILITIES

Santiago García Arango
Elkin Javier Guerra Galeano

Bachelor's final project to obtain the degree of:
Mechatronics Engineer

Director of the project:
Dolly Tatiana Manrique Espíndola, M.Sc., Ph.D.

EIA UNIVERSITY
MECHATRONICS ENGINEERING
ENVIGADO, COLOMBIA
2021

ACKNOWLEDGMENTS

Acknowledgments 1

Acknowledgments 2

Contents

List of Figures

List of Tables

List of Equations

Abstract

The exponential growth of technology has made it possible to achieve new solutions to improve the human beings' quality of life. One of the areas that has experienced the greatest development in the last 20 years is robotics and its derivatives. Currently, there has been a significant increase in the number of people with motor disabilities in the upper limbs, including more than 10 million people with Parkinson's disease and several individuals who, due to other circumstances, have lost the mobility of their upper limbs. This group of people not only have major difficulties in the daily task of feeding, but can also experience severe problems of malnutrition and loss of self-esteem. This is why, in this project, an exploratory research will be conducted focused on the development of an active robotic solution, using Baxter Robot, which can give support in the feeding process for these individuals and, at the same time, has conditions of improvement compared to the robotic alternatives that exist in the current market.

This development will seek a positive impact for all individuals who fit within the exposed problem and a design methodology will be carried out, oriented to the search for a scalable solution, with the ability to recognize the position of the mouth of individuals through Computer Vision algorithms and with the advantage of being Open Source. It is expected that, at the end of this research, relevant advances will be generated in the development of active robotic solutions for the assistance of this population and the scientific knowledge of robotics in Colombia and the world.

Keywords: Robotics, Computer Vision, Control Theory, Denavit Hartenberg, Algorithms, Software, Anomalies, Active Feeding, Autonomous System.

Introduction

The overall population that have upper limb special needs or suffer from motor disabilities are more likely to have problems of malnutrition, a reduction in the performance in the activities of daily living, and loss of self-esteem (ICBF, 2016; Poltawski et al., 2016).

In this research project, we will dive into the design of a possible solution for these problems, using Baxter Robot as an active feeding solution that enables individuals to accomplish of the most important daily life activities: eating.

The project's scope covers some important stages for the complete design:

1. General understanding of the overall design based on human-being needs.
2. Specific planning and implementation of each sub-system of the complete design.
3. Practical experiments to validate the functionalities of the designs.
4. Final results analysis and next steps for future improvements.

Throughout this article, we will be exposing the theoretical and practical steps that were involved in the design of the Baxter Feeding Robot solution.

At the end of the article, we will expose the most important ethical and real life considerations of designing a robotics solution that interacts with human beings.

Chapter 1

Initial Steps

1.1 Understanding the problem

1.1.1 Problem's context and analysis

According to the Technical Guide of the Food and Nutrition Component for the Population with Disabilities of the Colombian Institute of Family Welfare (ICBF), this group of individuals are more likely to have nutritional problems (ICBF, 2016). Likewise, individuals with upper limb disabilities, have a significant reduction in activities of daily living (especially feeding) and may become impaired in their psychological state (Poltawski et al., 2016).

It is estimated that there are more than 10 million people with Parkinson worldwide, of which a significant percentage have critical mobility problems in their upper extremities (Parkinson's Foundation, 2020). The incidence of people with Parkinson's disability increases with the age of the individuals, but there is an approximate 4% of them who are diagnosed before the age of 50 years. According to studies conducted by the Parkinson Foundation (PF), men are more likely than women to have this disease. Another factor that is of relevance to this problem is that there is currently a higher prevalence of pain and disabilities in upper limbs in young university populations, which implies an increase in the group of people who will have joint disabilities in the future and will be more likely to present malnutrition problems due to the difficulties of performing the task of feeding by their own hands (Park et al., 2020).

This is why it is necessary to look for technological solutions that allow this group of individuals to receive support and assistance in daily tasks, especially feeding. Considering this context, a branch of robotics known as "Assisted Robotics" is of interest, where robotic systems seek to support individuals with disabilities to perform daily tasks (Jaffe et al., 2012).

Advances in assisted robotics have led to the development of various systems that seek to assist people with disabilities in the task of feeding (scooping). There are commercial robotic platforms, which seek to fulfill the need to feed patients with upper limb motor disabilities, but these have a number of limitations that restrict and limit their ideal performance. The best known commercial robotic systems for these tasks, such as Myspoon,

Bestic Arm, Meal Buddy, Mealtime and Obi, are passive in nature. This means that they do not have the ability to adapt to the dynamic conditions of the user's mouth position, nor the ability to detect anomalies in the feeding process (Park et al., 2020).

A restriction of great relevance found in commercial robotic systems is that they are not able to adapt their behavior to temporary changes in the position of the user's mouth, because their system is passive, limiting the ability to know this information that is required for a correct feeding action. Another limitation of these systems is that they fail to identify anomalies or strange behaviors in the feeding process, generating additional risks for the patient in case of any emergency or eventuality. Finally, these systems are implemented with restricted code, limiting their modifications in the source code and restricting the possibility of replicating these algorithms for the entire population that may need them.

Taking into account this worldwide problem, with the aim of improving the quality of life for individuals who have motor disability problems in upper limbs, it is important and of great relevance to seek an active robotic solution that can support the feeding process of these individuals, with a number of significant improvements over current commercial robotic systems. This solution should be able to adapt its dynamic movements constantly according to the position of the user's mouth, be aware of the environment, check for possible anomalies that may occur in it, be accessible and affordable for the population affected by their motor disabilities, be scalable to any other similar robotic system that is programmable and, similarly, should promote and seek innovation and the technological development of related robotic systems in Colombia and the world.

Based on the previous arguments, this research will aim to solve the question of: How to implement an active robotic system for feeding (scooping) patients with motor disabilities in upper extremities, using the Baxter cobot of the EIA University?

In order to find a solution for the lack of active commercial robotic systems for feeding individuals, that also have the ability to adapt their movements according to the position of the user's mouth and the conditions of the environment, based on Mechatronics Engineering, several technological solutions can be proposed to integrate the main branches of electronics, mechanics, control systems and software development, to find an optimal solution to this identified problem (University EIA, 2021).

This is why it is being proposed to develop a programmed robotic solution, which has an user interface activated by voice commands or friendly buttons, to generate a reliable and safe solution for the needs of people with motor disabilities in the upper limbs, to perform the daily activity of feeding. This solution, being open source, can be easily extrapolated to any programmable robotic platform, generating an additional step in the development of assisted robotic systems.

Considering the concept of assisted robotics and the importance of helping these individuals, the answer to provide a viable, scalable and safe solution to this problem is the usage of collaborative robots under the approach of working with humans (Guizzo and Ackerman, 2012). This can be achieved through various robotic alternatives, but it was decided to use Baxter robot from the company Rethink Robotics, which was developed under the

concept of "cobot", that means, a collaborative robot to work together with humans and is available in the laboratories of the university campus (University EIA, 2021).

Due to the arguments exposed above, we want to look for a solution with the Baxter cobot, using the internal architecture of the software components integrated with the infrastructure offered by this company. This solution must be able to integrate the development of the software architecture, computational algorithms, kinematic models, video processing with computer vision, user interface, and the necessary connections of these components for the correct implementation of the robotic solution, which will be focused on improving the quality of life of individuals who will benefit from this technology. Likewise, the system will represent a series of elements scalable to any other robotic platform with similar operating conditions and will seek to generate a positive impact on the development of assisted robotic systems for the community.

The development of a solution to this problem has multiple benefits, both for individuals with motor disabilities in the upper limbs, as well as for the scientific and medical community in Colombia. That is why, by developing this project, we want to take an additional step in the research and applications of assisted robotics, in order to seek to improve the quality of life of people with motor disabilities in the upper limbs.

This contribution can allow a person with these conditions, to perform the essential activity of feeding, without the need of having an external person performing this task. In the same way, this generates an increase in self-esteem and quality of life for these individuals. At the same time, a key factor of the project is that it will positively contribute to the pursuit of three of the global objectives of sustainable development, especially: "Health and well-being", "Industry, innovation and infrastructure" and "Reduction of inequalities" (United Nations, 2012).

In addition to the previous reasons, one of the most important motives for carrying out this project is that there is a large gap between the robotics of the leading countries in technology and Colombian robotics. This is why the project will provide a methodological approach for robotics research and a technological progress that can be replicated in any Colombian institution that has access to robotic platforms, generating a community that aims to improve robotic developments at a national level.

1.2 Research objectives

In this chapter, we will dive into the objectives of the research that are expected by the end of the project.

It is of high importance to have a general and specific planning of the objectives to be developed in the evolution of the project. The following are the structured objectives for the final scope of the solution proposed in this project to obtain the degree of "Mechatronics Engineers".

1.2.1 General objective

Implement an active feeding system (scooping) for patients with motor disabilities in upper extremities, using the Baxter cobot of EIA University.

1.2.2 Specific objectives

1. Generate the smooth paths and trajectories between the position of the tool and the user's mouth, through a face detection system, which returns the absolute coordinates of the user's mouth based on the OpenCV library.

2. Implement a control strategy for tracking the proposed paths and trajectories, through a kinematic processing, to generate the action commands to move the joints of the Baxter cobot and validate its performance with a control algorithm, using the current position of the user's mouth as direct feedback to the system.

3. Design an algorithm for anomaly or error detection in the feeding process, which provides an action signal to stop the system in case of unexpected events.

4. Integrate the sub-systems of path and trajectory generation, control strategy and anomaly detection, using the Robot Operating System (ROS) libraries and tools.

5. Develop an user interface, which enables to send the commands to start the feeding and, if necessary, to stop the process.

1.3 Reference framework

1.3.1 Historical background

Robotics was initially conceived as one of the branches of science with the greatest impact on industrial automation and processes that allow repetitive tasks to be performed. However, the concept of robotics has been expanding its scope and applications, because there are new sectors and scenarios where robots with new objectives and capabilities are becoming more frequent. Some of these scenarios are focused on improving the quality of life of human beings. Two major current scenarios of robotics are: assisted robotics and collaborative robotics (Decker et al., 2017).

Assisted robotics refers to all types of robots that have the ability to collect information from the environment, process that information, and perform tasks or actions that benefit individuals with any type of disability (Jaffe et al., 2012). Based on this definition, it can be concluded that assisted robotics enables the improvement of the quality of life of people with any disability, including the human beings with motor impairments in the upper limbs.

The existing studies and researches related to robotics focused on patient feeding have had great advances in the last 10 years. Furthermore , it can be visualized that most current works are directed to the search and improvement of systems for feeding individuals in an active way, i.e., systems that autonomously feed patients (Park et al., 2020).

In order to effectively structure and summarize some of the researches, degree papers and patents related to the topic of assisted robotics, a literature review of the last few years was carried out to collect the objectives, methodologies, solutions, strategies and conclusions of these projects. Some of the most relevant ones are presented below:

In the article "Active robot-assisted feeding with a general-purpose mobile manipulator: Design, evaluation, and lessons learned" (Park et al., 2020), published in the scientific journal "Robotics and Autonomous Systems 124", an overview of robotic systems focused on patient feeding is presented, providing a contrast of commercial platforms and those under research and development. In this study, an algorithm and software architecture based on a PR2 robotic manipulator was implemented to achieve autonomous feeding of patients with upper extremity disabilities. The characteristics, methodologies and solutions found in similar projects were presented, making a contrast between them and exposing new proposals that made use of some of the knowledge found by all previous investigations. Besides, the article reflects an exhaustive work with a good overview of previous knowledge to develop similar robotic solutions in the future, as it not only exposes the development of their research in the particular context, but also proposes the most important elements to take into account in assisted robotics projects.

Another study of major relevance to the state of the art of assisted robotics is "SAM, an Assistive Robotic Device Dedicated to Helping Persons with Quadriplegia: Usability Study" (Fattal et al., 2019). In this research, important concepts about assisted robotics and the current delimitation of various robotic applications in this area of knowledge are exposed. Likewise, the authors expose the importance of defining the factors for the suc-

cess of robotic tasks, since the evaluation of the viability of implementation, is linked to correctly defining the success of the corresponding subroutines. Likewise, several interfaces that can be implemented for quadriplegic users are shown and their advantages and disadvantages are contrasted.

Additionally, by researching related studies, a very important element was identified in the development of a robotic system that interacts with individuals: the detection of anomalies. To give context to this feature, two research articles were analyzed, which were "A Multimodal Execution Monitor with Anomaly Classification for Robot-Assisted Feeding" (Park et al., 2016) and "A Multimodal Anomaly Detector for Robot-Assisted Feeding Using an LSTM-based Variational Autoencoder" (Park et al., 2018). From these scientific papers, we gained a deeper insight into more than 5 mathematical methodologies for strategically detecting anomalies. Similarly, they expose analytical procedures that are of great importance in the structure of algorithms that focus on searching for anomaly patterns, enabling us to have these as a starting point to develop new strategies for future robotic implementations.

Regarding patents, the United States patent called "Mobile human-friendly assistive robot" (Rensselaer Polytechnic Institute, 2017) was mainly reviewed. In this publication, it was possible to identify a viable structure for the communications architecture developed in assisted robotics projects, enabling the selection of the principal and secondary components in the infrastructure to be developed for robotic projects.

A very interesting article for the understanding of assisted robotics projects is "Robots for humanity: Using collaborative robots to help people with disabilities" (Chen et al., 2013). In this document, it is exposed the relevance of simplifying the graphical user interface, through the use of mathematical tools such as elliptic coordinates and an interface compatible with sound commands. Similarly, the authors propose a strategic error detection system by translating the forces implemented by the robotic system in different mathematical approaches.

1.3.2 Theoretical background

Primarily, it is important to delimit the suggested solution within a specific context, explaining the use cases for the conditions of the technology applied and the people who will benefit from it. A first group of people who benefit are individuals who suffer from advanced Parkinson's disease and are unable to perform smooth feeding movements due to tremors, shaking or bradykinesia in their hands (Mayo Clinic, 2020). Likewise, there is a large number of people with motor disabilities in upper limbs that fail to be categorized into a specific group of individuals, but because of their age conditions, diseases, symptoms or accidents, they have significant difficulties in the daily process of feeding and can be highly benefited by the technological advances that this solution provides.

There are a number of preliminary concepts of critical importance for the understanding and development of the overall project. The most relevant concepts and simple contextualization of these by branches of study are presented below.

1.3.2.1 Robotic concepts

Robotics is a field of engineering that has significantly impacted the technological development of manufacture, industrial, medical, security, and human support industries, among others (Genesis Systems, nd). In order to have a solid foundation of the theoretical and mathematical foundations of robotic systems models, it is necessary to go into detail on some concepts and terminology related to the subject. Some of these concepts and terminologies will be discussed in more detail below.

1.3.2.1.1 Transformation homogeneous matrices

A transformation matrix is an important mathematical tool to generate expressions of linear systems that have a combination of translational and rotational motions in space. Given the reference systems {A} and {B}, with unit vectors that meet the dextrorotating criteria of a linear system, a rotation matrix can be defined as a matrix representation that compacts the rotations that must be performed in space to bring the reference frame A to B by the three unit vectors expressing the directions of these spatial changes ${}^{A}_{B}\hat{X}$, ${}^{A}_{B}\hat{Y}$, ${}^{A}_{B}\hat{Z}$ (Craig, 2004).

To generate the resulting rotation matrix between systems {A} and {B}, we proceed to create a matrix with the following structure:

$$ {}^{A}_{B}\mathrm{R} = \begin{bmatrix} {}^{A}_{B}\hat{X} & {}^{A}_{B}\hat{Y} & {}^{A}_{B}\hat{Z} \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{1.1} $$

1.1: Rotation matrix general structure

As show in equation 1.1, the rotation matrix structure is a 3 by 3 matrix with orthogonal conditions with determinant 1. These properties are useful to compute some important matrix multiplications for different types of linear problems.

The next relevant conceptual step for the creation of a homogeneous transformation matrix is the translation. To represent a translation from a coordinate system to a point P, at least one vector indicating the relative translations of each of the unit vectors governing the orthogonal basis is required $\hat{X}$, $\hat{Y}$, $\hat{Z}$:

$$ {}^{A}\mathrm{P} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \tag{1.2} $$

1.2: Position translation vector structure

Finally, when a mathematical strategy is required to represent both the translation and rotation of a system in space, this transformation can be expressed as a transformation matrix with the following structure:

$$ {}^{A}_{B}\mathrm{T} = \begin{bmatrix} {}^{A}_{B}\mathrm{R} & {}^{A}\mathrm{P} \\ \emptyset & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & P_x \\ r_{21} & r_{22} & r_{23} & P_y \\ r_{31} & r_{32} & r_{33} & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.3} $$

1.3: Transformation matrix general structure

1.3.2.1.2 Forward Pose kinematics

The Forward Pose Kinematics (FPK) transform are all those equations that relate the behavior of a robotic system, taking into account as input parameters the values of each of the joints associated with the degrees of freedom, allowing to obtain as output the absolute Cartesian coordinates for the robot's end effector (Craig, 2004).

There are two main techniques for obtaining the Forward Pose Kinematics (FPK) of a robotic system, the geometrical method and the analytical method. In the geometrical method, a set of equations obtained from the model of the robot in the Cartesian space is formulated and a series of expressions is sought to represent the absolute spatial coordinates of the global system, that is, the expected position of the end effector. For this method, it is common to refer to the desired point as "Ptool" (Craig, 2004).

The analytical approach is a systematic methodology where rigorous steps must be followed to obtain the transformation matrices that govern the robotic system. This approach is divided into three main stages, which are: locating the axes of rotation or translation of each degree of freedom (DOF), defining the origin and direction of rotation of the main axes of each degree of freedom and, finally, locating the missing axes to obtain a series of reference systems that fulfill the desired translations and rotations (Barrientos et al., 2007).

The following is an example of the FPK of a revolute-revolute planar robot:

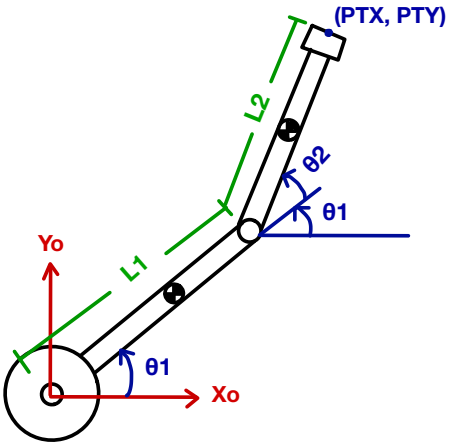


Figure 1.1: General schematic model for RR planar robot.

By solving the FPK of the RR robot, from fig[1.1], it can be shown that the equations governing the position and orientation of the robot tool are the following:

$$
\begin{aligned}
P_{Tx} &= L_1 \cdot cos(\theta_1) + L_2 \cdot cos(\theta_1 + \theta_2) \\
P_{Ty} &= L_1 \cdot sin(\theta_1) + L_2 \cdot sin(\theta_1 + \theta_2) \\
\alpha &= \theta_1 + \theta_2
\end{aligned}
\tag{1.4}
$$

1.4: Forward Pose Kinematics results for RR robot example.

1.3.2.1.3 Inverse Pose Kinematics

The Inverse Pose Kinematics (IPK), are all those equations that relate the behavior of a robotic system, having as input parameters the absolute Cartesian values, allowing finding the equivalent values of each one of the degrees of freedom (DOF) of a robotic system (Craig, 2004).
It is common for a robot to have different kinds of actuators for the linear and rotational movements of its joints, so it is of high importance to obtain a series of expressions that can describe the possible values of these joints, for a specific position of the robot's tool.

Based on the following schematic, we can find the IPK of the given RR robot example:

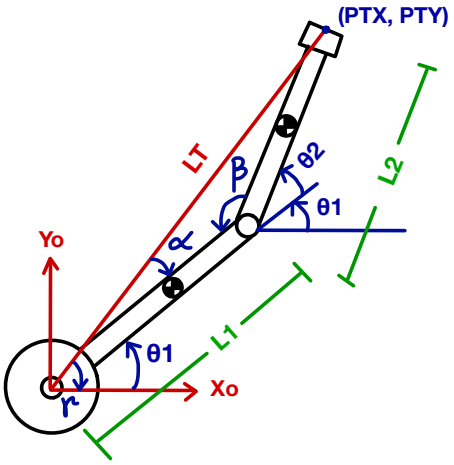


Figure 1.2: General schematic with the angles convention for RR planar robot.

The equations that describe the IPK of a planar RR robot, as shown in Fig[1.2], are shown as follows:

$$
\begin{aligned}
L_T &= \sqrt{P_{Tx}^2 + P_{Ty}^2} \\
\gamma &= atan2(P_{Tx}, P_{Ty}) \\
\alpha &= acos\left(\frac{L_T^2 + L_1^2 - L_2^2}{2 \cdot L_1 \cdot L_T}\right) \\
\beta &= acos\left(\frac{L_1^2 + L_2^2 - L_T^2}{2 \cdot L_1 \cdot L_2}\right) \\
\theta_1 &= \gamma + \alpha \\
\theta_2 &= \beta - \pi
\end{aligned}
\tag{1.5}
$$

1.5: Inverse Pose Kinematics results for RR robot example.

It's important to state that in robotic systems with more than one degree of freedom, it is possible to reach the same point in space through different configurations of its joints, usually known as "elbow up" and "elbow down" for the same joint. For this reason, it should be kept in mind that there could be more possible solutions, as more degrees of freedom are available.

1.3.2.1.4 Path planning

Path planning refers to the strategy of generating spatial routes to achieve the movements of a robotic system between two points in space, using techniques to find the most appropriate route, taking into account the constraints of the environment. The methods to generate the paths may vary according to the type of robot and the dynamic conditions that it has to face in the context of the surrounding environment. It is relevant to mention that, in order to generate the planning of a given path, a series of preliminary elements of the environment must be previously known through integrated sensors that are able to translate their signals into valuable information for the robot (Klančar et al., 2017).

1.3.2.1.5 Higher-order polynomials trajectories

Soft trajectories are parametric vectors that define all the points in space that a robotic manipulator must follow, in order to change its position in space from an initial point to an final point, having as input the desired position, velocity and acceleration conditions. There are two mathematical trajectories that allow an acceptable robotic performance, these are the 3^{rd} and 5^{th} order trajectories. The 5^{th} order trajectories are those that allow the control of any kind of dynamic system, with a correct control of variables such as jerk (Craig, 2004). In this section, it is important to understand in depth the mathematical expressions that govern a behavior with 3^{rd} and 5^{th} order trajectories.

The 3^{rd} order trajectories fulfill the following mathematical expressions:

$$
\begin{aligned}
\theta(t) &= a_0 + a_1 \cdot t + a_2 \cdot t^2 + a_3 \cdot t^3 \\
a_0 &= \theta_0 \\
a_1 &= \dot{\theta_0} \\
a_2 &= \frac{3(\theta_f - \theta_0)}{t_f^2} - \frac{2\dot{\theta_0}}{t_f} - \frac{\dot{\theta_f}}{t_f} \\
a_3 &= \frac{2(\theta_0 - \theta_f)}{t_f^3} + \frac{(\dot{\theta_0} + \dot{\theta_f})}{t_f^2}
\end{aligned}
\tag{1.6}
$$

1.6: Equations for third order trajectories.

Similarly, the 5^{th} order polynomial trajectories, follow these mathematical equations:

$$
\begin{aligned}
\theta(t) &= a_0 + a_1 \cdot t + a_2 \cdot t^2 + a_3 \cdot t^3 + a_4 \cdot t^4 + a_5 \cdot t^5 \\
a_0 &= \theta_0 \\
a_1 &= \dot{\theta_0} \\
a_2 &= \frac{\ddot{\theta_0}}{2} \\
a_3 &= \frac{20\theta_f - 20\theta_0 - (8\dot{\theta_f} + 12\dot{\theta_0})t_f - (3\ddot{\theta_0} - \ddot{\theta_f})t_f^2}{2t_f^3} \\
a_4 &= \frac{30\theta_0 - 30\theta_f + (14\dot{\theta_f} + 16\dot{\theta_0})t_f + (3\ddot{\theta_0} - 2\ddot{\theta_f})t_f^2}{2t_f^4} \\
a_5 &= \frac{12\theta_f - 12\theta_0 - (6\dot{\theta_f} + 6\dot{\theta_0})t_f - (\ddot{\theta_0} - \ddot{\theta_f})t_f^2}{2t_f^5}
\end{aligned}
\tag{1.7}
$$

1.7: Equations for fifth order trajectories.

1.3.2.1.6 Assistive robotics

A collaborative robot or "cobot" is a robotic device capable of manipulating objects in cooperation with or providing assistance to humans. Thus, cobots have the ability to perform critical tasks for humans or work in conjunction with them (Edward et al., 1999). A field of collaborative robotics is assisted robotics, defined as any robotic device that performs tasks for the benefit of people with disabilities in the context of activities of daily living (Jaffe et al., 2012).

1.3.2.1.7 Robot's workspace

In the context of robotics, the workspace is defined as all possible spatial locations where a robotic manipulator is able to reach thanks to the strategic movement of its degrees of

freedom independently. The mathematical delimitation of this is a process that requires finding multiple possible spatial solutions for the maximum access points and plotting them in a simulation environment. It is common for manufacturers of industrial robots to provide a detailed review and explanation of the workspace in the robot's data-sheets, because it varies not only by geometric constraints, but also by the mechanical restrictions existing in the joint connections (Craig, 2004).

1.3.2.1.8 Baxter cobot

Baxter is a collaborative robot developed for the industrial sector by the company Rethink Robotics. It was designed with a series of sensors, cameras and control systems that allows it to work in a cooperative environment and to adapt easily to its workspace (Rethink Robotics, 2020).

Baxter was announced to the community in September 2011 and can be broadly defined as a two-armed robot with 7 degrees of freedom in each arm. Baxter was introduced to the market as a robot specialized in simple collaborative tasks, such as loading and unloading, organizing and interacting with materials (Rethink Robotics, 2015c). Likewise, being a robot that seeks to interact with humans, it has a screen that simulates the face of the Baxter cobot and additional safety protection features that will be considered in the development of the project.

Figure 1.3: Front view for Baxter cobot. Taken at EIA University Laboratory.

Considering the general specifications, it is of major relevance to have a clear understanding of each one of the degrees of freedom that the Baxter joints have and their standard nomenclatures for their operation. The following figure is an image of the official technical data-sheet of Baxter's documentation, where the seven DOF and their names are illustrated:

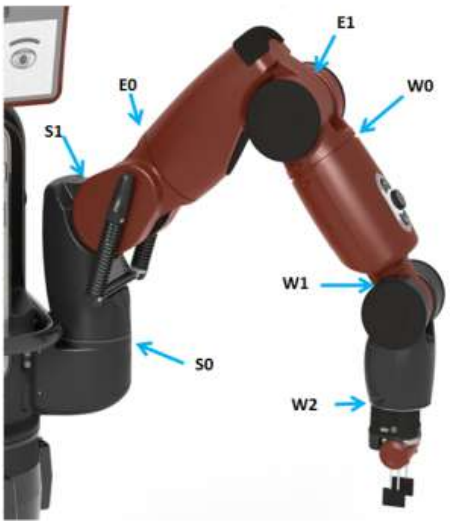


Figure 1.4: Baxter left joint names for each DOF. Taken from (Rethink Robotics, 2015b).

When working and programming the Baxter cobot, it is essential to have a good understanding of the workspace conditions and the general restrictions of each of the degrees of freedom. The following image is a visual illustration of Baxter's workspace and its numerical limits for each joint:

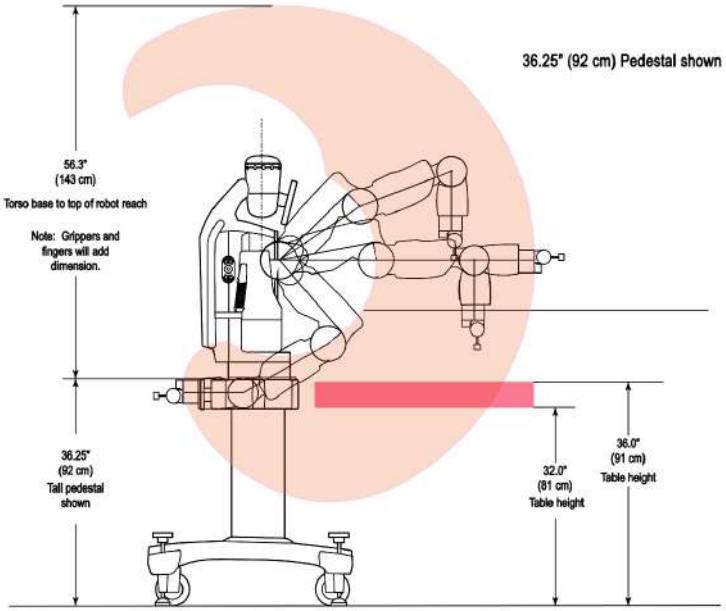


Figure 1.5: Baxter workspace side view. Taken from (Rethink Robotics, 2015e).

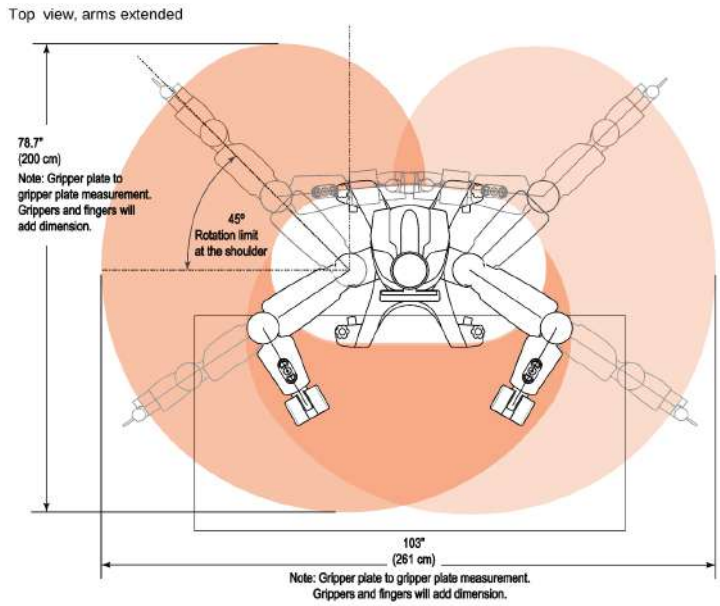


Figure 1.6: Baxter workspace top view. Taken from (Rethink Robotics, 2015e).

1.3.2.2 Software concepts

In most robotic systems, it is critical to implement a series of algorithms, data-structures and architectures that allow the consistent and safe development of the robot's actual performance. These commands must be programmed through various software schemes and strategies, which are integrated with the electronic and mechanical components that govern the robot's joints and sensors. For this reason, the development of this project should be explained in a comprehensive way, from a software approach.

1.3.2.2.1 Python

Python is a multi-platform programming language, developed by Guido van Rossum in 1989 and formalized to public in 1991. It is interpreted, with the ability to update itself dynamically. Its philosophy is based on the simplicity and readability of the codes, allowing several programming paradigms, such as: object-oriented programming, imperative programming and functional programming. It has the advantage of being open source with a significant worldwide community (Python Software Foundation, 2021).

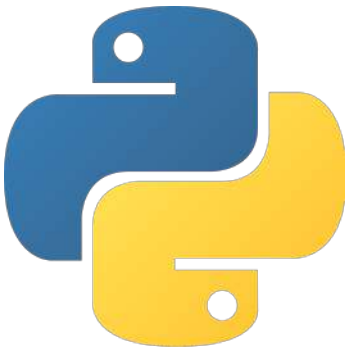


Figure 1.7: Python official logo. Taken from (Python Software Foundation, 2021).

The advantage of using a language that is open source and with a flourishing community, is that there are a large number of libraries and tools that are fully accessible, in order to simplify and increase the development of technological solutions based on them. This is why, with the implementation of a single main syntax, multiple solutions can be achieved for software development approaches, such as graphical interfaces, web backend servers, the usage of simulation environments, artificial intelligence and computer vision algorithms.

1.3.2.2.2 Robot Operating System (ROS)

ROS stands for "Robot Operating System", which is a set of tools, libraries and open source software for the management of robotic systems. These tools seek to standardize, simplify and perform robust robotic tasks through simple operating protocols for nodes and topics integration. ROS was built with the objective of improving and facilitating the development of software for collaborative robots and there is a growing open source community, which allows the migration of a large number of legacy technologies to these new tools (ROS Community, 2021d).

Figure 1.8: ROS official logo. Taken from (ROS Community, 2021d).

In the context of a Robot Operating System, a node refers to an executable program used to communicate with other nodes through a "client" type resource library. There are also "messages", which allow subscribing or publishing native information in ROS to a "topic". Finally, topics are buses of information on which nodes exchange messages (ROS Community, 2019).

In any program that has a ROS infrastructure and architecture, there is a main element known as the "master", which aims to verify and validate that the operation of the nodes and their networks for communication are performing their corresponding tasks and prop-

erly integrating with each other (ROS Community, 2019).

1.3.2.3 Computer Vision Concepts

Computer Vision represents one of the most important fields in the robotic development of projects that must be able to detect dynamic conditions in a given environment. For the correct implementation and execution of this research, the fundamental and relevant concepts within this area of knowledge must be taken into account.

1.3.2.3.1 Overview of Computer Vision

Computer vision can be defined as the branch of study responsible for extracting information from one or multiple images through computational algorithms. There are many areas where it is applied, one of the most complicated being the creation of robotic trajectories, where this information is used to control robotic manipulators or autonomous vehicles (Rosenfeld, 1988).

One of the most important applications of computer vision is face detection and face recognition. This utility allows detecting a human face through relevant features found by image processing. It can be divided into two processes: identification of an unknown face and validation of a specific face that can be found encoded in a database (Singh and Prasad, 2018). In this project, it will be implemented a face-detection algorithm.

1.3.2.3.2 Face detection algorithms

Face detection is a specific field of computer vision algorithms that involve Machine Learning. The main purpose of this field is to estimate the face position and direction from a given image (YuichiAraki et al., 2001). Facial recognition algorithms started around 1960s and there have been multiple improvements of different approaches to solve the objective of detecting human faces.

One of the most recognized and significant article in the history of face detection is the famous "Rapid object detection using a boosted cascade of simple features", popularly known as "Viola–Jones object detection framework" (Viola and Jones, 2001). This algorithm enabled the fast processing of digital images with great accuracy via machine learning algorithms for object detection.

Viola Jones Algorithms, established three main relevant literature contributions (Viola and Jones, 2001). These are:

1. The introduction of a new image abstraction known as the "Integral Image". This allowed the quick processing of an analyzed image.

2. A learning algorithm that is based on "AdaBoost", that selects few relevant features from a big data-set and enables an efficient classifier.

3. A innovative method to combine separate classifiers that increased their complexity in a cascade approach. These allowed the background sections of an image to be discarded quickly in the process of detection.

Taking into account these considerations, most modern algorithms use features that were developed by Paul Viola and Michael Jones and evolved their specific classifiers in diverse aspects. One example of the importance of these features is the recognized open-source project called "OpenCV", that implements some variants of these algorithms in their previously trained "Face Detection" libraries (Open Source Computer Vision, 2021).

Figure 1.9: Face detection algorithm in action. Own development.

1.3.2.3.3 Stereoscopic Camera

These are frames acquisition devices, which have the ability to acquire images or videos, while being able to identify the spatial quality associated with a third dimension, i.e., they are able to capture the depth of the surfaces of the image. These cameras use strategies

based on the morphology of mammalian eyes to map a three-dimensional structure of the captured scene (Montalvo M, 2010).

Figure 1.10: Vuze stereo camera 3D device. Taken from (Vuze, 2021).

1.3.2.4 Control theory concepts

Within the framework of preliminary knowledge for the development of this research, it is very important to have a solid prior knowledge of classical control theory, in order to propose and validate robust control strategies that attempt to achieve the correct trajectory tracking of the dynamic system to be implemented.

1.3.2.4.1 Motion control systems

A motion control system aims to follow a given defined trajectory and seeks to reduce the error between the actual performance of the system and its planned trajectory. This control system is very important at the industrial level and are usually guided by 3 integrated control loops: position, velocity and torque. These motion control tasks have a high degree of complexity and usually involve variables such as system load, actuator saturation and robot workspace limitations (Jalani et al., 2008).

1.3.2.4.2 Decentralized control systems

A decentralized control system is one that has the ability to integrate multiple independent control systems and handle these global inputs to a master processing of each of the individual controllers, allowing the generation of independent control events managed by a master algorithm (Davison et al., 2020).

1.3.2.4.3 Impedance control systems

In the context of a motion control system, there is a relevant term known as “impedance”. This concept refers to the relationship between force and position and/or velocity variables. Taking into account this definition, a completely rigid system is one that has simple

impedance.

Based on these concepts, an impedance control seeks to carry out an intermediate regulation between the force variables of a system, applied to the desired positions in a specific trajectory. The underlying goal of this control strategy is to assign a prescribed dynamic behaviour of the motion system while its components are interacting with their environment (Canudas et al., 1996).

Impedance control seeks to model a real-life system into a "Generalize Dynamic Impedance", by a set of linear or non-linear second order differential equations representing a spring-damper-mass system (Canudas et al., 1996).

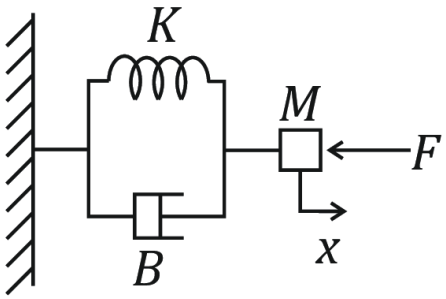


Figure 1.11: Abstraction model of motion system for implementing an Impedance Controller. Taken from (Pertuz et al., 2018).

1.3.2.4.4 Model predictive controller

Model Predictive Controllers, known as MPC, are control algorithms in which the dynamics of the system must be known through a previous identification of itself, by traditional methods of system identification. These algorithms have advantages such as simplicity, robustness, convenience in implementation, taking into account the system's constraints and a good handling of Multiple-Inputs/Multiple-Outputs (MIMO) systems.

Model Predictive Controllers seek to predict the changes in the dependent variables of the system model, which will be caused by the influence of the changes in the independent variables of the system. Similarly, they are widely used in systems with multiple constraints, such as those in which the actuators or sensors have saturation points (Mehta and Reddy, 2015).

MPC literature has been developed significantly over the last decades. One of the reasons that impulsed the community research in MPC algorithms is attributed to the fact that MPCs is a wide-open domain. It integrates and involves: optimal control, control processes, stochastic control, multi-variable control and future set-points of variables (Camacho and Bordons, 2007).

One great advantage of Model Predictive Controllers is the fact that they allow to predict a behaviour based on a prediction horizon that considers input constraints of the system

and the affected variables. This is done with linear and non-linear processes, allowing MPC to interact with a lot of important industrial processes.

Another great conditions that has been a golden-key for the development of MPC algorithms, are the fact that embedded system's processors are being improved with exponential characteristics over the last years. This has come with the circumstance that there are plenty of open-source projects that implement the mathematical approach of convex-optimization problems, an intermediate feature in MPC control strategies.

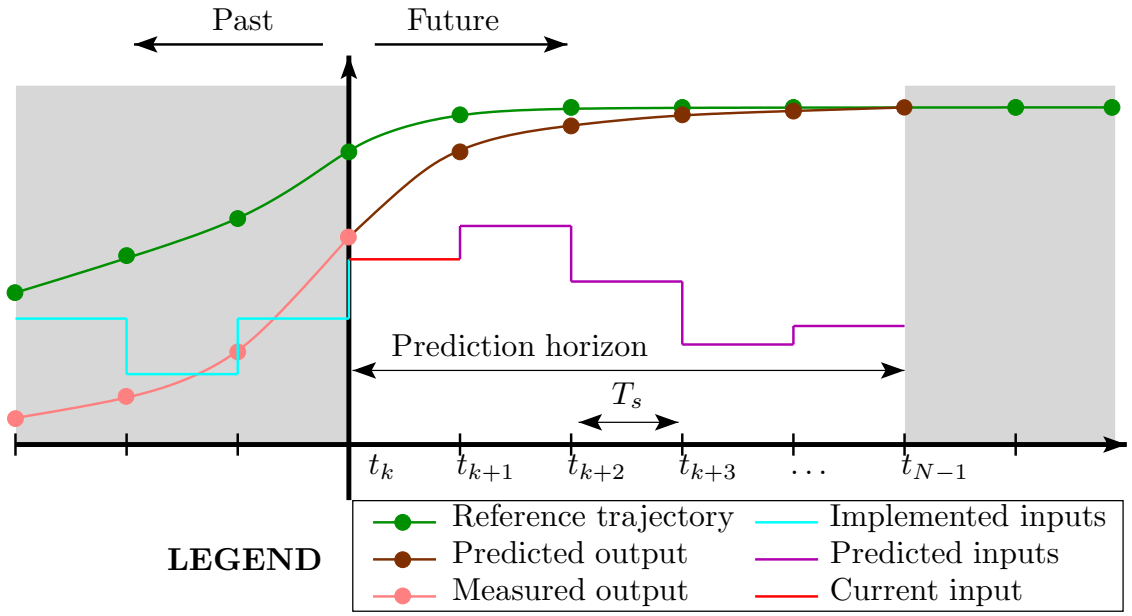


Figure 1.12: General Model Predictive Control plot for relevant variables. Taken from (Drgoňa and Ján, 2017).

Chapter 2

Methodology

There are several methodologies for scientific design to carry out a process of research divulgation. In this investigation it was decided to implement a methodology applied to engineering processes, where there are a series of iterative designs that seek to achieve a final detailed product (Dieter and Schmidt, 2012).

It is also important to point out that the development of this research will have an exploratory approach to the technological and practical solution of the problem to be solved, so the design stages focused on the market study and the financial components to generate a start-up plan at a commercial level will not be developed. Because of this statements, these two stages will be left as a proposal for future research.

2.1 User requirements identification

The selection process of the most relevant characteristics in terms of user requirements was carried out taking into account the existing literature on assisted robotics focused on feeding-systems and the general considerations that were determined to be relevant for the considerations of Baxter robot. In this choice for the user requirements, the following needs were identified:

#	Requirements	Relevance	Measurement
1	User must feel safe	5	Subjective
2	System must be able to be initialize through an user-interface	4	Objective
3	System must be able to detect the user's mouth	5	Binary
4	System must be able to generate a trajectory between two points in space	5	Objective
5	System must be able to measure the current end-effector's position	5	Objective
6	System must be able to move the end-effector following a trajectory	5	Objective
7	System must be able to move the end-effector close to the user's mouth	5	Objective
8	System must be able to stop in case of an anomaly detection	5	Subjective

Table 2.1: User requirements for active feeding system.

2.2 House of Quality

The House of Quality (HoQ), is a design tool that enables a simple and powerful approach for a "quality function deployment". The tool was originated at the Mitsubishi's Kobe shipyard site and it has helped the development of multiple industrial solutions, such as home appliances, infrastructures, electronics, clothing, integrated circuits and more (Harvard Business Review, 1988).

The methodology of a House of Quality, maps the corresponding requirements with important engineering conditions that are measurable. The result of these characteristics, generate two important parts: the main matrix and the roof correlation matrix. These matrices are important to understand the underlying relationships and their importance in a measurable way.

The result of the HoQ for an active feeding system for patients with motor disabilities in upper extremities, can be seen at fig[2.1].

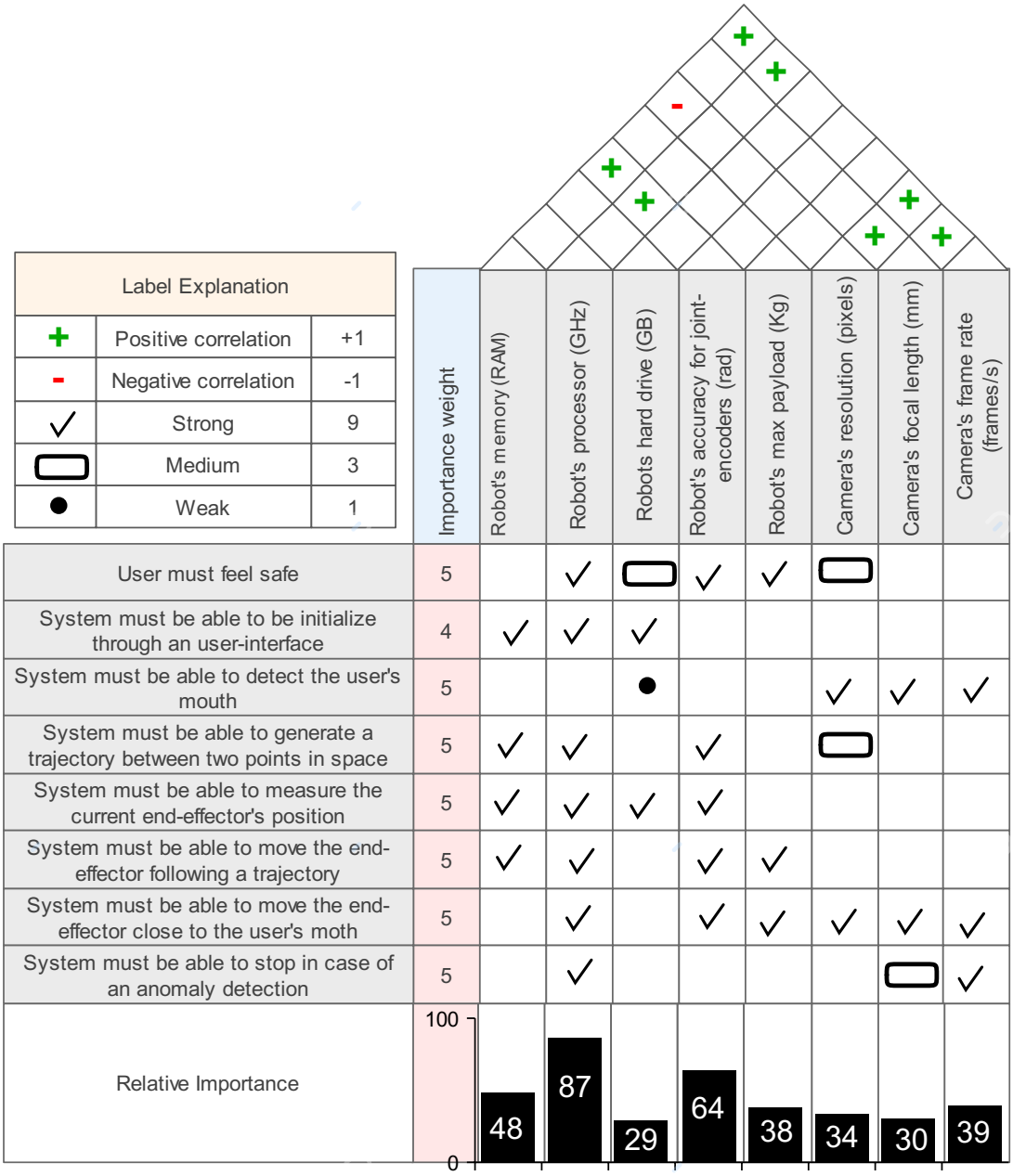


Figure 2.1: HoQ for an active feeding robotic system. Own development.

As a general conclusion for the House of Quality design analysis, it is important to notice that the three most important engineering conditions are:

1. Robot's processor (GHz)
2. Robot's accuracy for joint-encoders (rad)
3. Robot's memory (RAM)

As a general result, the camera conditions are similarly important and there is no significant difference on the relative relevance for those aspects. As well as that, the robot's payload is not a key aspect, because there is no need for high amounts of robotic load to manipulate.

2.3 System architecture

Having an understanding of the overall system's architecture is a key element in the development and analysis of an engineering solution. A block diagram of the complete system will be shown below to understand the signals and information that govern the proposed design.

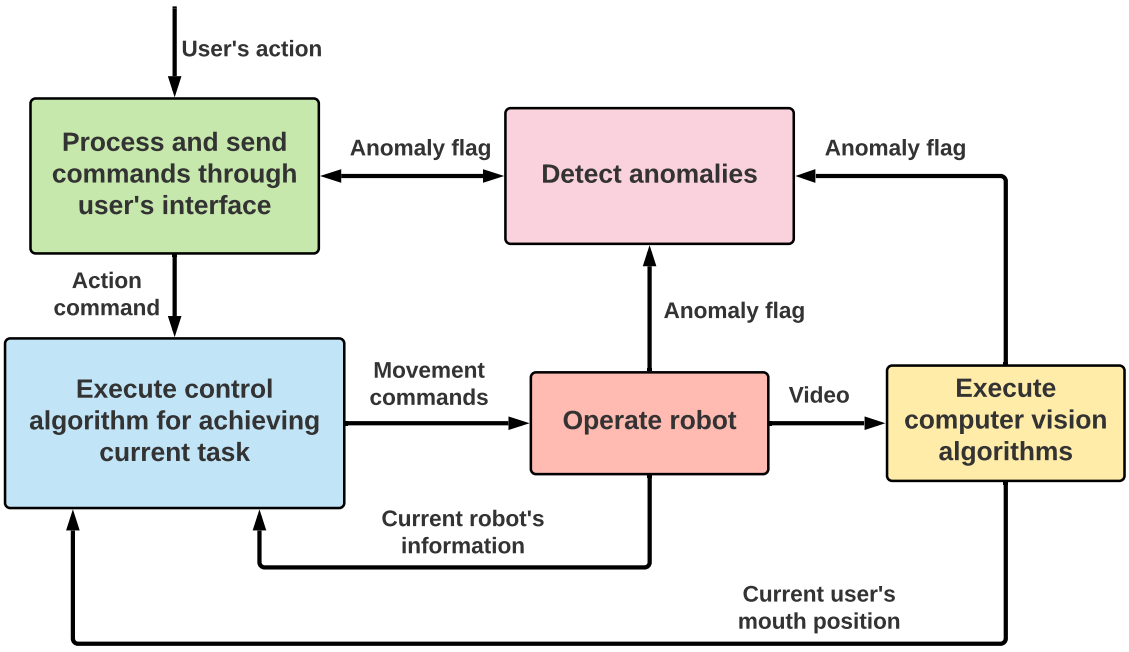


Figure 2.2: General system architecture. Own work.

As it can be seen in fig[2.2], the main input to the whole system is the action command generated by the user through the user interface. After these action commands are sent, the interface will transform them into signals that initialize the control processes, which enables the robot's strategic motion, depending on the current system conditions.

It is important to clarify that the current position of the user's mouth is identified from the robot's sensors, allowing the system to process the video from its cameras, find the position of the mouth and send it to the control algorithm.

Similarly, there is an anomaly detector, which is constantly monitoring the current conditions of the system (i.e. control algorithms, Baxter conditions and computer vision algorithms) in order to stop the process in case of any abnormal condition that could

harm the patient.

2.4 Morphological matrix

The morphological matrix is a powerful tool that makes it possible to generate solutions based on potential variations in a problem's characteristics (Cheshire, 2010).

For our design, the main problems or conditions are detected from the general block diagram seen in fig[2.2]. These are the main problems to evaluate the best solutions for our specific context. In the following sub-sections, all the alternatives for the morphological matrix, will be shown:

2.4.1 Solution alternatives

In this subsections, it will be discussed some possible valuable alternatives for successfully executing each one of the system's tasks. Their advantages and disadvantages will be taken into account for the final decision of the best possible solutions.

2.4.1.1 Process and send commands through user's interface

In order to ensure a pleasant user experience, there must be a subsystem that receives, processes and sends the necessary action signals to the rest of the system. This part of the system can be understood as a series of representative algorithms that allow interacting and regulating the tasks required by the user in each instant of time.

Solution Alternative	Advantages	Disadvantages
Tkinter  (Python Software Foundation [US], 2021)	Tkinter is a built-in Python library. It has a simple learning curve and is compatible with most Python programs. It is also open-source and there are external APIs that make the designs easier.	It is not compatible (out-of-the-box) with mobile devices. It does not give a simple solution for multi-interface programs.

Table 2.2: Tkinter solution alternative.

Solution Alternative	Advantages	Disadvantages
Bash Terminal  (GNU Operating System, nd)	It is the fastest possible solution for sending commands to interact with the system. It gives more possibilities for sending commands with different variables and scripts.	There are more possibilities of sending wrong commands, wrong parameters and errors. It is not friendly for the interaction. A great technical knowledge is needed for its usage.

Table 2.3: Bash solution alternative.

Solution Alternative	Advantages	Disadvantages
ElectronJs  (OpenJS Foundation, 2021)	It is a Graphical User Design tool that makes it simple to create interactive and pretty desktop applications. It has a high-performance interaction and a reusable framework.	It is normally a slow-start/slow-running program. It requires high amount of resources in the execution. It requires necessarily a web-server.

Table 2.4: ElectronJs solution alternative.

2.4.1.2 Execute control algorithm for achieving current task

In order to deliver correctly the food to the user, the system must be able to track and move its structure to the final mouth position in a dynamic and secure way.

The following alternatives, are different general control strategies that govern real solutions for different applications. There will be a comparison to determine which one adapts better for the feeding problem.

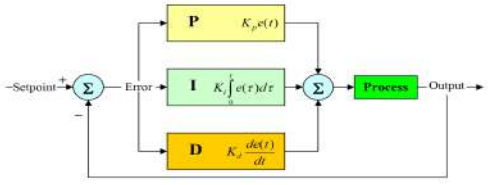
Solution Alternative	Advantages	Disadvantages
PID  (ACS Publications, 1996)	It is a well-known control strategy with multiple examples, documentation and real-life controllers ready-to-go. It is easy to implement.	They have poor control performances for an integrating process and a large time delay process. Hard to incorporate ramp-type set-point change or slow disturbance conditions.

Table 2.5: PID solution alternative.

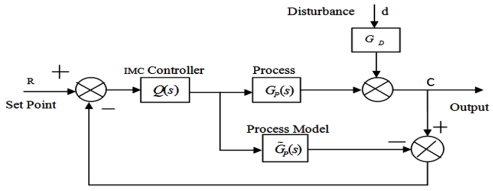
Solution Alternative	Advantages	Disadvantages
IMC  (Poe and Mokhatab, 2017)	It is a control strategy extremely simple to implement. It also has a condition that makes it possible to add extensions for MIMO systems as well as for some nonlinear systems straightforward.	It is common that IMC strategies have drawbacks such as a "recovery from process disturbances time" very slow, compared to the system dynamics.

Table 2.6: IMC solution alternative.

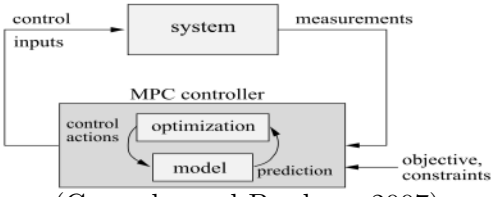
Solution Alternative	Advantages	Disadvantages
MPC  (Camacho and Bordons, 2007)	It is a flexible control strategy for linear and non-linear systems and/or multi-variable systems. It takes into account the constraints of the system and its variables.	It needs an accurate dynamic model of the system, and a computational algorithm for an optimization problem, that requires high computational cost.

Table 2.7: MPC solution alternative.

2.4.1.3 Operate robot

In order to have a robot structure that is able to give food to the patients, it is necessary to choose a robot that has the right compliance for this activity.

The next alternatives, are robots that are available in the "Laboratory of Industrial Automation" of the EIA University:

Solution Alternative	Advantages	Disadvantages
ABB IRB140 Robot  (Robots Done Right, 2021)	It has a great documentation and its structures and controllers are easily replaced. It has an excellent support assistance from ABB robotics.	It does not have a collaborative approach and is only used for industrial applications. It does not have have any embedded external sensors for its environment.

Table 2.8: ABB IRB140 robot solution alternative.

Solution Alternative	Advantages	Disadvantages
Baxter Robot  (Guizzo and Ackerman, 2012)	It is a collaborative robot designed for interaction with people and its environment. It has a ROS open-source system interconnected with multiple sensors.	It does not have external support by the manufacturer (Rethink Robotics). Its operating system runs in software from 2015.

Table 2.9: Baxter robot solution alternative.

2.4.1.4 Execute computer vision algorithms

One fundamental step in the process of giving food to the user, is detecting where his mouth is located in an accurate way and in a real-time approach.

To correctly implement this condition, the system must have a great camera that is able to capture the user's face and then, send this information to a computer vision algorithm that makes it possible to detect the mouth.

The considered alternatives for the vision sub-system cameras are:

Solution Alternative	Advantages	Disadvantages
Stereo Camera  (ZED Cameras, nd)	It is able to detect the depth of the image with its two cameras placed similarly to a human-vision-condition.	It requires high computational resources and constant calibration for the depth algorithms. Usually are more expensive than other cameras.

Table 2.10: Stereo camera solution alternative.

Solution Alternative	Advantages	Disadvantages
Baxter hand cameras  (IEEE Robots, 2012)	It is integrated with Baxter ROS signals and has the commands already implemented. The cameras are already calibrated and there are examples of its usage. Does not require any external structure or design.	It does not have depth considered in the information given by the cameras. Is not easily replaced if it is damaged.

Table 2.11: Baxter hand cameras solution alternative.

Solution Alternative	Advantages	Disadvantages
RGBD Camera  (Tang et al., 2016)	It is able to detect the depth of the image with its infra-red embedded projector with a low cost price. Multiple algorithms for SLAM applications. The community and online support is excellent.	It requires high computational resources. The depth is limited because sensors only allow measurement ranges of a limited distance and a limited field of view

Table 2.12: RGB-D cameras solution alternative.

2.4.2 Morphological matrix result

After diving into each alternative for the overall solution, the result morphological matrix for the autonomous robot feeding problem is the following:

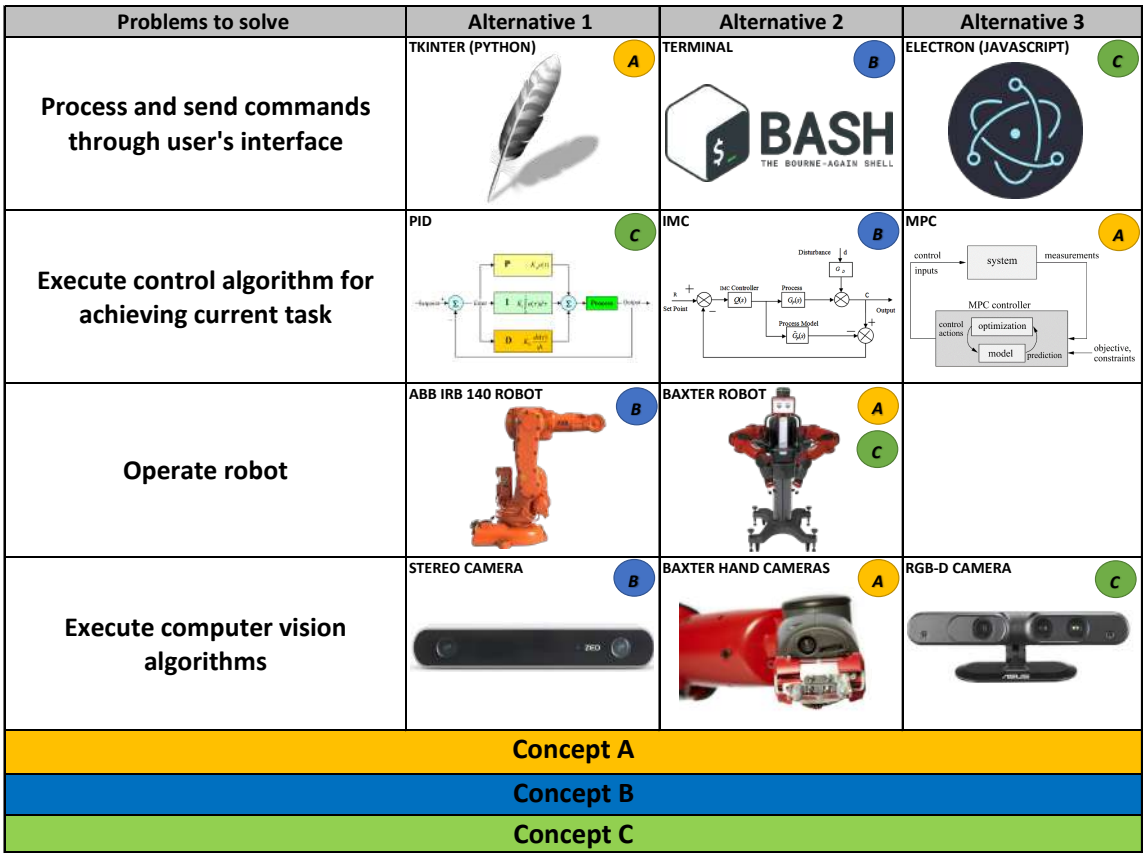


Figure 2.3: Morphological matrix for the possible concepts A, B and C. Own work.

2.5 PUGH Decision matrix

After the development of the morphological matrix, it is mandatory select the best possible solution based on a quantitative qualification approach. There are multiple ways to discern into the most recommended solution, and for this article, it will be chosen with the help of the "PUGH Decision Matrix" (PM) (Burge, 2009).

The PUGH matrix, compares the concepts in a objective-like ways, to determine a possible heuristic solution to the problem based on the engineering conditions defined in the HoQ of the methodology (the ones that impact directly to the user requirements). This decision matrix is considered a great approach, because it involves the most important part of the design, which is what the user really needs.

The result of the PUGH matrix for the assistive robotic feeding is:

Concepts			1	2	3
Name of concept			A	B	C
	1	Robot's memory	=	-	=
	2	Robot's processor	=	-	=
	3	Robot's hard drive	=	-	=
	4	Robot's accuracy for joint encoders	=	=	=
	5	Robot's max payload	=	+	=
	6	Cameras resolution	=	+	=
	7	Cameras focal length	=	=	-
	8	Cameras frame rate	=	=	-
Sum of the positive signs			0	2	0
Sum of the negative signs			0	3	2
Overall result			**0**	**-1**	**-2**

Table 2.13: PUGH decision matrix for active feeding system

As it can be seen in the table tab[2.13], the overall result A was the best concept, based on the given engineering conditions for the analysis.
After evaluating and analyzing the "PUGH Decision Matrix", the result was clearly aiming towards the A concept, which is explained in detail in the solution alternatives section.

After this part of the investigation, all of the procedures will be based on the following selected solution alternatives:

1. Process and send commands through user's interface: Tkinter.
2. Execute control algorithm for achieving current task: MPC.
3. Operate robot: Baxter robot.
4. Execute computer vision algorithms: Baxter hand cameras.

2.6 Software architecture

Although the general system architecture discussed on fig[2.2], enables the understanding of the signals and important information of the complete process workflow, it is extremely valuable to dive into how the software operates in order to achieve these functionalities. In the next diagram, the general software architecture will be presented divided into several layers of abstraction.

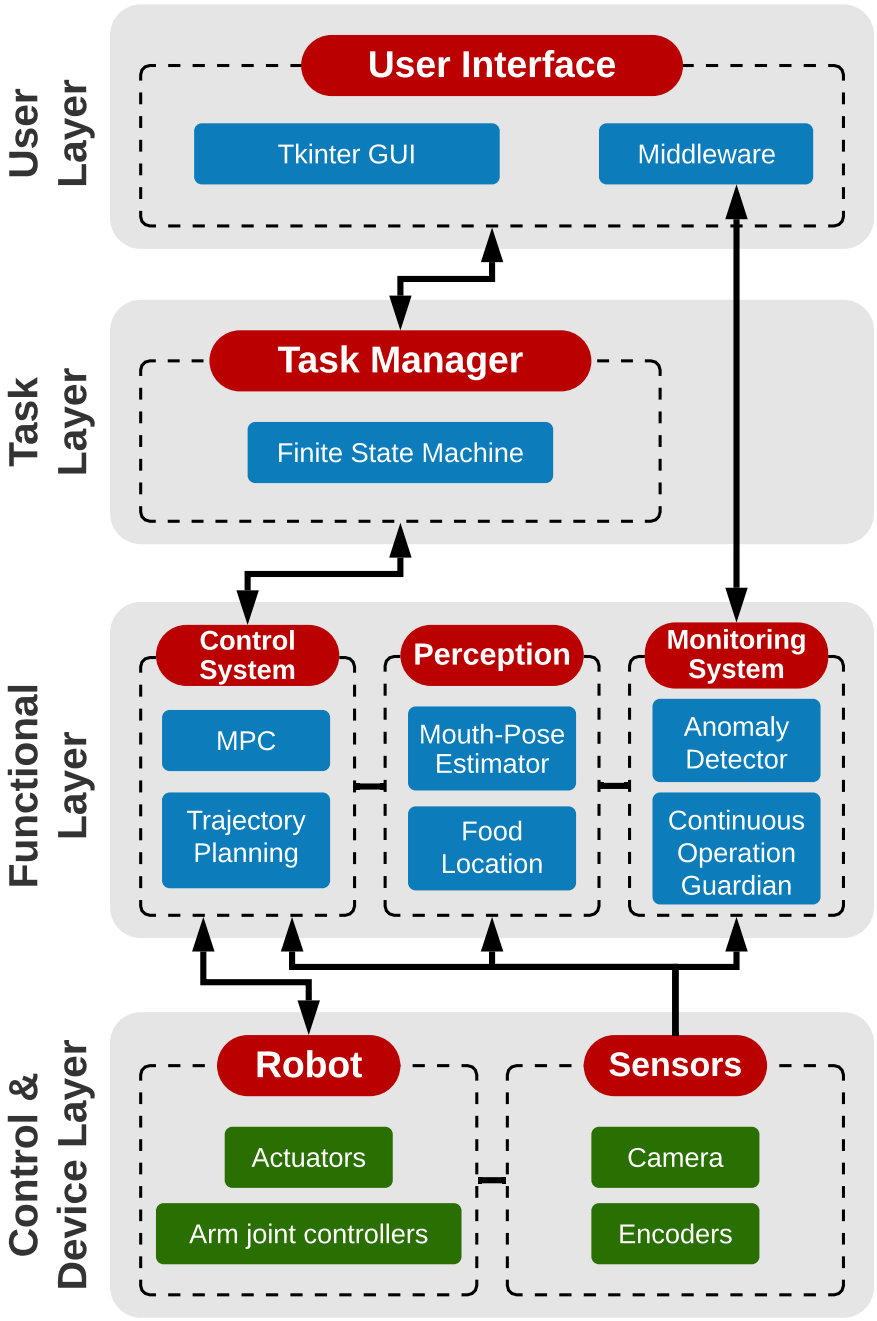


Figure 2.4: General software diagram for system's functionalities. Own work.

2.6.1 Control & Device Layer

The control and devices layer has low-level controllers and sensors that will allow to measure the current state of Baxter at all times. This layer has several embedded elements, where some of them can be subtly modified and others are black boxes that can not be edited at a low level operation.

In the control layer, it is important to keep in mind that the controllers mentioned are designed for Baxter and have a number of additional protections to prevent damaging them, ensuring that Baxter is a very safe robot and suitable for educational tests.

2.6.2 Functional Layer

The functional layer enables the most challenging and complex processing algorithms to be performed throughout the system. This layer is intended to perform the main tasks of the system, allowing control strategies, computer vision algorithms, anomaly detection and, continuous operation guardians for each one of the critical components.

The control algorithms will be divided into an MPC for the robot trajectories and an a trajectory planning in order to decide where the end effector should move, achieving a coordination of more than one control algorithm at the same time.

There will be a perception subsystem, where the correct detection of the position of the user's mouth and the position of the food will be guaranteed through computer vision algorithms. These will be connected in cascade as feeds for the inputs to the control algorithms.

The anomaly detector acts as a "supervisory agent" that is constantly receiving signals from the other subsystems and validating non-conventional behaviors in order to stop the system in case of the detection of an anomaly. This should be integrated with a "continuous supervisor" that restarts a given system if it detects any strange failure, such as a run-time exception or a memory failure, so the system keeps working normally at every moment.

2.6.3 Task Layer

The task layer is in charge of orchestrating each one of the steps of the finite state machine that will allow the dynamic operation of Baxter for each one of the tasks required in the food process (such as, for example, picking up the food and feeding the patient).

This finite state machine (FSM), understood as "a computation model that can be implemented with hardware or software and can be used to simulate sequential logic and some computer programs" (Brilliant.org, 2021), will decide which of the Baxter's tasks will be implemented in the main control algorithms, in order to generate the desired output of the system.

The task layer is a fundamental element in the system, because it understands and translates the user's inputs, into states of the FSM, enabling Baxter to move according to the user's needs.

This FSM must be designed with a correct architecture for guaranteeing that each state does not fall into an unstable switching condition between two or more tasks.

2.6.4 User Layer

This layer works as the highest layer of abstraction, where the user will interact with a simple Graphical User Interface (GUI) and will decide how Baxter should move based on the current state of his meal.

The GUI will be developed as a desktop application that connects strategically with the lower-level algorithms developed for the Finite State Machine. It will also have an integrated middle-ware, that acts as a connector between the subsystems that govern the feeding processes, so that it can take important actions if something wrong happens (like the sudden stop of an specific subsystem).

A middle-ware is defined as "a software that bridges gaps between other applications, tools, and databases in order to provide unified services to users" (Talend, nd).

2.7 Finite State Machine

The finite state machine has the purpose of managing the general behaviour of the system in each of the main tasks that Baxter (and the associated subsystems) will perform in the feeding tasks, according to the user's commands.

It is important to notice that there are multiple states that have transitions based on a specific condition that changes in the system (for example, a command given by the user). In the next diagram, it will be shown the general structure of the Finite State Machine for the feeding system with its specific transition conditions and states:

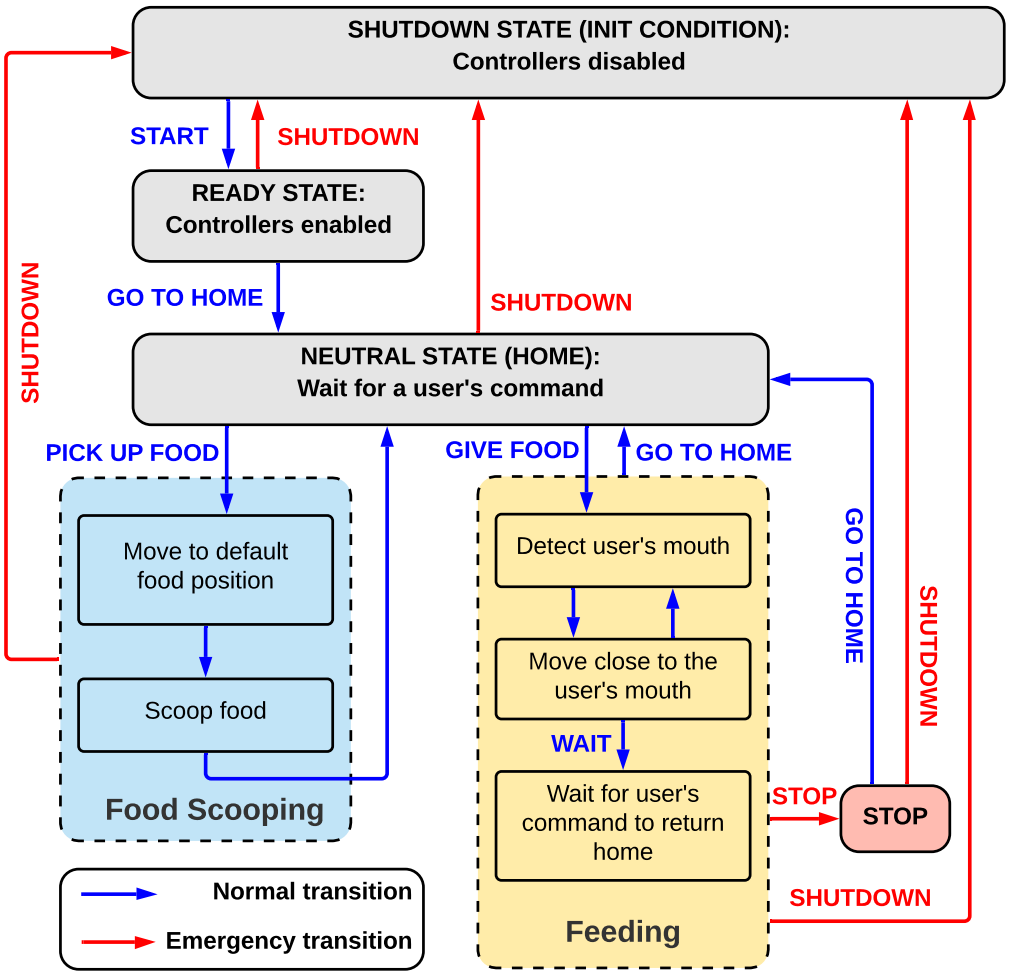


Figure 2.5: General software diagram for system's functionalities. Own work.

2.7.1 Shutdown state

The shutdown state is the default state in which the Baxter robot is completely static, because all of the joint-controllers are turned down. This state is the nearest one to a real hard drive shutdown of the robot, and any trial to transition to another state that is not the "Ready State", will end up in nothing.

This state, not only works as an initialization condition, but also determines a safety state in which the robot behaviour can be completely stopped, similarly to the emergency physical button that Baxter has.

It is important to notice that any other state of the system, is able to change directly and immediately to the "Shutdown State". This was designed thinking about the importance of the safety of the patients.

2.7.2 Ready State

As its name implies, the "Ready State" is the one that enables the whole system and prepares it to start working correctly. The transition to this state is only possible with the "Start Button" transition.

When the user starts the system, it is a condition that should enable the robot's controllers and it runs the important initialization programs for all the sub-systems algorithms and run-time routines.

2.7.3 Neutral State

This state is considered to be one of the central and most used states for the overall functionalities. This state should prepare the robot to start the movements of any other state conditions, such as "Pick Up Food", "Give Food" and "Shutdown".

The neutral state is popularly known as the "Home Condition", because is a well known workspace disposition for the robot environment and conditions of its degrees of freedom for the limbs.

It is also important to notice that every other movement state, usually ends up in the "Neutral State", which means that this state is a logical connector between each one of the tasks of the system.

2.7.4 Food Scooping States

The "Food Scooping States" are composed of the "Move to default food position state" and the "Scoop food state". These states work together as a complete task of food scooping, and will always be executed in series, starting by the first one.

It is important to realize, that there is no default transition condition to go to the "Neutral State", which implies that at the end of the execution of the tasks of this state, the system will automatically return to home by its own algorithm.

Both states of the "Food Scooping States", will be able to stop their procedure, when the user executes a shutdown transition. This will cancel the tasks and return the system to the "Shutdown State".

2.7.5 Feeding States

The "Feeding States" are designed with three main internal states. These are:

1. Detect user's mouth state.
2. Move close to the user's mouth state.

3. Wait for user's command to return home state.

After analysing the structure and transitions of these internal states, it is clear that there is an internal control loop as a result of the first two states. This control loop is a logical consequence of the need of a feedback-wise system that interacts dynamically with the environment and the patient.

At the end of the "Feeding States", the user can decide to stop the system or to return to the neutral state. This decision will also direction the finite state machine, to prepare the system for a future command (such as "Food Scoping State", "Feeding States", or, a final "Shutdown State").

2.7.6 Stop State

As mentioned before, the "Stop State" is a condition that guarantees that the robot does not move or execute any motion command. This pause is a software condition that does not mean that the controllers are off, instead of that, this state works based on a series of conditions and regulations that stop the robot in a safe way.

2.8 Important Project Remarks

This research project will be designed implementing an active robotic solution with Baxter cobot. There are some important concepts to understand for the reach of this investigation.

The first key condition, is that the project will be named "Baxter Bon Appetit", referring to the french common expression that has the objective of positively implying someone to enjoy their meal. The purpose of this name, is to create an etymological meaning that transcends in an outstanding robotic solution, that generates a safety feeling of appreciating a good meal.

Figure 2.6: Baxter Bon Appetit project logo. Own development.

Another valuable appreciation, is that the initial experiments of this research project, will be tested on the authors (Santiago Garcia Arango and Elkin Javier Guerra Galeano), but for future improvements, there will be a greater development of quality assurance tests in multiple patients. This will be discussed in the "Ethical Considerations" on the final chapters.

Chapter 3

Baxter Robot

The robot that will be implemented in this research is the Baxter robot from Rethink Robotics (founded by Rodney Brooks)(Rethink Robotics, 2015d). Baxter cobot is a collaborative robot with two arms and an interactive head. This robot has a series of tools and internal safety systems that makes it possible to develop robotic solutions at the industrial and academic levels.

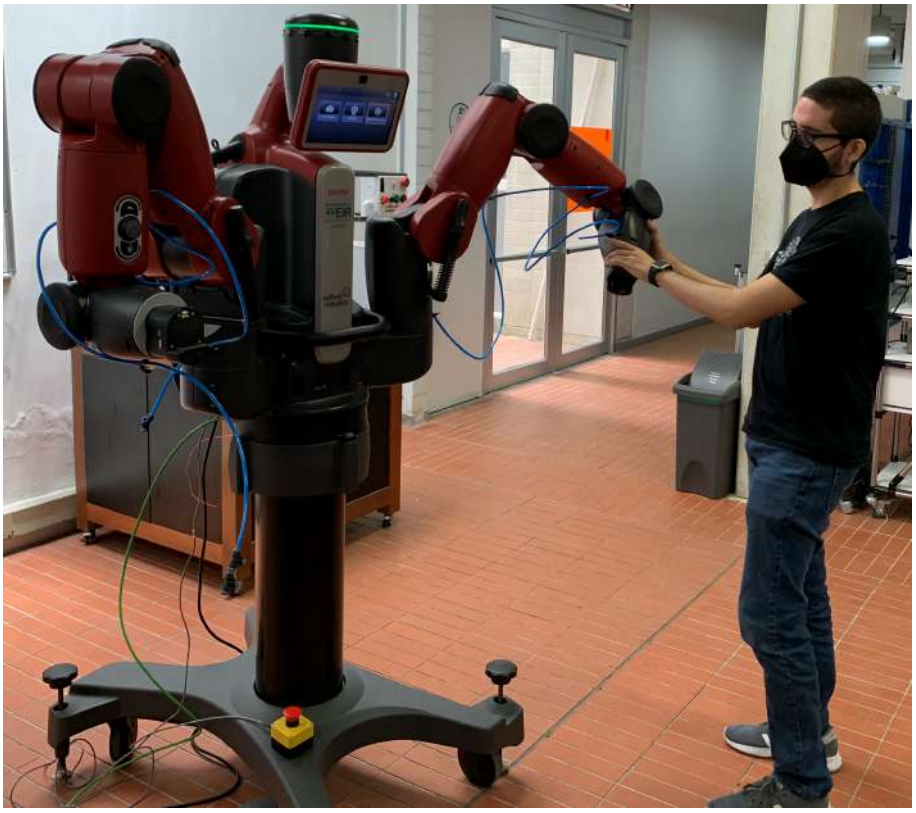


Figure 3.1: General view of Baxter Robot. Taken at the EIA University Laboratory

3.1 Baxter Robot Discovery and Characterization

To work with Baxter robot, it is important to refer to the official Baxter Rethink Robotics Wiki. This website has multiple valuable features and information about the setup, functionalities, SDKs, considerations and development of solutions with Baxter (Rethink Robotics, 2015d).

To dive into the official documentation of Baxter robot, please refer to:

- Baxter Rethink Robotics Wiki

In this section, it will be discussed some important aspects of Baxter's initial discovery and characterization for the research work.

3.1.1 Baxter Initial Setup

To correctly configure Baxter for the first time, it is important to refer to the "Initial Setup" tutorials. These tutorials can be found at:

- Baxter Initial Setup

For the development of this research project, Baxter was already assembled with its pedestal and physical connections already plugged in.

The first steps to configure and check Baxter right performance, was to connect a physical keyboard to Baxter's back USB connector. This connections was done in the back of Baxter with the USB entry as can be seen in fig[3.2]:

Figure 3.2: Back view of Baxter robot. Taken at the EIA University Laboratory

After that connection, we ran a pre-boot configuration to validate low-level hardware checks to confirm that each one of Baxter components was working correctly. The tests passed successfully in the validation.

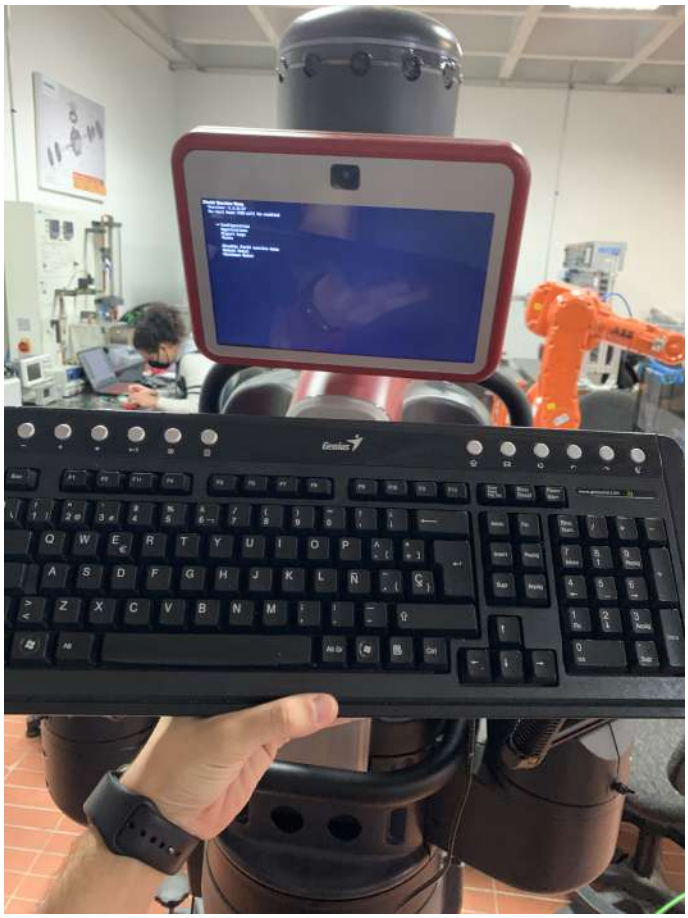


Figure 3.3: Implementation of low-level hardware checks in the FSM menu. Taken at the EIA University Laboratory

As well as the hardware checks, the official documentation implies that it is important to run Baxter joint-controller calibrations every month (Rethink Robotics, 2015i). This process is relevant in order to guarantee that the robot's arms responding reliable on its movements and on its specific sensor signals.

This calibration was implemented with an internal Baxter ROS node that ran a programmatic routine for about 30 minutes. The step-by-step documentations are found in this wiki:

- Baxter Arm Calibration

After this initial configurations, it was concluded that Baxter general low-level conditions and controller calibrations were working perfectly.

3.1.2 Baxter Components Specifications

It is mandatory to have a clear and descent understanding of Baxter components. This knowledge can help to determine the way in which the robotic solutions are developed and optimized. A robotic system is usually assembled with default general parts, such as arm(s), end tools, sensor and security components. In this section, Baxter components will be examined in a general point of view.

The main Baxter components are:

3.1.2.0.1 Robot

As clear as it could be seen, Baxter robot has a central computer unit that is located inside of its torso. This CPU is able to run multi thread programs to perform complex robotic operations that involve multiple applications.

It is also important to specify that the most important interactive safety measure is known as the "Emergency Button". This E-Button is able to completely shutdown Baxter robot and kill all the underlying processes that are being executed at any time. This acts similarly as an "Enable/Disable" condition.

3.1.2.0.2 Grippers

Grippers, also known as end effectors, are the final part of a robotic arm. The grippers act as a crucial part of a robot because they enable the manipulation of external elements in a workspace.

Baxter robot supports two default type of pre-built grippers. These are: electric grippers and suction grippers. Many applications require one of these end effectors as a primary tool for the correct development of an automation task.

Baxter robot, also allows external "own-made" grippers to be attached to its hands, making it possible to design multiple solutions that are not initially considered.

3.1.2.0.3 Head

Baxter head is a subtle but fundamental component of the robot, because it can be used as an interactive tool to display friendly images and show possible tasks to the user. The head is also a way to demonstrate that Baxter is designed for collaborative environments that are easily adapted to multiple scenarios with people around him.

The head is also capable of panning a total of 180° in horizontal directions and nodding vertically as well. This condition is a way to implement an extra layer of interactions with the work-space's users.

3.1.2.0.4 Screen

The screen is a 1024 x 600 SVGA LCD display that is completely integrated with Baxter internal components. This screen is intended to be an extension of the robot's functionalities.

3.1.2.0.5 Arms

One of the particularities of Baxter robot, is that is has two independent arms (which is different than most industrial robots that only have one). The fact that Baxter was developed in this way, is a design strategy to show that the robot is a "friendly system" that looks similar to human being, increasing the trust that users have for it.

Each arm is composed of seven degrees of freedom that are moved, controlled and measured with internal electro-mechanical components. These arms are also conceived to have multiple "Series Elastic Actuators (SEAs)", that enhance different available control modes, to ensure the overall safety of the system (Rethink Robotics, 2015a).

Baxter hands also implement extra signals sensors, such as cameras, infrared sensors, accelerometers and navigators. These sensors are totally integrated with the internal electronic and software architecture of the system.

3.1.2.0.6 Cameras

A key component of the whole system is the fact that it has multiple high-resolution cameras. These cameras are named the "left_hand_camera", "right_hand_camera" and "head_camera". All cameras are internally connected with Baxter software architecture and enable extra processes such as computer vision algorithms based on the analyzed images.

One important limitation to keep in mind is that the hardware conditions do not allow Baxter to operate more than two cameras at the same time (this is because of an internal USB hardware requirement).

Baxter also allows to individually configure camera conditions, i.e, window width, window height and max frame rate.

3.1.2.0.7 Input and Output

Although the other components may have internal I/O systems, Rethink Robotics considered the "Input/Output" conditions as a key feature of the final solution. These I/O are the direct conditions to interact with Baxter robot, and are designed in a robust, secure and easy-to-program way.

The three main I/O categories are “Analog I/Os”, “ Digital I/Os” and “Navigator Interface I/Os” (Rethink Robotics, 2015h). All three have different electronic signal behaviours and are composed of: sensors, buttons, switches, lights, then navigators, then digitals and analog IOs.

3.1.3 Baxter Hardware Specifications

Baxter robot is documented officially in the Rethink Robotics Wiki with multiple measurements and information about its hardware components.

As it is said in the robot's data-sheet, Baxter was designed for continuous operation and can run 24/7 for expended periods of time without the risk of damaging the hardware of the robot(Rethink Robotics, 2015g).

In this section, we will dive into important characteristics and features that are going to be essential in the future chapters. All of these measurements are taken from the official documentation and were guarantee to have minor errors.

3.1.3.1 Baxter Arm Specifications

Both Baxter arms are exactly the same in terms of distances, angles and internal conditions, with the only exception of being symmetrical.

In the following image (fig[3.4]), there is a detailed explanation of Baxter hand with the notation of its angle names (that are used for the internal processing and execution of Baxter commands):

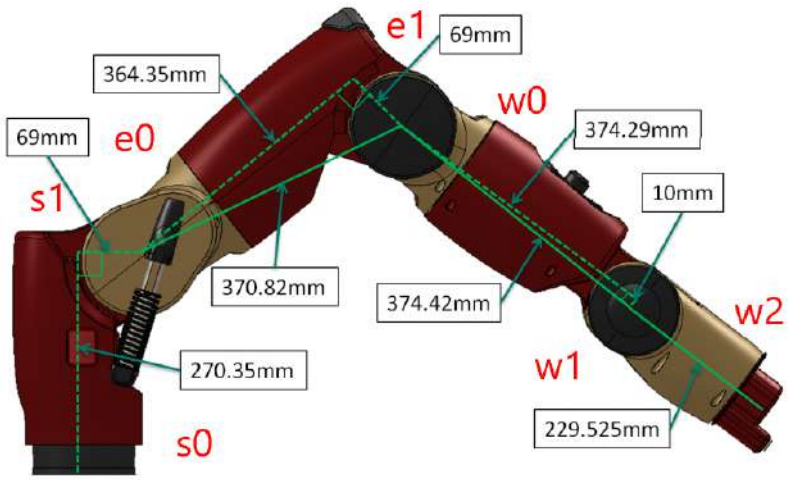


Figure 3.4: Hand measurements and angle names for Baxter robot. Adapted from (Rethink Robotics, 2015g).

Joint	(Degrees) Min limit	Max limit	Range	(Radians) Min limit	Max limit	Range
S0	-97.494	+97.494	194.998	-1.7016	+1.7016	3.4033
S1	-123	+60	183	-2.147	+1.047	3.194
E0	-174.987	+174.987	349.979	-3.0541	+3.0541	6.1083
E1	-2.864	+150	153	-0.05	+2.618	2.67
W0	-175.25	+175.25	350.5	-3.059	+3.059	6.117
W1	-90	+120	210	-1.5707	+2.094	3.6647
W2	-175.25	+175.25	350.5	-3.059	+3.059	6.117

Table 3.1: Joint range values for Baxter Robot. Adapted from (Rethink Robotics, 2015g).

Another interesting condition that is required for implementing control solutions is the torque and speed of the arms. In the following chart, those conditions are going to be shown.

Joint	Maximum Speed (rad/sec)	Peak Torque (Nm)
S0	2.0	50
S1	2.0	50
E0	2.0	50
E1	2.0	50
W0	4.0	15
W1	4.0	15
W2	4.0	15

Table 3.2: Maximum joint speed values and torques for Baxter robot. Adapted from (Rethink Robotics, 2015g).

These joint speed values and torques are considered to be fast, considering the normal operating conditions of collaborative robots.

To correctly acquire Baxter current joint angles, Baxter has high quality sensor at each on of its degrees of freedom. These sensors have the following specifications:

- The resolution for the joint sensors is 14 bits (over 360 degrees); so $360/(2\hat{1}4) = 0.021972656$ degrees per tick resolution.
- The joints have a waveform non-linearity, based on an accuracy on the order of about +/-0.10 degrees, worst case +/-0.25 degrees accuracy when reaching the joint limits.

3.1.3.2 Baxter Camera Specifications

Camera signals play a fundamental role in the development of autonomous solutions. For Baxter robot, these conditions are similar for each on of the cameras and are the typical values are the following:

Description	Spec	Unit
Max Resolution	1280 x 800	pixels
Effective Resolution	640 x 400	pixels
Frame Rate	30	frames per second
Focal Length	1.2	mm

Table 3.3: Camera specifications for Baxter robot. Adapted from (Rethink Robotics, 2015g).

It is important to notice that these conditions will be analyzed better in the computer vision algorithms developed in the final sections of this research.

3.1.3.3 Baxter CPU Specifications

CPU specifications are a direct limitation for any robotic solution, because the processor is in charge of executing multiple complex threads for the correct real-time applications.

Description	Spec
Processor	3rd Gen Intel Core i7-3770 Processor (8MB, 3.4GHz) w/HD4000 Graphics
Memory	4GB, NON-ECC, 1600MHZ DDR3
Hard Drive	128GB Solid State Drive

Table 3.4: CPU specifications for Baxter robot. Adapted from (Rethink Robotics, 2015g).

3.1.3.4 Baxter Additional Specifications

There are some final additional important specifications that Rethink Robotics has considered as relevant to consider in order to develop disruptive robotic solutions. These considerations are specified in table[3.5]:

Description	Spec
Battery Operation	DC-to-120V AC Inverter
Max Consumption	6A at 120V AC, 720W max per unit
Total weight (with pedestal)	298 lbs / 135.2 kg
Screen Resolution	1024 x 600 pixels
Positional Accuracy	+/- 5 mm
Max Payload (including end-effector)	5 lb / 2.2 kg
Infrared Sensor Range	1.5 – 15 in / 4 – 40 cm

Table 3.5: Additional specifications for Baxter robot. Adapted from (Rethink Robotics, 2015g).

3.1.4 Baxter Software Specifications

Robots are devices that must integrate multiple sensors, actuators, protocols, validations, security features, programmable processes and more. This requirements are only possible

with the right software and hardware integration for the desired robot features. Based on this statement, there must be an overall layer of abstraction for controlling the specified aspects and executing the robot´s functionalities in a reliable way. For Baxter robot, this software and hardware conditions are run on based on a "Robot Operating System" (ROS) (ROS Community, 2021d).
Baxter runs on ROS Indigo and implements an Ubuntu 14.04 Linux distribution. This software requirements can be eventually upgraded to newer versions with minor work, but is not a straight-forward approach, because some important functionalities could get unstable or not compatible. For this reason, for this research these software requirements are going to be used.

(a) ROS Indigo Igloo implemented in Baxter robot. Taken from (ROS organization, 2014)

(b) Ubuntu 14.04 Linux distribution implemented in Baxter robot. Taken from (Canonical and Ubuntu Community, 2014)

Figure 3.5: Required software dependencies for running Baxter robot.

3.2 Baxter Robot Mathematical Approach

The mathematical approach of any dynamic system, has its foundations on reference frames that enable the absolute and relative Cartesian systems. For robotic systems, it is common to define a main workspace reference and start developing models to transform these coordinates into multiple other references throughout the system.

Baxter is a completely symmetrical two armed hyper-redundant 7DOF humanoid robot. This condition makes it useful to define a central reference as the workspace center, so that the transforms are similar in each one of the arms. It's also relevant to notice that both limbs are exactly the same in measurements and the only key difference the sign of an angle.Each arm has a total of seven degrees of freedom (DOF) of rotational joint type (**R**).

In order to be determine the position and orientation of Baxter end effector in a mathematical way, it is important to understand that Baxter has multiple joint-encoder sensors that are only able to measure the angle of an articulation based on the angle that it makes with respect of the previous one.

These encoders are already calibrated and are the main way to map the end effector to

the central reference frame. To do this mapping, it is mandatory to have a mathematical representation with Transformation Matrices for each one of the reference frames that are in Baxter's joints.

In this section, there are some important values to keep in mind for the mathematical results of the transformation matrices. These values correspond to the distances shown in figures fig[3.6], fig[3.7] and fig[3.9]:

Length	Value (mm)
L0	270.35
L1	69.00
L2	364.35
L3	69.00
L4	374.29
L5	10.00
L6	368.30
L	278.00
h	64.00
H	1104.00

Table 3.6: Lengths of Baxter robot. Adapted from (Williams, 2017).

There is an extra consideration for these values, because the final distance "L6", is considered as the distance from the wrist pitch center, to the final part of the end effector. Keep in mind that the current value specifies the distance without any tool attached to it. This distance changes based on the distance of the tool that is implemented on Baxter.

3.2.1 Baxter Fixed Transformation Matrices

There are some important transformation matrices that enable the mapping of Baxter arms with respect of the main reference frame (this is from {W0} to {BR} or from {W0} to {BL}, as it can be seen in figures fig[3.6] and fig[3.7]. These transforms are considered "fixed", because they are constant in time. That means that we do not need to implement any addition algorithm to calculate the current values (those calculations are only relevant in the angles for Baxter arms).

In the following figures, it is possible to determine the relevant reference frames to do the initial transformation matrices:

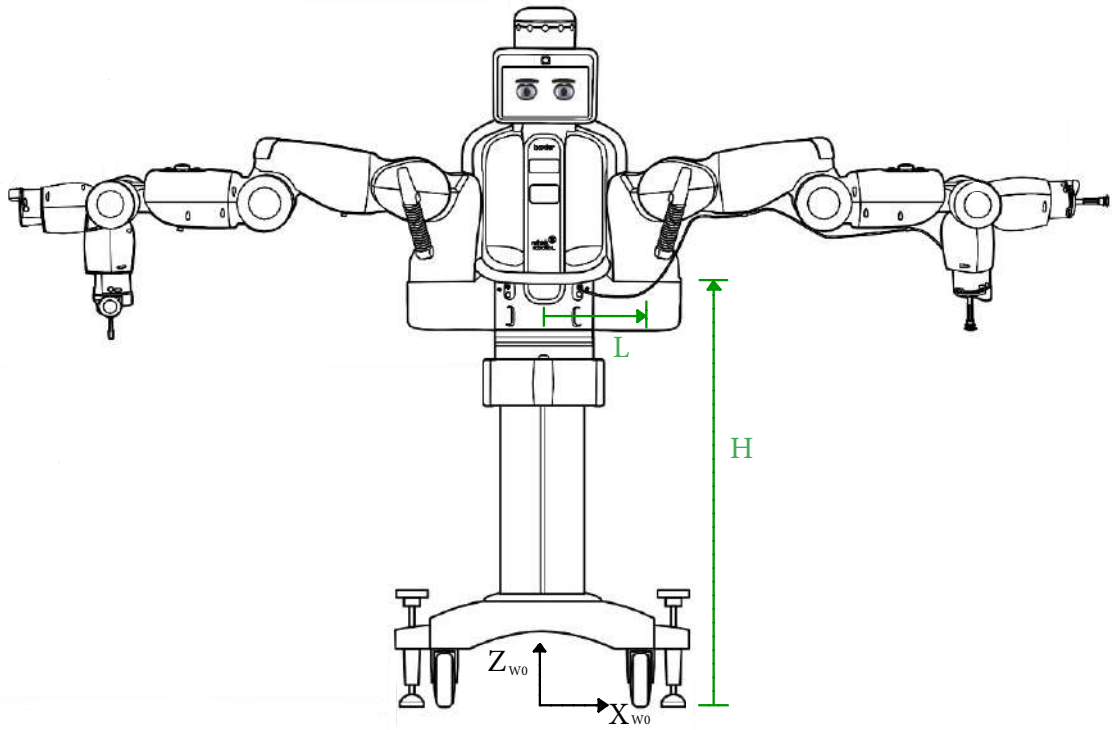


Figure 3.6: General Baxter schematic with reference frames. Adapted from (Williams, 2017).

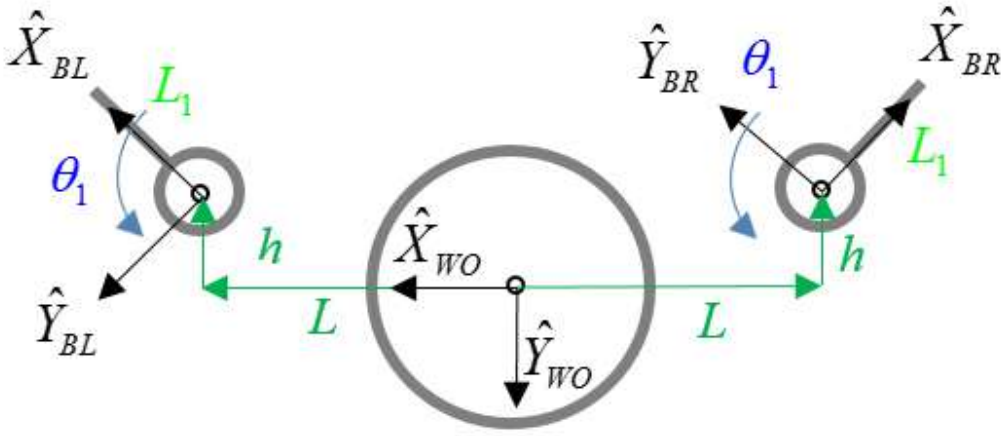


Figure 3.7: Torso to arm schematic with reference frames. Adapted from (Williams, 2017).

The transformation matrices from the central reference, to the beginning of Baxter arms, are:

$$\begin{bmatrix} {}^{W_0}_{B_L}\mathrm{T} \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & L \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & -h \\ 0 & 0 & 1 & H \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} {}^{W_0}_{B_R}\mathrm{T} \end{bmatrix} = \begin{bmatrix} -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & -L \\ -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 & -h \\ 0 & 0 & 1 & H \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

3.1: Transformation matrices from W0 to BL and from W0 to BR.

These matrices have exactly the same absolute values in terms of the distances, but the only key difference is the sign of the translations and rotations. This difference is due to the fact that the robot is symmetrical and thus, the values are the opposite respect to X-axis in translation and a 90 degree difference in Z-axis the rotation (with respect to each other).

After these fixed transformation matrices are taken into consideration, there are two extra constant transformations that must be done before applying the DH parameters algorithm. These transformations are based on the frames specified on fig[3.9]:

- Transformation from the reference frame of the arm of Baxter ({BR} or {BL}) to the beginning of the variable part of Baxter's arm ({0}).
- Transformation from the reference frame of the end of Baxter arm ({7}) to the end effector ({GL} or {GR}).

$$\begin{bmatrix} {}^{B_L}_{0}\mathrm{T} \end{bmatrix} = \begin{bmatrix} {}^{B_R}_{0}\mathrm{T} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & L_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} {}^{7}_{G_L}\mathrm{T} \end{bmatrix} = \begin{bmatrix} {}^{7}_{G_R}\mathrm{T} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & L_6 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

3.2: Transformation matrices from BR/BL to 0 and from 7 to GL/GR.

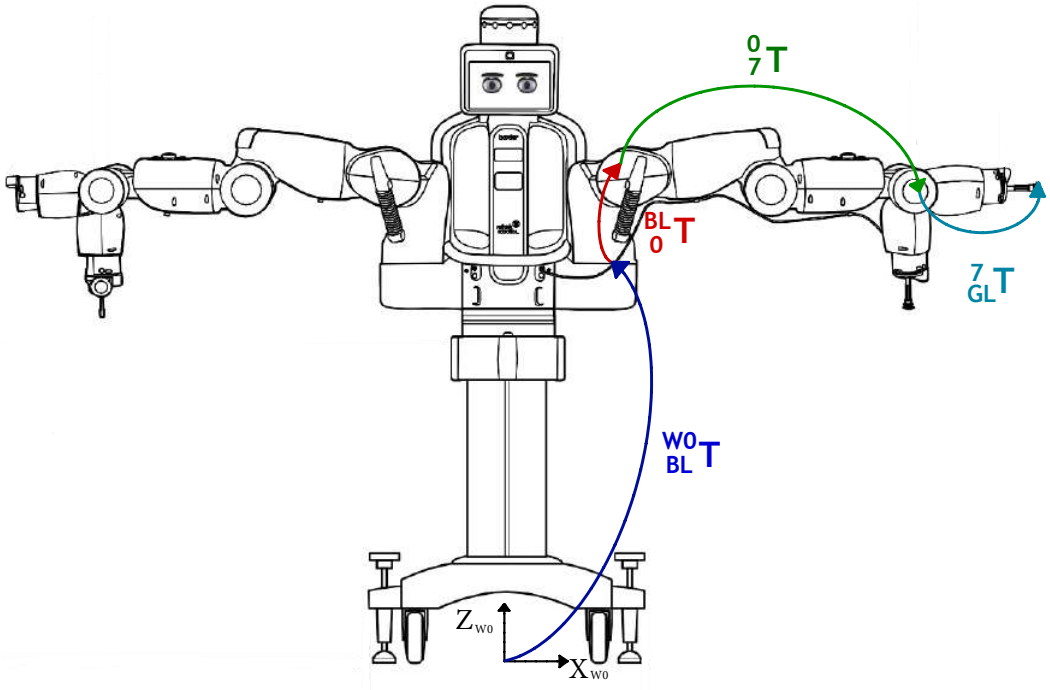


Figure 3.8: Visual complete transformation matrices for Baxter left hand. Own development.

After these fixed transformation matrices were explained by inspection, the next section is going to explain how to use the "DH notation" to specify how to calculate the missing transformations of the reference frames from {0} to {7}.

3.2.2 Baxter Robot Denavit-Hartenberg Parameters

In robotics, Denavit-Hartenberg Parameters (DH params), are used to specify the relationships and notations of multiple reference frames that are present in a chain-wise robotic system. This algorithm makes it possible to standardize the coordinate frames for spatial linkages easily (J. Denavit and R. Harterberg, 1955).

For Baxter DH Parameters, it is going to be used the modified Denavit-Hartenberg algorithm, also known as the "Craig Style DH Params". This modification only differs in a rotation of frames that happens directly at a joint, instead of a distal joint) (Craig, 2004).

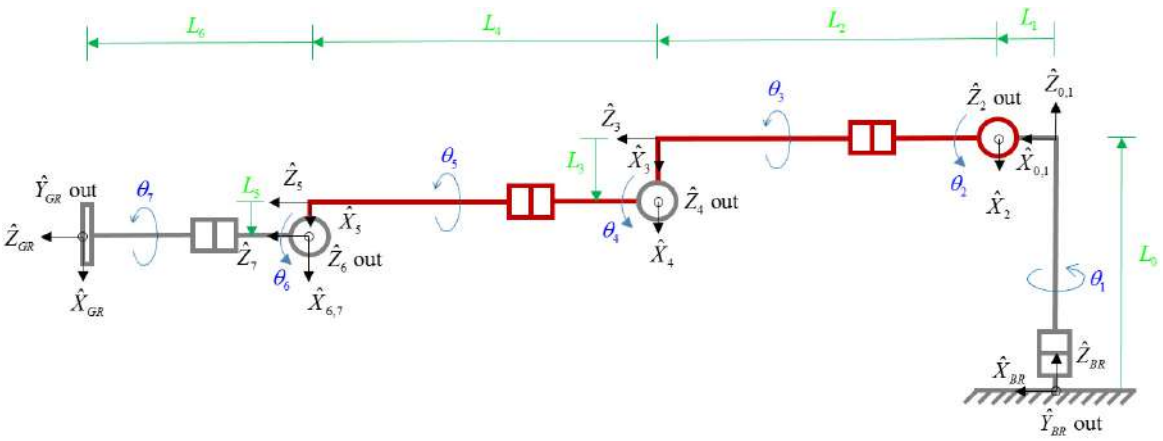


Figure 3.9: Right arm schematic with reference frames. Adapted from (Williams, 2017).

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	-90°	L_1	0	$\theta_2 + 90°$
3	90°	0	L_2	θ_3
4	-90°	L_3	0	θ_4
5	90°	0	L_4	θ_5
6	-90°	L_5	0	θ_6
7	90°	0	0	θ_7

Table 3.7: Denavit Hartenberg Parameters for the arms of Baxter robot. Adapted from (J. Denavit and R. Harterberg, 1955).

One great advantage of Baxter robot is that the Denavit Hartenberg parameters are identical for both of its arms, making it simple to describe the kinematics and dynamics of the entire system, due to the symmetrical configuration. The main difference will only be considered for the fixed transformation matrices shown in the last section in equations eq[3.1] and eq[3.2].

After inspecting and analyzing the Baxter DH parameters table, it is now possible to determine a specific transformation matrix of any of its joints, based on the measurement of the encoder sensors of all of its degrees of freedom. This affirmation will be extremely important, for future calculations that require multiple transformation matrices at a given time.

3.3 Baxter Robot Algorithms

Two important mathematical algorithms for robot kinematics are the "Forward Pose Kinematics" (FPK) and "Inverse Pose Kinematics" (IPK). These approaches have different purposes and create the basis of any robotic solution. In FPK, the length of each one of

the links and the angles of the DOF are known and the goal is to calculate the position and orientation of the end effector (or any intermediate part) in the physical space of the robot. In the IPK, the length of each one of the links is known and the current position of the end effector is given, and the goal is to calculate the value of the angles for each one of the DOF (Yavuz, 2009).

The FPK and IPK algorithms will be explained in the following sub-sections:

3.3.1 Baxter Forward Pose Kinematics

The objective of the FPK for Baxter, is to determine the position and orientation of the end-effector frame. To solve this problem, we can proceed to implement the DH parameters in the following way:

1. Substitute each line of Denavit-Hartenberg Parameters shown in table tab[3.7] with the current joint angle values.
2. Then, apply the necessary rotations and translations in the DH axes, to determine the intermediate transformation matrices for each one of the links.
3. Calculate the overall end-effector transformation matrix from the result of matrix multiplication of each one of the intermediate matrices.

The previous steps can be expressed in a matrix-wise way as it follows (these abbreviations are used: $c\theta_i$=cosθ_i, $s\theta_i$=sinθ_i, $c\alpha_{i-1}$=cosα_{i-1}, $s\alpha_{i-1}$=sinα_{i-1}):

$$\left[{}^{i-1}_{i}\mathrm{T}\right] = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -d_i s\alpha_{i-1} \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & d_i c\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & & & \\ & {}^{i-1}_{i}\mathrm{R} & & {}^{i-1}\mathrm{P}_i \\ & & & \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

3.3: Transformation matrices for Denavit Hartenberg algorithm.

The transformation matrices specified in eq[3.17], correspond to the position and orientation of the desired joint articulation that we want to analyze, and if the final reference frame of Baxter arm is the goal, this transformation matrix would become to the values that go from {0} to {7}.

To obtain the complete matrix transformation from {0} to {7}, it can apply the properties of homogeneous transformation matrices, like this:

$$\left[{}^{0}_{7}\mathrm{T}\right] = \left[{}^{0}_{1}\mathrm{T}(\theta_1)\right]\left[{}^{1}_{2}\mathrm{T}(\theta_2)\right]\left[{}^{2}_{3}\mathrm{T}(\theta_3)\right]\left[{}^{3}_{4}\mathrm{T}(\theta_4)\right]\left[{}^{4}_{5}\mathrm{T}(\theta_5)\right]\left[{}^{5}_{6}\mathrm{T}(\theta_6)\right]\left[{}^{6}_{7}\mathrm{T}(\theta_7)\right] \tag{3.4}$$

3.4: Complete Baxter arm transformation matrix from frame {0} to frame {7} using DH intermediate transforms.

As it can be perceived in the transformation matrix obtained in eq[3.4], it only takes into consideration the frames that change in time, which correspond to the shoulder, wrist and elbow of Baxter arm. However, to apply a complete transform from the defined center of reference {W0} to the final end-effector {GL} or {GR}, the already discussed fixed transformation matrices must be applied correctly.

To mathematically express the absolute final position and orientation of the end-effector expressed in homogeneous transformation matrix, the successive expression must be calculated:

$$\begin{bmatrix} {}^{W_0}_{G_L}\mathrm{T} \end{bmatrix} = \begin{bmatrix} {}^{W_0}_{B_L}\mathrm{T} \end{bmatrix} \begin{bmatrix} {}^{B_L}_{0}\mathrm{T} \end{bmatrix} \begin{bmatrix} {}^{0}_{7}\mathrm{T} \end{bmatrix} \begin{bmatrix} {}^{7}_{G_L}\mathrm{T} \end{bmatrix} \tag{3.5}$$

3.5: Expression to calculate absolute transformation matrix from the reference frame {W0} to the left end-effector frame {GL}.

$$\begin{bmatrix} {}^{W_0}_{G_R}\mathrm{T} \end{bmatrix} = \begin{bmatrix} {}^{W_0}_{B_R}\mathrm{T} \end{bmatrix} \begin{bmatrix} {}^{B_R}_{0}\mathrm{T} \end{bmatrix} \begin{bmatrix} {}^{0}_{7}\mathrm{T} \end{bmatrix} \begin{bmatrix} {}^{7}_{G_R}\mathrm{T} \end{bmatrix} \tag{3.6}$$

3.6: Expression to calculate absolute transformation matrix from the reference frame {W0} to the right end-effector frame {GR}.

An important remark is that for these equations, the only variable matrices are the ones that go from the frames {0} to {7}, and the value is obtained from the current measurement of the joint encoders angles θ_1 to θ_7. The other ones were explained before and are constant in time.

The software implementation of the FPK algorithm for this research project can be found at our GitHub repository "baxter-bon-appetit":

- Baxter-Bon-Appetit Forward Pose Kinematic Algorithm (Source Code)

3.3.2 Baxter Inverse Pose Kinematics

The goal of the IPK for Baxter, is to determine the current angles of each one of the degrees of freedom, based on the Cartesian position and orientation of the end effector. To solve this problem, we can proceed to derive a general expression based on an analytical solution.

One key condition to consider is that Baxter has 7 DOF, which makes him an hyper-redundant robot that can usually get to a given point in space in many different ways. This redundancy makes it more complex to calculate an analytical solution taking into consideration each one of the angles as variables. This problem will be solved by a strategic approach in which one of the degrees of freedom will be fixed at a constant value.

This degree of freedom that will have its angle fixed to zero will be the one corresponding to the first elbow joint ("E0", which is θ_3), as it can be seen in fig[3.10]:

Figure 3.10: Baxter e0 joint that will be fixed for the analytical IPK.

Another important distance that will be discarded for this analytical calculation, is L_5, which is really small compared to the other limb distances. This assumption will make it simpler to obtain the analytical IPK Baxter solution.

After the θ_3 and L_5 condition are said, the next part of the problem is to understand the inputs and outputs for getting the IPK.:

Input Matrix:

$$ {}^{0}_{6}\mathrm{T} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & {}^{0}\mathrm{x}_6 \\ r_{21} & r_{22} & r_{23} & {}^{0}\mathrm{y}_6 \\ r_{31} & r_{32} & r_{33} & {}^{0}\mathrm{z}_6 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.7} $$

3.7: Input matrix for Inverse Pose Kinematic analytic solution.

Output variables:

$$[\theta_1, \theta_2, \theta_4, \theta_5, \theta_6, \theta_7] \tag{3.8}$$

3.8: Output variables for Inverse Pose Kinematic analytic solution.

It is important to notice that the indices of the frames are assuming that θ_3 is none, so they are shifted to the next one by one after the third DOF. This means that frames "4, 5, 6, 7" are now "3, 4, 5, 6".

As well as that, keep in mind that the distance from the reference frames of joints {2} to {4}, is the euclidean distance from the center of each frame, that corresponds to: $\sqrt{L_2^2 + L_3^2}$.

The analytical solution can be proved by starting with equation shown in eq[3.9], and applying linear algebra concepts for each one of the DOF.

$$\begin{bmatrix} {}^{W_0}_{G_L}\mathrm{T} \end{bmatrix} = \begin{bmatrix} {}^{W_0}_{B_L}\mathrm{T} \end{bmatrix} \begin{bmatrix} {}^{B_L}_{0}\mathrm{T} \end{bmatrix} \begin{bmatrix} {}^{0}_{6}\mathrm{T} \end{bmatrix} \begin{bmatrix} {}^{6}_{G_L}\mathrm{T} \end{bmatrix} \tag{3.9}$$

3.9: Expression to obtain the analytical IPK solution.

The final angle values are:

$$\theta_1 = atan2({}^0\mathrm{y}_4, {}^0\mathrm{x}_4) \tag{3.10}$$

3.10: Joint angle 1 for the IPK solution

$$\begin{aligned} x &= {}^0\mathrm{y}_4 \\ z &= {}^0\mathrm{z}_4 \\ E &= 2 \cdot L_h \cdot \left[L_1 - \frac{x}{c_1} \right] \\ F &= 2 \cdot L_h \cdot z \\ G &= \frac{x^2}{c_1^2} + L_1^2 + L_h^2 - L_4^2 + z^2 - 2 \cdot \frac{L_1 x}{c_1} \\ t_{1,2} &= \frac{-F \pm \sqrt{E^2 + F^2 - G^2}}{G - E} \\ \theta_{2_{1,2}} &= 2atan(t_{1,2}) \end{aligned} \tag{3.11}$$

3.11: Joint angle 2 for the IPK solution

$$\theta_{4_{1,2}} = atan2(-z - L_h s_{2_{1,2}}, \frac{x}{c_1} - L_1 - L_h c_{2_{1,2}}) - \theta_{2_{1,2}} \tag{3.12}$$

3.12: Joint angle 4 for the IPK solution

$$\left[{}^{3}_{6}\mathrm{R}(\theta_5, \theta_6, \theta_7)\right] = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \tag{3.13}$$

3.13: Rotation matrix from reference frame 3 to reference frame 6 for the IPK procedure.

$$\theta_5 = atan2(R_{33}, R_{13}) \tag{3.14}$$

3.14: Joint angle 5 for the IPK solution

$$\theta_6 = atan2(\frac{R_{21}}{c_7}, -R_{23}) \tag{3.15}$$

3.15: Joint angle 6 for the IPK solution

$$\theta_7 = atan2(-R_{22}, R_{21}) \tag{3.16}$$

3.16: Joint angle 7 for the IPK solution

The Baxter IPK analytical solution will be a necessary algorithm to develop robotic solutions that map the Cartesian space to the joint space. This solution could be improved with a 7 DOF Inverse Pose Kinematics, but the cost of developing a solution that involves every join value, would represent a higher computational cost with lower response times for the robot.

To dive deeper into the software implementation of this algorithm, please feel free to explore our GitHub repository "baxter-bon-appetit" with the source code for the IPK:

- Baxter-Bon-Appetit Inverse Pose Kinematic Algorithm (Source Code)

3.3.3 Baxter Jacobian Analysis

The Jacobian is a matrix of partial derivatives of a vector function. It is extremely useful for finding transformation of coordinates systems.

In robotics, jacobian matrices are commonly used for mapping joint velocities to Cartesian velocities (Craig, 2004). This mathematical tool enables a direct relationship between a fix world frame of a robot to its specific articulation frames.
It can be proved that the jacobians follow the mathematical expression:

$$\left[{}^{k}\mathrm{J}\right] = \begin{bmatrix} \left[{}^{k}_{w}\mathrm{R}\right] & 0 \\ 0 & \left[{}^{k}_{w}\mathrm{R}\right] \end{bmatrix} \left[{}^{w}\mathrm{J}\right] \tag{3.17}$$

3.17: Jacobian transformation to another frame.

Based on the article "Baxter Humanoid Robot Kinematics" (Williams, 2017), it is known the jacobian matrix from frame 4 to frame 0 (^{4}J). This expression is really useful to calculate the jacobian matrix from frame 0 to the end-effector's frame.

The calculations and expressions to find Baxter's jacobian matrices can be found at the Baxter Bon Appetit GitHub repository:

- Baxter Jacobian calculation
- Baxter Jacobian implementation

3.4 Baxter Robot Work-Space

A robot workspace is defined as the set of all reachable end-effector positions (Robot Academy, nd). These multiple configurations could be theoretically infinite, but a general understanding of the workspace can be achieved with a rigorous mapping of the area.

For the further understanding of Baxter robot, this research developed a mapping algorithm that works in the following way:

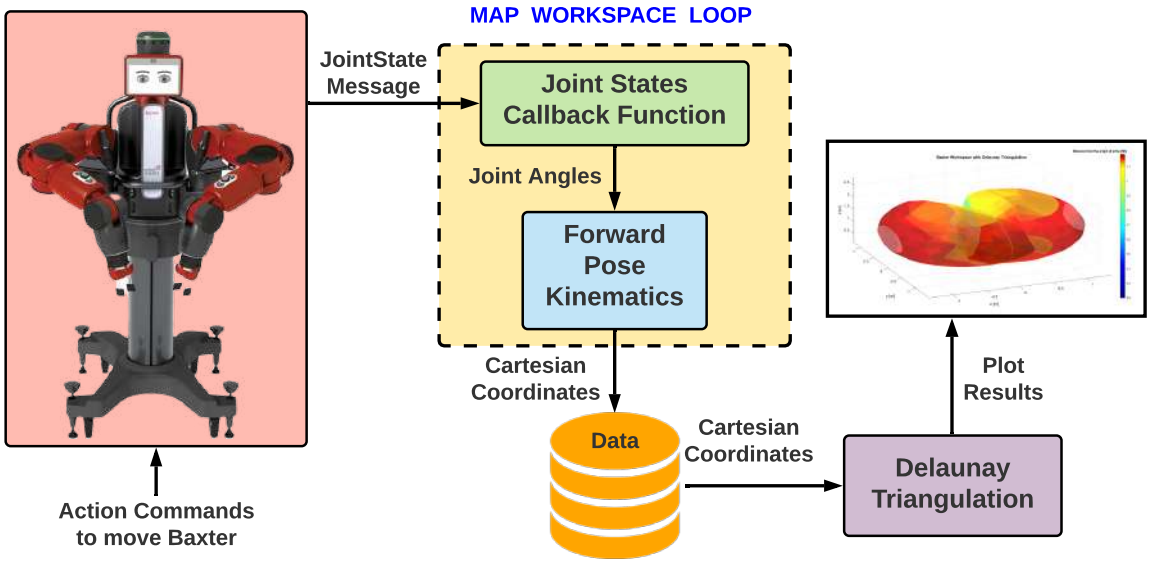


Figure 3.11: Baxter mapping algorithm for calculating robot's workspace. Own development.

As it can be seen in fig[3.11], the mapping algorithm has the following steps:

1. Move Baxter arms to the reachable positions and orientations to replicate multiple trajectories.
2. From a JointState ROS Message, get the current Joint Angles measurements from the internal Baxter encoder sensors.
3. Calculate current Cartesian coordinates from the Forward Pose Kinematics algorithms explained in the previous chapter.
4. Repeat previous steps multiple times, appending the Cartesian coordinates to a database that holds multiple values for the Baxter mapping process.
5. After the data is collected, apply these Cartesian values to the Delaunay Triangulation to obtain the desired workspace plot results.

When the algorithm was implemented for the mapping, the first results were the "General Mapping Workspace Trajectories", that would cover most of Baxter workspace. These trajectories were taken with the explained algorithm for 30 minutes in each of Baxter's arms. It is important to notice, that to avoid robot damage, the trajectories were implemented by moving Baxter arms manually by touching the cuff sensor and driving the arms as much as possible.

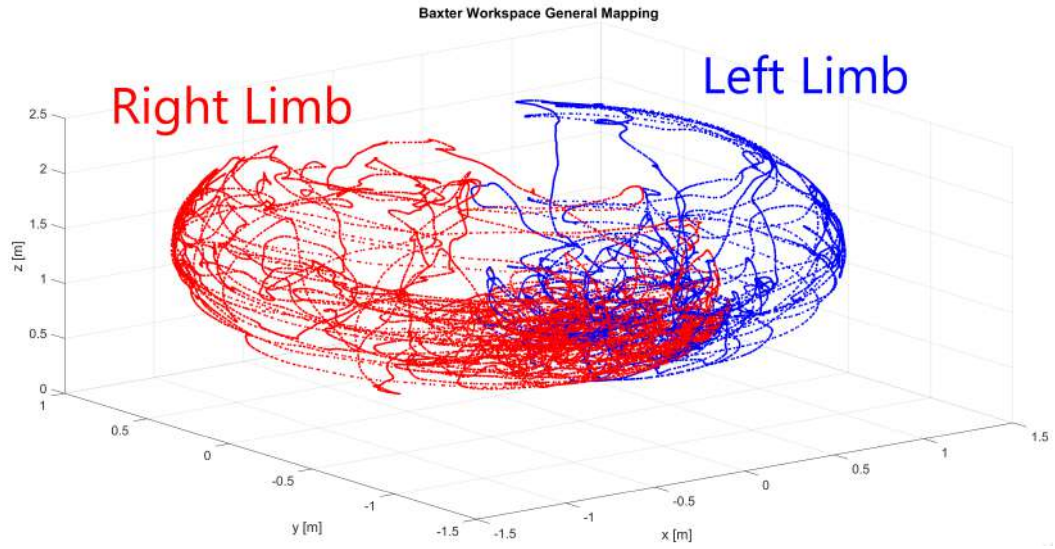


Figure 3.12: Baxter robot general mapping for the workspace. Own development.

After the database of the trajectories for the Cartesian values was filled, the Delaunay Triangulation was applied to the overall results. The objective of this process, was to obtain the maximum reachable spherical surface for Baxter robot workspace. The result is extremely valuable, because it gives us some important information, such as:

- Baxter robot's workspace shape and symmetry.
- Baxter robot's limits for X, Y, and Z dimensions.
- Overview of good and bad manoeuvrability positions for both of Baxter's arms.
- A good understanding of the mathematics to relate joint-states with Cartesian-coordinates.

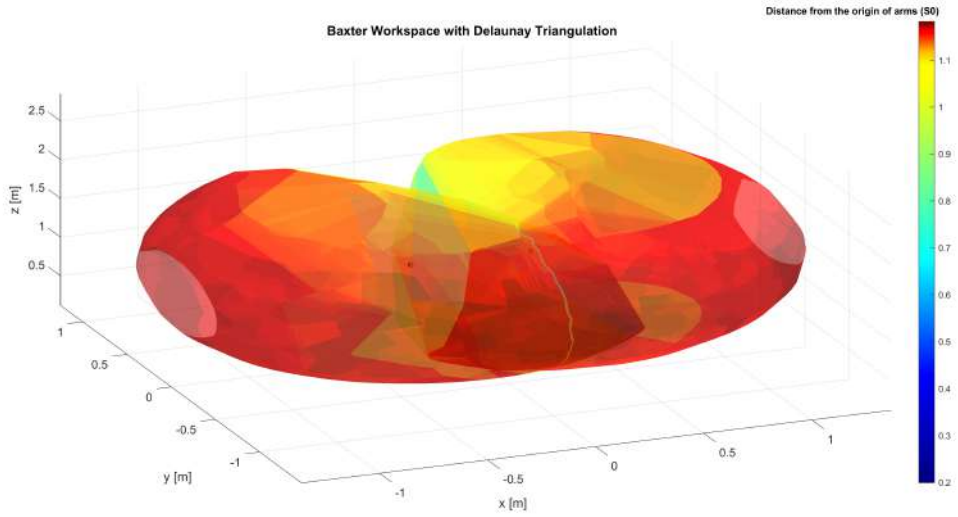


Figure 3.13: Baxter workspace after Delaunay Triangulation. Own development.

The algorithms to map Baxter workspace can be analyzed at our GitHub repository "baxter-bon-appetit" with the source code for both processes explained in fig[3.11]:

- Baxter-Bon-Appetit MATLAB Delaunay Triangulation Algorithm (Source Code)
- Baxter-Bon-Appetit Map Workspace Algorithm (Source Code)

3.5 Baxter Robot Tool

A robot's tool corresponds to a functional extension of the robot's basic capabilities. This final component not only implies a change in the mathematical equations governing the robot's dynamics, but also allows extending the practical behavior of the features that the robot can perform in its classical programming.

For this application (Baxter-Bon-Appetit), the most important capability to be implemented with Baxter robot, is to be able to carry out food for an user. This circumstances, imply that the tool designed for Baxter robot must follow up a series of requirements, such as:

- The tool must be able to be placed firmly on Baxter hand.
- The tool must carry out (hold) food.
- The tool must be able to carry different cutlery utensils, so that they can be changed easily depending on the patient and type of food.

After taking into consideration these requirements, it was chosen that a 3D printed design would satisfy all the requirements and be cost-efficient for the economic budget of this research project. The 3D printed approach was a viable way to develop an accurate mechanical tool, that would match the necessary tolerances of Baxter's arms.

First of all, the software in which the design was made, was Autodesk Fusion 360, a cloud-based 3D modeling, CAD, CAM, CAE, and PCB software platform for product design and manufacturing (Autodesk, 2021). This allowed us to work simultaneously on the design and develop a practical re-usable tool system.

The design of tool had to match the following Baxter hand attaching system, so that it could be placed correctly and guarantee a stable connection to Baxter arm:

Figure 3.14: Baxter hand for attaching the tool. Taken at EIA University Laboratory.

Based on the hand, it was developed a design that matched the default Baxter mechanism for adding external tools designed by the user. This solution was developed as a stereolithography CAD design (also known as STL), which is an universal format that can be

understood by most CAD-CAM software and 3D printers.

The overall mechanical solution achieved for Baxter-Bon-Appetit research project with Fusion 360 software can be seen at figs[3.15, 3.16]:

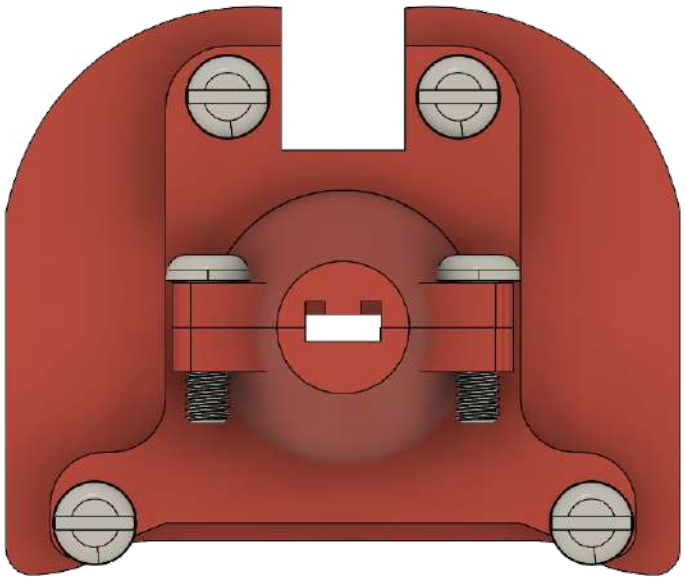
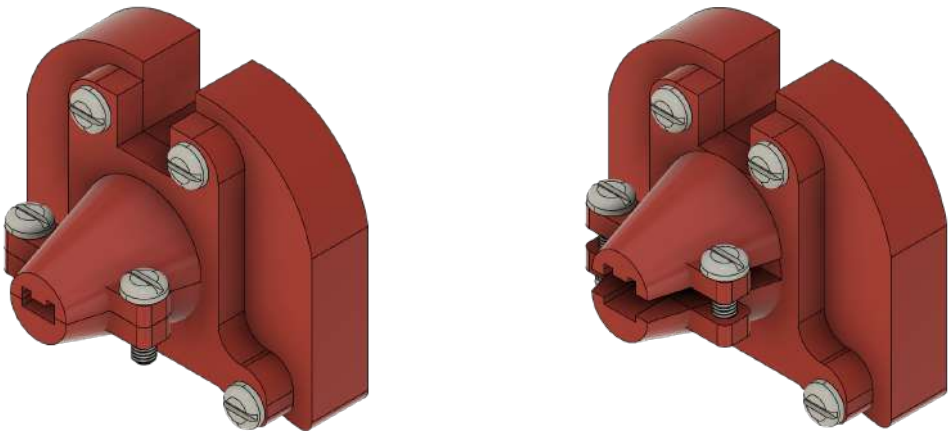


Figure 3.15: Baxter tool front view in Fusion 360 software. Own development.

One important advantage that is full-filled with the developed design, is that there can be multiple cutlery coupled in the tool, so that the user can change the necessary cutlery, based on the food that he or she wants and also be able to change them based on the user (for more than one different patient).



(a) Baxter tool general view closed. Own development.

(b) Baxter tool general view opened. Own development.

Figure 3.16: Baxter tool viewed opened and closed for attaching the cutlery. Own development.

After the general measurements were implemented and verified at the CAD STL design, the next step was to print it into a real tool by a 3D printer. This process was achieved successfully with the "Ender-3 V2 3D Printer", a low-price system, suitable for business start-ups and research projects (Creality, 2021).

This 3D printer was able to create the tool in 4 hours and 30 minutes. The reason of this time, is because the tool was created with a high-density speed constant, so that it would be more resistant to fatigue and mechanical stresses.

(a) Printing process of Baxter tool in action.

(b) Printing process of Baxter tool finished.

Figure 3.17: Baxter tool 3D printing process.

When the design was completely printed, the next part of the methodology was to verify that it could fit correctly (and properly) into Baxter's hand. This was verified and was achieved successfully at EIA University Laboratory.

The following images show the final result of Baxter's tool to correctly be able to give food to the users in a correct and natural way:

Figure 3.18: Baxter tool general view implemented on real life. Taken at EIA University Laboratory.

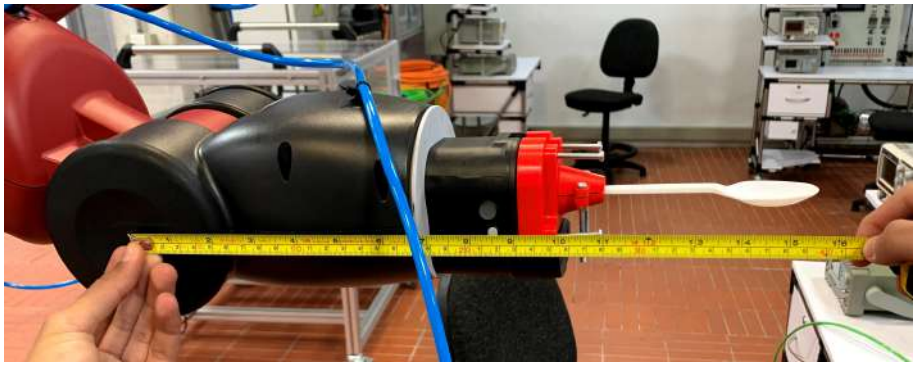


Figure 3.19: Baxter tool side view implemented on real life. Taken at EIA University Laboratory.

The results of Baxter tool for giving food correctly, will be considered as an important condition to take into account in the mathematical calibrations of the FPK algorithm (as an additional offset of the final Baxter hand transformation).

3.6 Baxter Physical Configuration

For Baxter-Bon-Appetit project, the physical workspace configuration for the robot/patient space is a differential factor to fulfill the development of an active feeding system. This configuration is a deterministic decision that will guide the design of the solution and will

modify the evolution of the project based on how it is chosen.

To determine the physical configuration, there are some key values to take into consideration:

1. The patient must be seated facing directly to one of Baxter's cameras.
2. The patient must be located in a place where Baxter maneuverability is high.
3. The position of the camera and the patient should be always the same (or really similar).

Based on these user requirements, the chosen configuration for Baxter and the patient is the one shown in figures fig[3.20,3.21]:

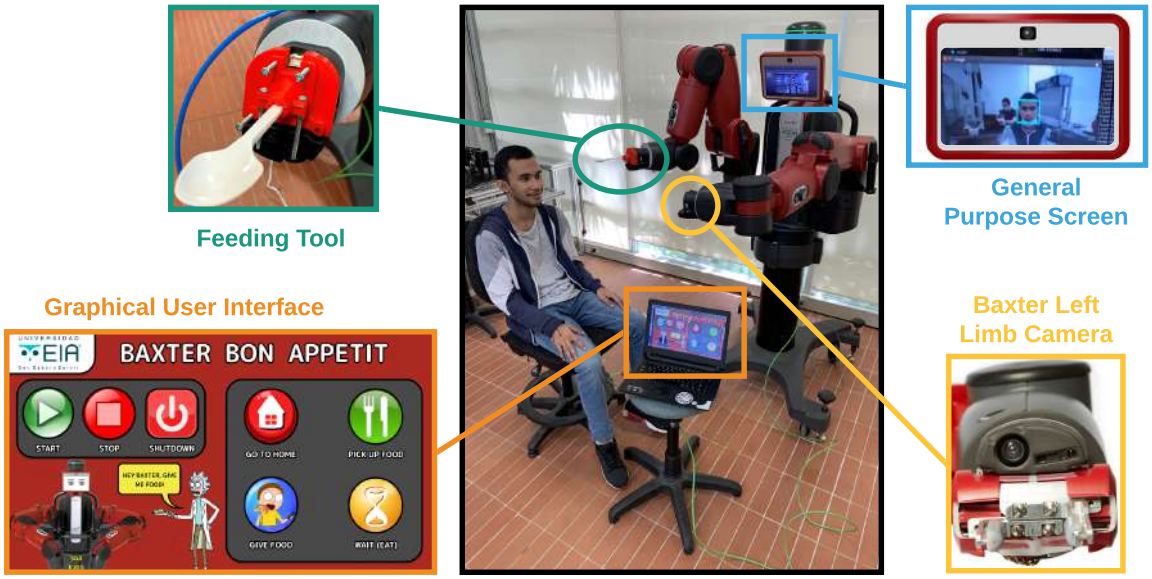


Figure 3.20: Baxter physical configuration explained. Own development.

(a) Physical configuration for Baxter and the user (side view). Taken at EIA University Laboratory.

(b) Physical configuration for Baxter and the user (front view). Taken at EIA University Laboratory.

Figure 3.21: Physical configuration for Baxter and the user. Taken at EIA University Laboratory.

As it can be seen in the shown figures, the configuration was chosen because Baxter's right arm can reach the user's mouth directly in order to do the feeding action. This will enable Baxter cobot to have a great functionality of helping the patient to eat his meal in a dynamic and active way.

Another remarkable condition of the exposed arrangement, is that the camera that is going to be used is directly embedded on Baxter's left arm, implying that it can be tweaked in position and orientation (something that would not be achievable with the camera on Baxter's head). As well as this extra translation and rotation approach, one advantage of using a camera that is directly designed into Baxter's arm, is that, with the encoder measurements, it will always be possible to find the current position of the camera (even after external modifications). This measurements will be explained in future chapters.

One extra interesting benefit of implementing this physical layout, is that the patient will always be facing directly to the camera, and even after having its meal given to his mouth, his face will be mostly visible to the camera, resulting in a great clear-vision approach for constantly detecting the user's face.

The research will use the exposed configuration for the control, computer vision and integration algorithms.

Chapter 4

Computer Vision

Most robotic systems that are aware of their environment, use sensors of various types to ensure proper data acquisition to provide valuable information for the dynamic variables of interest for the system.

For the case in which cameras are used as primary sensors, it is extremely important to understand that the acquired signal can not go directly to a control scheme, but must be processed and debugged through computational algorithms that transform the multiple RGB matrices obtained by these cameras, in more accurate variables according to the control scheme.

The computer vision algorithms that will be implemented on Baxter robot, will start directly from the embedded robot's cameras, as it was described in the design exposed at fig[2.3]. This decision

4.1 Baxter Camera Configuration and Setup

To implement a computer vision algorithm, it is relevant to have a deep understanding on the camera specifications. The most relevant characteristics for Baxter cameras are exposed in table tab[3.3] and it is important to guarantee that the camera's resolution is fixed in a constant value (so that the trained networks work properly).

The chosen value for the camera resolution is "640x400 pixels". This value is obtained from the best practices specifications given in Baxter camera SDK documentation (Rethink Robotics, 2015g).

The software and mathematical implementations for this chapter, will be calculated based on the chosen resolution of Baxter's left hand camera, so that the algorithms work in a proper way.

4.2 Baxter Camera Manipulation

To understand how Baxter's cameras work, we highly recommend to dive deeper into the official SDK documentation, so that every detail can be considered (Rethink Robotics,

2015f).

Baxter cameras are relevant sensors designed to gather direct information from Baxter's environment as fast as possible. These devices are assembled and integrated with Baxter functionalities through ROS. To control the cameras and the information acquired, we must understand some key features:

- Cameras are controlled with an Application Programming Interface (API) that makes it possible to interact with their software configurations.
- Camera information is understood as ROS messages and topics that are constantly being updated and published based on the requirements.
- To read Baxter's cameras photos/videos, they must be transformed into a standard image matrix-wise format.
- Only two out of the three Baxter's cameras can operated at the same time.

For the discovery of Baxter's cameras and their functionalities, the robot was controlled and operated through the official camera control algorithms given by Rethink Robotics:

- RethinkRobotics/baxter_tools: camera control (Source Code)

After understanding the underlying low-level configurations, the computer vision algorithms designed for this research, were developed using the internal messages of Baxter robot regarding its cameras and the necessary subscribers to their information. A deeper analysis of these implementations, will be discussed in the source code and in ROS integration section.

4.3 Face Detection Algorithms

For Baxter-Bon-Appetit project, the face detection algorithms are considered to be one on the first key components of the whole control structure. This is due because they convert the original images obtained from Baxter's cameras (complex matrices that do not give straight valuable information), into strategical information that is understood as a reference feature for mapping the user's face.

The two algorithms that are going to be developed for this research, are variants of:

1. Vioja-Jones Classifier with Haar Cascade Feature Selection.
2. Face Recognition API based on deep learning using dblib.

Both face detection algorithms are going to be built on top of the ones mentioned, as a particular extension of their functionalities.

4.3.1 Viola-Jones Classifier

The Viola-Jones classifier was developed by Paul Viola , Michael Jones, and it was one of the pioneer programs to correctly solve the face detection problem in a fast and accurate way (Viola and Jones, 2001). The problem that it solved was to be able to detect any face in a given input image.

This algorithm expects the face to be positioned directly to the camera so that it is arranged as a frontal upright face. As well as this, it important that the face is not tilted to either one of the sides, avoiding inclinations.

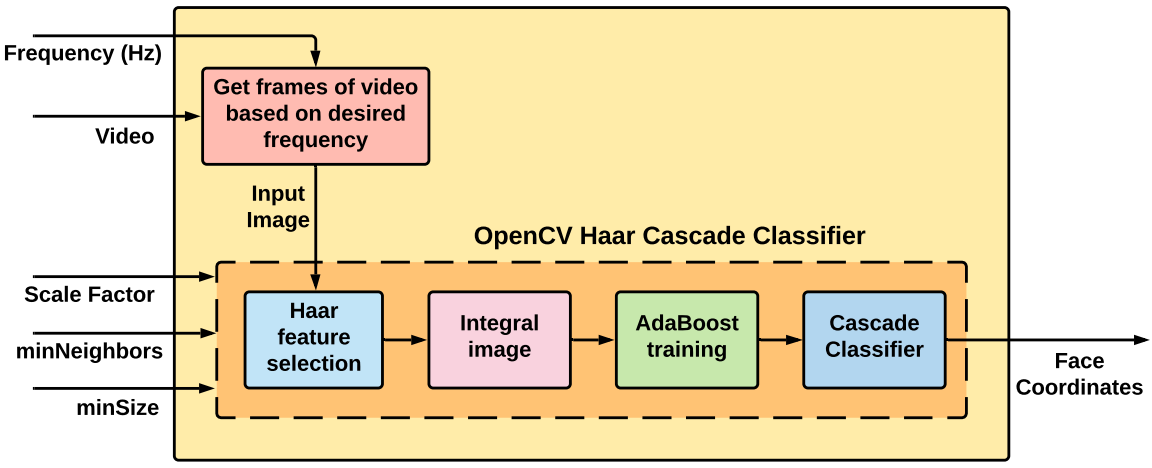


Figure 4.1: Face detection block diagram implementing Viola-Jones Haar Cascade Classifier and AdaBost. Own development.

The algorithm that was developed on top of Viola-Jones classifier, has multiple extra inputs, such as frequency, video acquisition, scale factors, minimum neighbors, minimum size and other internal parameters.

The described face detection algorithm was developed with Python programming language with the libraries of OpenCV and numpy. The source code of Baxter Bon Appetit project for face detection with Viola-Jones approach, can be found at "baxter-bon-appetit" GitHub repository:

- Baxter-Bon-Appetit Face Detecion Algorithm based on Viola-Jones Framework (Source Code)

Each one of the steps of the block diagram is going to be explained with a high level of abstraction:

4.3.1.1 Get frames of video based on desired frequency

This process is the one that manages how the input to the OpenCV Haar Cascade Classifier is going to be analyzed. This implies, that the sample time and video conditions are

configured as general inputs to the algorithm.

To do this step, a continuous loop is processing the video input in a Python Thread, and it has specific wait times to only process the video input at a desired time. This processing converts the video frames into specific images with a given window size.

4.3.1.2 Haar feature selection

Feature selection is a mandatory step in object detection algorithms. For Viola-Jones detection framework, the Haar feature selection has to process and analyze more than 180.000 features (in a 24x24 average image), and has the main goal of getting the useful features and reducing the overall processing time.

The four basic Haar features implemented in Viola-Jones algorithm are:

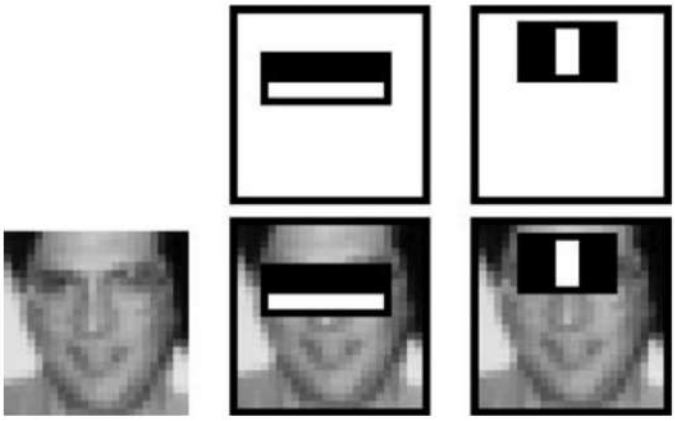


Figure 4.2: Example of Haar features used for Viola-Jones algorithm. Taken from (Zaghetto et al., 2016).

These features will play a special role in the determination of what is a face or not. These features must be calculated and analyzed in a short amount of time and the selection of the relevant ones is usually implemented with criteria such as:

1. Horizontal white-black feature.
2. Vertical white-black feature.
3. Vertical white-black-white feature.
4. Diagonal white-black feature.

4.3.1.3 Integral Image

The integral image approach, also known as summed-area-table is a data structure and algorithm that is commonly used in computer vision and computer graphics (Mathworks, 2021). This process calculates the sum of the values of a matrix in a subsection of the grid. This implies that the original matrix is then processed into a new integral matrix

that has relevant values of the inner elements sums.

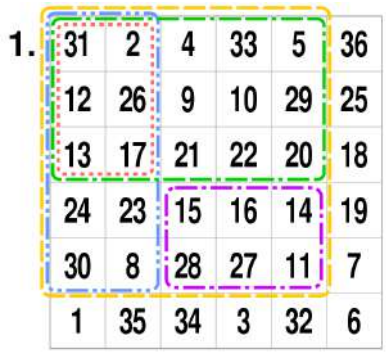
The advantage of generating integral images, is that they reduce significantly the amount of operations involved in computer vision algorithms, and therefore, they improve the performance of the algorithm undoubtedly.

The process of generating an integral image is based on the following equation:

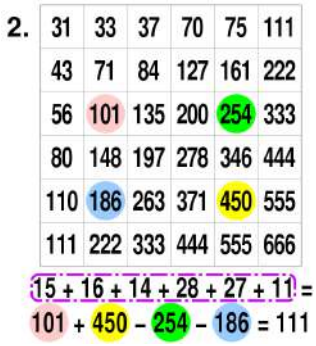
$$\begin{aligned} ii(x,y) &= \sum_{x'<=x,y'<=y} i(x',y') \\ ii(x,y) &= \text{Integral Image} \\ i(x,y) &= \text{Original Image} \end{aligned} \tag{4.1}$$

4.1: Equations for integral image calculation

This algorithm sums up the corresponding image pixels, and generates the final integral image that will be implemented in the following steps.



(a) Original image. Taken from (Cmglee, 2021)



(b) Result of integral image calculation. Taken from (Cmglee, 2021)

Figure 4.3: Example of a process of integral image calculation.

4.3.1.4 AdaBoost Training

The AdaBoost training, also called Adaptive Boosting, is an ensemble method in Machine Learning algorithms. In order to understand the adaBoost method, it is necessary to know how the boosting algorithm actually works, this technique makes 'n' number of decisions trees during the data training time, as the first decision tree/model is made, the incorrectly classified record in the first model is given priority. Only these records are sent as input for the second model. The process goes on until we specify a number of base learners we want to create (Kumar, 2020).

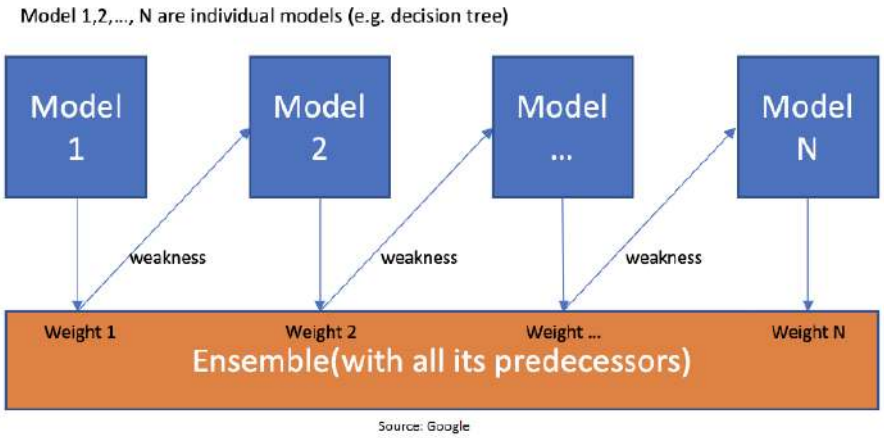


Figure 4.4: Example of the decision trees created in the training period by boosting technique. Taken from (Kumar, 2020).

In AdaBoost algorithm, the main objective is to improve the performance of other type of learning classifiers. It combines the output of the other given learning algorithms, into a weighted sum that ends up in the output of the boosted classifier. The algorithm is adaptive due to its abilities to improve the performance from the weak learners. The most interesting part of AdaBoost process, is that even if the performance of the intermediate algorithms is not that great, by guaranteeing that their accuracy is better than guessing, the final model can be proven to converge to a strong-wise learner (Winston, 2014).

For the Viola-Jones algorithm, the training data-sets are composed of:

1. 4960 images of faces.
2. 9544 non-face images.

4.3.1.5 Cascade Classifier

A cascade classifier is the strategical union of individual classifiers, attaching them to gather an input as the output of the previous classifier. These classifiers are trained individually based on a series of "positive" inputs and "negative" inputs (Gama and Brazdil, 2000).

The importance of developing cascading classifier solutions, is that they can be applied directly to a region of a matrix (in this case, image), and detect object from it.

The main advantage of Viola-Jones cascade classifier, was not only the fact that it had a great accuracy, but also the speed of the calculations, meaning that it could be ran in low-power CPU devices (Viola and Jones, 2001).

4.3.2 Face Recognition Deep Learning Framework

The second approach to achieve face detection for Baxter Bon Appetit project, is by implementing a deep learning library that focuses on face detection and face recognition

(Geitgey, A et al, 2021).

The library is called “Face Recognition” and it implements multiple low-level deep neural networks and convolutional networks built on top of o the famous C++ library called “dlib”. This dlib C++ library is a modern framework that has robust machine learning algorithms and tools for developing complex software solutions for real-life problems (E, King et al, 2021).

The advantage of implementing “dlib” with the Face Recognition module, is that it’s built on top of C++, increasing its performance and the speed of the algorithms.

The principal author of the library, Adam Geitgey, has an amazing series of articles explaining the undelying processes of the deep neural networks algorithms that generate the amazing results of his Face Recognition Framework. Some of the articles can be found in his tech blogs:

- Deep Learning and Convolutional Neural Networks, by Adam Geitgey

To understand how the library works in a high-level of abstraction, we can think that it works as a convolutional complex neural network that receives an input image and has multiple multi-threading processes with many layers, that is able to learn complicated patterns based on a hierarchy or conceptual structure that propagates through the network (Geitgey, A et al, 2021).

The processes of the library, can be simplified as follows:

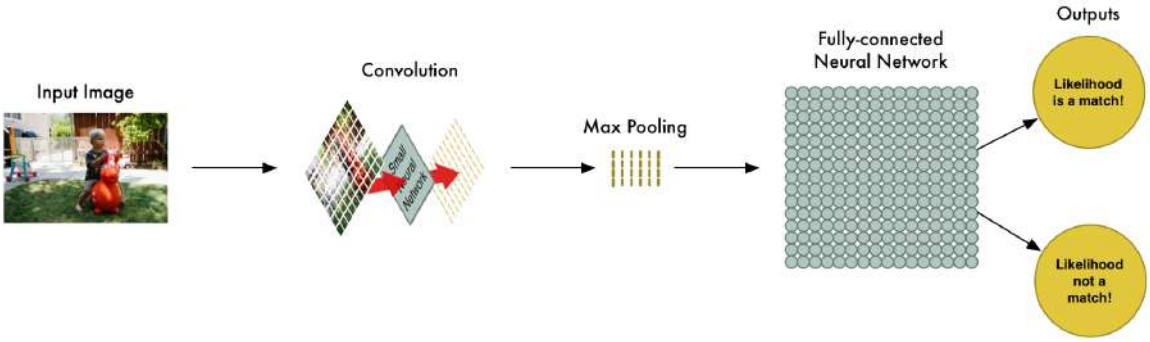


Figure 4.5: Example of a convolution neural network explained in a simple way. Taken from (Geitgey, 2016).

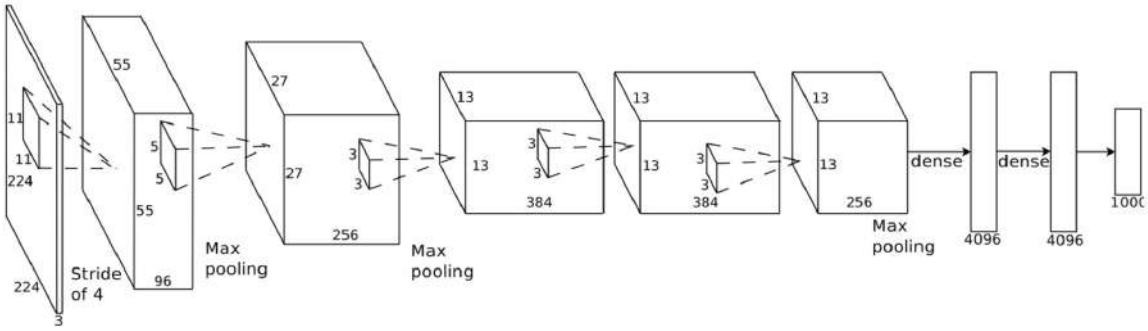


Figure 4.6: Example of a convolution neural network explained in a mathematical way. Taken from (Geitgey, 2016).

The previous figures fig[4.5, 4.5], are the basic blocks to implement the face detection algorithms for our research project. These mathematical concepts and algorithms are implemented in the libraries, and they are the low-level steps that make it possible to detect a patients face on a given video.

The development of the face detection solution implementing the face recognition framework can be found at:

- Baxter-Bon-Appetit Face Detection Algorithm based on Face Recognition Framework (Source Code)

4.4 Selection of Face Detection Algorithm

Now that we have two face detection algorithms written in Python and compatible with the conditions of the operating system of the Baxter robot, it is time to perform a series of real tests to validate which one of the algorithms has better performance.

This performance will be validated through some criteria established previously, and a series of tests will be performed with the same conditions for both algorithms, in order to decide which one is the most appropriate for this research.

There will be a total of 3 tests validating the following features:

- Time to process a single image frame.
- Percentage of correctly detected face frames in video.
- Ability to only detect the biggest face frame in video.
- CPU resources.
- Memory resources.
- Software dependencies.

4.4.1 Test 1: general movements facing camera

The first test had the objective of validating if the face detection algorithms were able to correctly detect the patient's face in a real controlled scenario. This test is the most similar one to a real user's environment for Baxter Bon Appetit.

The input video was the same for both algorithms, making the input of the system identical. Both processes performed really well in terms of the most important parameters.

The video had a person looking directly into the camera and moving its face in a vertical and horizontal way (always being facing the camera without any angle).

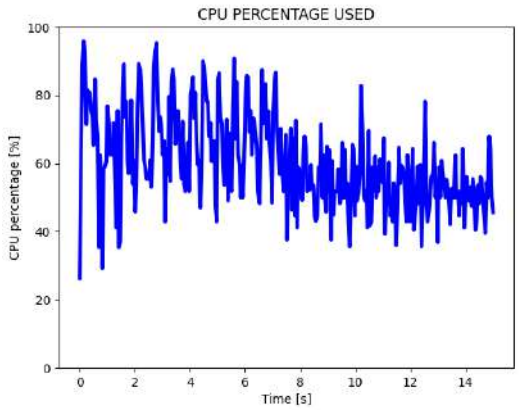
(a) Sample frame of face detection algorithm in test 1 with Viola-Jones approach. Own development.

(b) Sample frame of face detection algorithm in test 1 with Face Recognition approach. Own development.

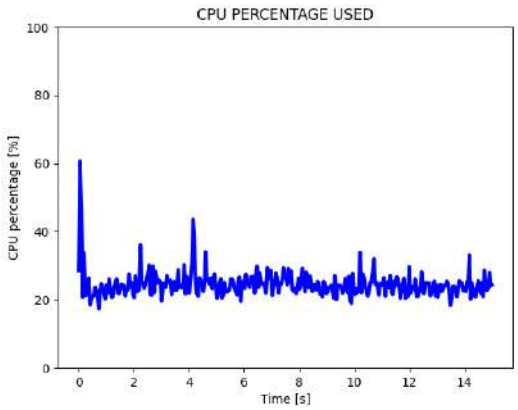
Figure 4.7: Sample execution of test 1 for face detection algorithms.

As it can be analyzed in figure fig[4.7], both algorithms had a great performance in terms of detecting the face. However, it is clear that the square of the face is slightly different for each algorithm. For example, in the second picture, the square tends to be lower than the first one. This remark will be important for estimating the exact position of the mouth of the user.

In terms of performance, some important results of this test are:

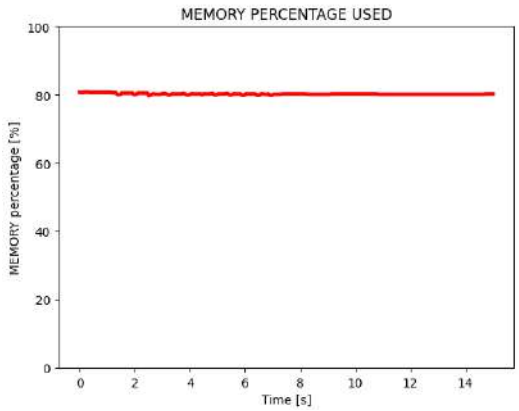


(a) Results of CPU percentage plot for test 1 with Viola-Jones approach. Own development.

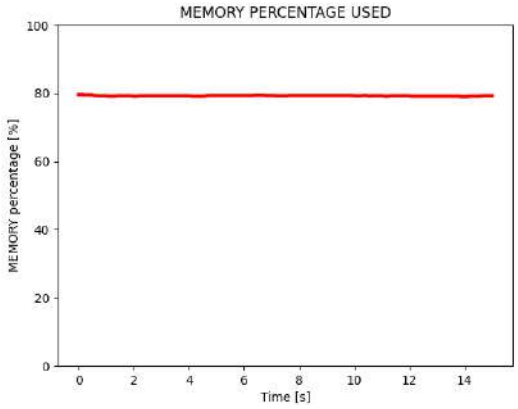


(b) Results of CPU percentage plot for test 1 with Face Recognition approach. Own development.

Figure 4.8: Execution of test 1 for face detection algorithms CPU percentages.



(a) Results of memory percentage plot for test 1 with Viola-Jones approach. Own development.



(b) Results of memory percentage plot for test 1 with Face Recognition approach. Own development.

Figure 4.9: Execution of test 1 for face detection algorithms memory percentages.

After running the comparison scripts for both algorithms, the compact results were:

Measurement	Viola-Jones	Face Recognition
Time to process single image	0.45s	0.52s
Percentage of correctly detected face frames in video	100%	100%
Does it only detect the biggest face frame?	Yes	Yes
Spent CPU resources	57.97%	22.75%
Spent memory resources	80.04%	79.83%
Software dependencies	15	21

Table 4.1: Results of test 1 for face detection algorithms. Own development.

As shown in table tab[4.2], the most important criteria (which is the percentage of cor-

rectly detected face frames in the video), had a 100% success rate for both algorithms. However, the overall performance of CPU and memory resources was better with the Face Recognition approach.

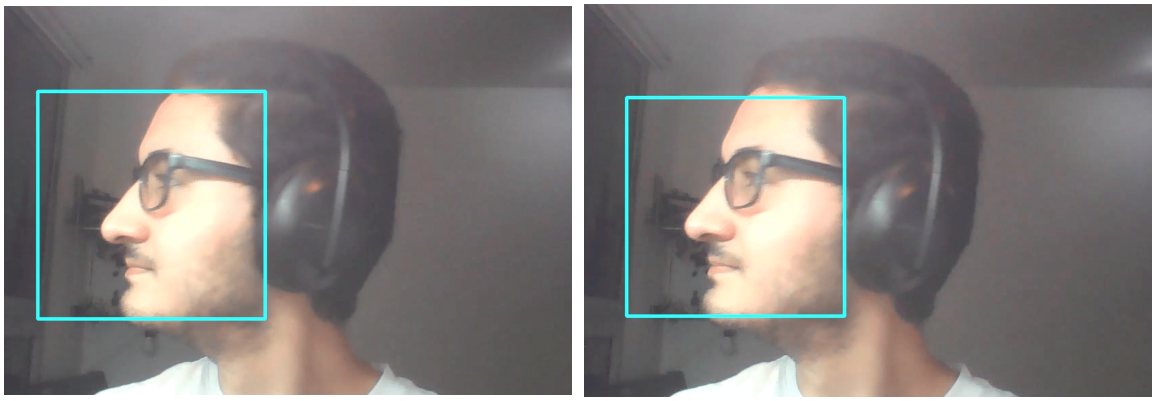
The time of processing a single image (assuming loading libraries and initial dependencies), was really similar for both algorithms, so its not a relevant differential condition.

Both algorithms did a great job in this test.

4.4.2 Test 2: general movements with face rotations

The second test had the goal of validating if the face detection algorithms were able to correctly detect the patient's face in a scenario where the user would rotate its face with extreme angles (greater than 90°).

The input video was the same for both algorithms, making the input of the system identical. The video had a person that started looking directly to the camera, and after that, the person rotated his face around in the 2 principal axis directions (horizontal and vertical rotations). The main objective of this inclinations, was to determine which of the algorithm was able to perform better in extreme scenarios where the user would tilt his head.



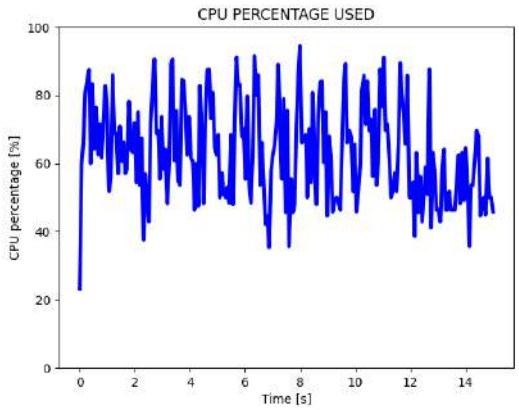
(a) Sample frame of face detection algorithm in test 2 with Viola-Jones approach. Own development.

(b) Sample frame of face detection algorithm in test 2 with Face Recognition approach. Own development.

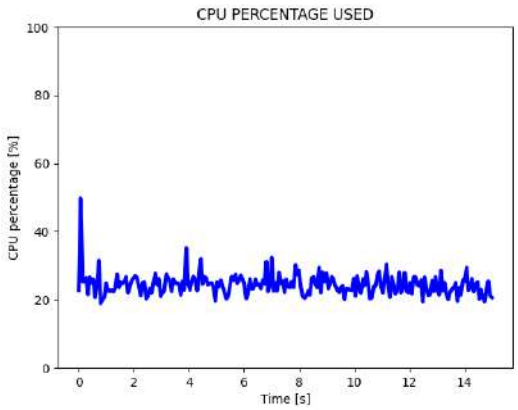
Figure 4.10: Sample execution of test 2 for face detection algorithms.

The previous figure fig[4.10], shows that both algorithms did an excellent job when it was necessary to detect the user's face when the person is looking to the side. These results are a great example of showing that Baxter Bon Appetit project will be reliable, even the the patients look extremely to the sides, scenario common for when someone is being called by another person in a room.

When analyzing the performance, the results were the following:

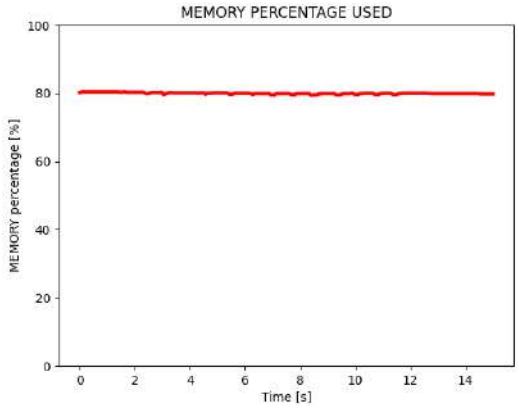


(a) Results of CPU percentage plot for test 2 with Viola-Jones approach. Own development.

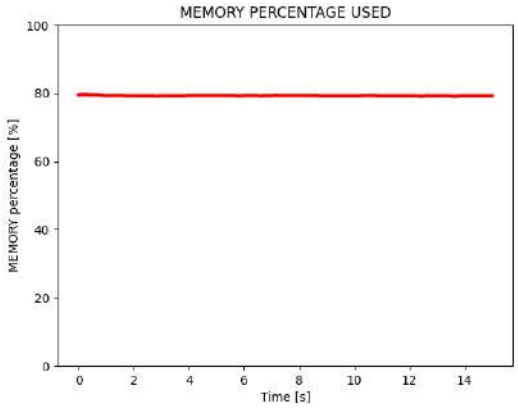


(b) Results of CPU percentage plot for test 2 with Face Recognition approach. Own development.

Figure 4.11: Execution of test 2 for face detection algorithms CPU percentages.



(a) Results of memory percentage plot for test 2 with Viola-Jones approach. Own development.



(b) Results of memory percentage plot for test 2 with Face Recognition approach. Own development.

Figure 4.12: Execution of test 2 for face detection algorithms memory percentages.

After running the comparison scripts for both algorithms, the compact results were:

Measurement	Viola-Jones	Face Recognition
Time to process single image	0.45s	0.52s
Percentage of correctly detected face frames in video	80.70%	71.05%
Does it only detect the biggest face frame?	Yes	Yes
Spent CPU resources	61.03%	25.63%
Spent memory resources	80.05%	79.89%
Software dependencies	15	21

Table 4.2: Results of test 1 for face detection algorithms. Own development.

These results are really interesting, because on the "Percentage of correctly detected face

frames in video" measurement, there is a significant gap between both approaches, being more accurate the Viola-Jones processes, when detecting faces that are rotated. This is a major discovery, because it will be a crucial factor for the selection of the final face detection algorithm.

The Viola-Jones algorithm had the best results on this test.

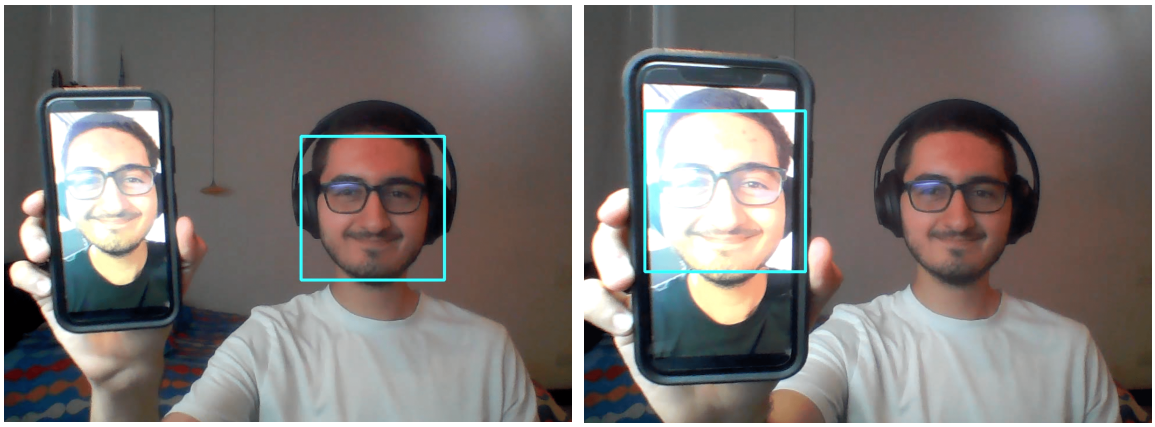
4.4.3 Test 3: multiple faces in video

The final test for the comparison of both face detection algorithms was a video with multiple faces on it. The importance of this test, was to prove that each process was able to only detect the biggest face in the frame, and that face would be the one of the patient in a real-life scenario.

In order to be able to replicate the same conditions for the test, two main factors were considered. The first one was that the processing was done on the same video, and the second one, is that the faces involved in the video were exactly the same (and only changed in size through time).

The expected results, were that the algorithms would be able to alternate in the selected face, according to the size of the biggest face at any given time.

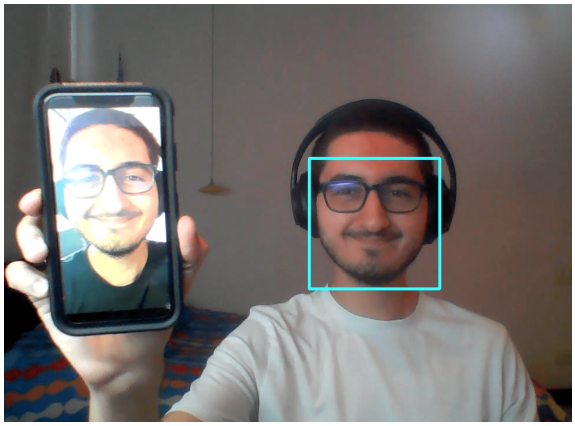
The results were amazing for both algorithms, and they did show only the biggest one in the video.



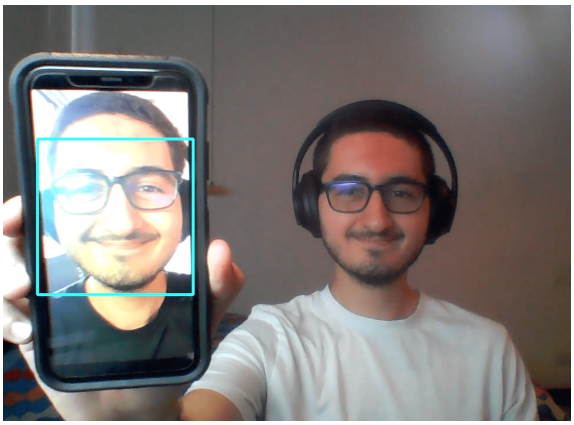
(a) Example of video frame executing face detection algorithm with Viola-Jones approach and detecting biggest face.

(b) Example of video frame executing face detection algorithm with Viola-Jones approach and detecting biggest face.

Figure 4.13: Sample video frames executing face detection algorithm with Viola-Jones approach and detecting biggest face.



(a) Example of video frame executing face detection algorithm with Face Recognition approach and detecting biggest face.



(b) Example of video frame executing face detection algorithm with Face Recognition approach and detecting biggest face.

Figure 4.14: Sample video frames executing face detection algorithm with Face Recognition approach and detecting biggest face.

After getting the results for both algorithms in test 3, the results were outstanding. Both were able to only detect the biggest face and correctly do this process at all frames of the video. The results can be shown in figures fig[4.13, 4.14].

One important remark, is that the rectangle of the detected frame had some variations in size and position for each algorithm, a relevant aspect that will be important to consider in the development of Baxter Bon Appetit project.

4.4.4 Source code for Baxter Bon Appetit Face Detection Comparison Algorithms

The algorithms that enabled the plots and information gathering, were developed using Python programming language and can be found at Baxter Bon Appetit GitHub repository:

- Baxter-Bon-Appetit Sample Scripts for Face Detection with Viola-Jones Framework (Source Code)
- Baxter-Bon-Appetit Sample Scripts for Face Detection with Face Recognition Framework (Source Code)

4.4.5 Selecting the Face Detection Algorithm for Baxter Bon Appetit

As seen in sections sec[4.4.1, 4.4.2 4.4.3], both algorithms were evaluated in multiple use cases and scenarios. The general results pointed that both would do a great job in Baxter Bon Appetit research project, because of their great performance and accuracy.

The most important measurement for the evaluated algorithms was the one of "Percentage of correctly detected face frames in video". This is due to the fact that the process

must detect the user's face as frames much as possible. Based on this requirement, on section sec[4.4.2], the Viola-Jones algorithm had a better percentage accuracy, than the Face Recognition algorithm (with about 10% of improvement).

Even if the CPU and memory percentages are not as great in the Viola-Jones approach as in the Face Recognition approach, Baxter robot has a powerful processor and RAM specifications to execute multi-threading programs, meaning that these conditions will not be a limiting factor for the project.

Based on these analyses, the research will be developed executing all the face detection algorithms with the Viola-Jones approach (see section sec[4.3.1]).

4.5 Evaluating Viola-Jones Face Detection Algorithm in Multiple Users

Now that the algorithm for face detection is selected, one important part of the development of Baxter Bon Appetit research, is to verify that the face detection results work properly for multiple individuals, looking forward to real-life conditions.

To validate the Viola-Jones framework with multiple users, 49 pictures of friends and colleagues were collected to verify that the algorithm would work correctly with different gestures and person conditions. All of these individuals approved to be added to this research and had their photos published.

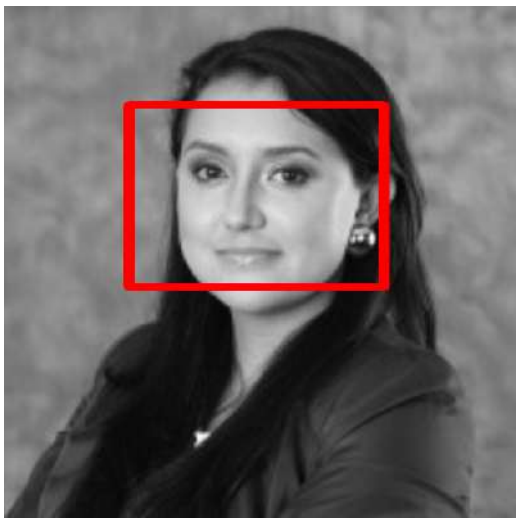
The process of this validation was to give the Viola-Jones algorithm an input image (with a face on it), and it had to mark an approximate rectangle on top of each face. For example:

(a) Example of input photo for validating Viola-Jones algorithm.

(b) Example of output photo after executing Viola-Jones algorithm.

Figure 4.15: Example of Viola-Jones algorithm for male user.

(a) Example of input photo for validating Viola-Jones algorithm.



(b) Example of output photo after executing Viola-Jones algorithm.

Figure 4.16: Example of Viola-Jones algorithm for female user.

After the validation of the face detection algorithm for each user, the results were successful and it detected correctly a total of 49 individuals (25 males and 24 females).

Figure 4.17: Collage of different users with their faces detected using Viola-Jones algorithm. Own development.

4.6 Mouth Pose Estimation Algorithm

The goal of this part of the process, is to obtain a mathematical representation for the specific point of the user's mouth with respect of the main robot's frame. This can be done with a series of strategical transformations through Baxter robot, starting from {W0} (word frame) and ending in {F} (face frame).

To calculate the transformation matrix for the user's mouth at any given time, we must consider the already explained transformations in fig[3.8] (from {W0} to {GL}), and after that, apply the next transformations to map the coordinates to the user's face. This would be the following expression:

$$\left[{}^{0}_{F}\mathrm{T}\right] = \left[{}^{0}_{GL}\mathrm{T}\right]\left[{}^{GL}_{CV}\mathrm{T}\right]\left[{}^{CV}_{F}\mathrm{T}\right] \tag{4.2}$$

4.2: Complete mathematical transformations from frame {0} (world reference) to frame {F} (face reference).

As it can be observed in fig[4.18], the position and orientation for the left limb of Baxter robot was selected in a strategical configuration. This setup had the objective of having a wide reach of the user's face and maintaining the horizontal orientation fixed. Another beneficial factor of this configuration, is that the user will be approximately located in one of the best positions for the right limb in terms of manoeuvrability (see fig[3.12]), giving the robot an excellent and reliable area to move its right limb and reach correctly the user's mouth to feed the patient.

In this section, the last two transformation matrices of eq[4.2] (from {GL} to {CV} and from {CV} to {F}) will be discussed in a detailed way. For this chapter, the naming convention of {GL} will be equivalent to {T}.

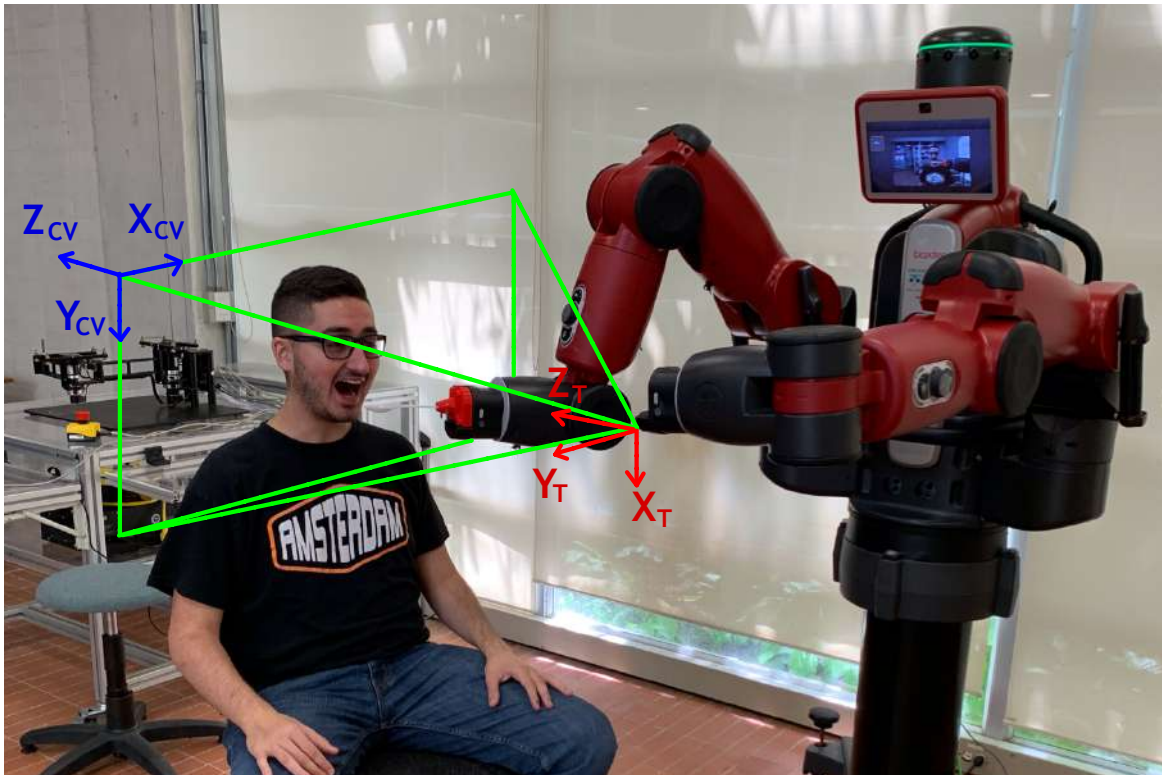


Figure 4.18: General mappings for transformation matrices from {T} to {CV} frames. Own development.

This procedure is a combination of the specific coordinates of Baxter camera (that change in time based on Baxter's left limb conditions), and the overall position of the user's mouth, that also changes in time based on the patient's current position.

To start analyzing how the user's mouth is detected (frame {F}), it is critical to look back into how the face detection algorithm works, giving an output of a rectangle that has its edges on the limits of the detected face. The user's mouth is inside that rectangle at its lowest part, and for Baxter Bon Appetit considerations, the specific position of the user's mouth will be at the lowest 1/3 of the vertical distance, and at the middle of the rectangle, as shown in fig[4.19]:

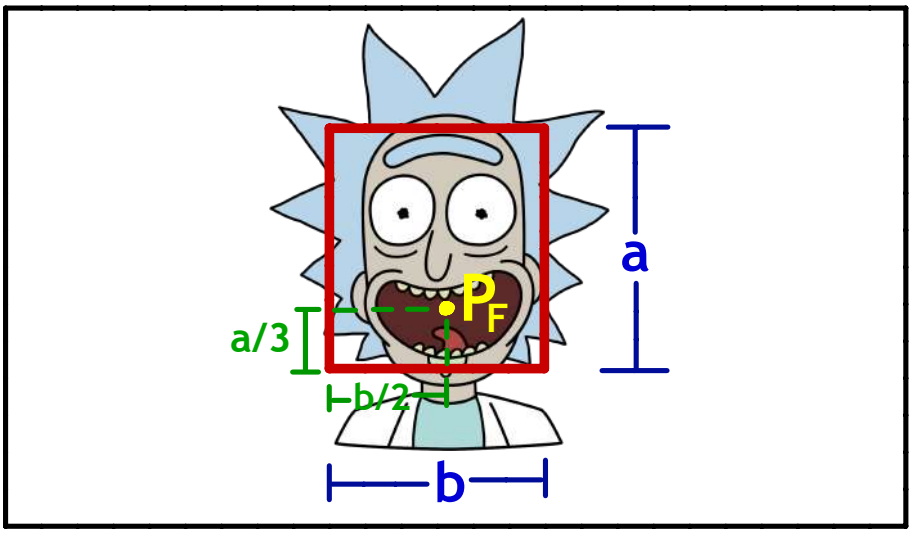


Figure 4.19: Position of user's mouth based on face detection rectangle output. Own development.

The next essential aspect of developing a solution where a computer vision algorithm is involved, is that there must be a correct mapping between pixels and meters (or the equivalent unit of measurement). This consideration had to be taken into account for the final transformation matrices of the user's mouth detection and the general process can be understood as follows:

- The relationship between pixels and meters must be achieved from a mathematical and a experimental way.
- The distance that represents the depth on where the image plane is located, is an independent variable that modifies the planar distances of the exposed plane.
- The values obtained from the mappings can only be applied for the implemented camera, because of the unique focal length of the lens of the device and, if the camera is replaced, tweaking factors must be added.

Now that these considerations are taken into account, the first step is to understand the underlying math that relate the geometrical space of the image planes and the physical distances that they represent.

Figures [4.20, 4.21], specify how the distances and planes are related based on the original center of projection for Baxter's left camera, and the corresponding generated planes (potentially infinite amount of them). The point {P} is where the camera is located and the planes $\{A_{cv}\}$ and $\{B_{cv}\}$ are example planes for possible parallel representations of where the user is located at any given time.

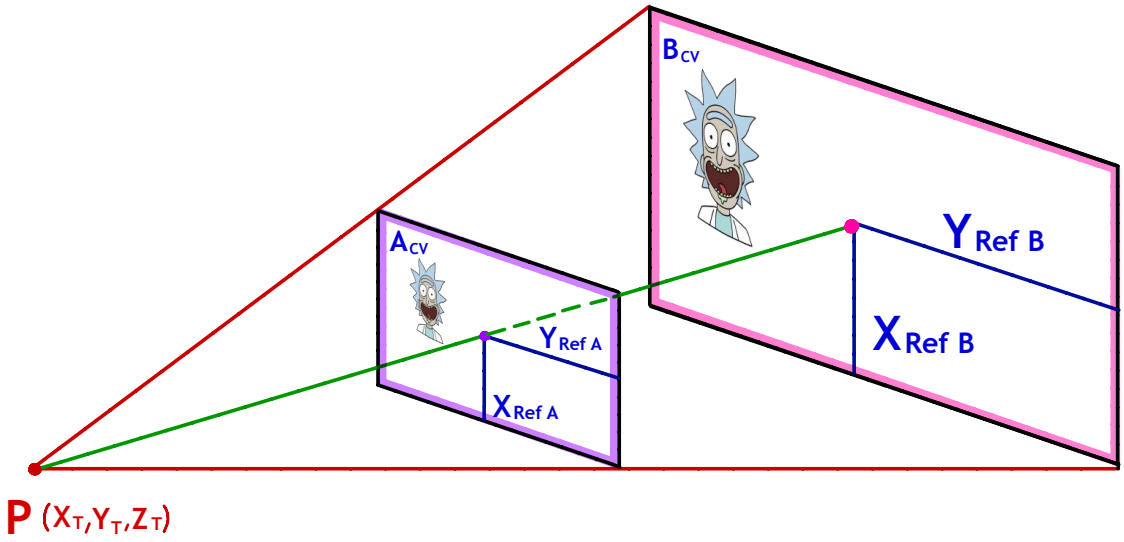


Figure 4.20: Detailed 3D mappings and frames from Baxter left camera to possible image planes where the user's face could be located. Own development.

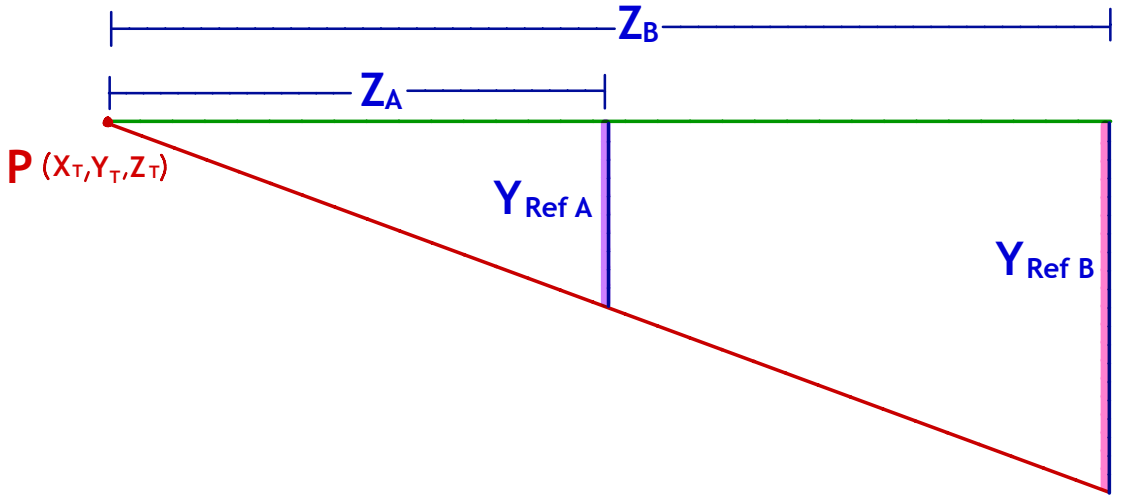


Figure 4.21: Detailed 2D mappings and frames from Baxter left camera to possible image planes where the user's face could be located. Own development.

These euclidean space configurations have strong-wise mathematical relationships that can be proven to be:

$$
\begin{aligned}
R &= \frac{Y_{ref_A}[m]}{Z_A[m]} = \frac{Y_{ref_B}[m]}{Z_B[m]} \\
S_X &= \frac{X_{ref_A}[m]}{X_{ref_A}[pix]} \\
S_Y &= \frac{Y_{ref_A}[m]}{Y_{ref_A}[pix]} \\
R &= \text{Constant ratio of space triangulation} \\
S_X &= \text{X scaling factor [m]/[pix]} \\
S_Y &= \text{Y scaling factor [m]/[pix]}
\end{aligned}
\tag{4.3}
$$

4.3: Geometrical space relationships for computer vision mappings.

The following steps after gathering these equations, is to understand that the "Z" distance will alter the constants R, S_x and S_Y in a direct way. For Baxter Bon Appetit project, the conceived "Z" distance, will be assumed to be a constant value and be decided from real-life experiments. This decision was taken based on the specifications of Baxter camera discussed on tab[3.3], where it can only obtain a 2D space representation and does not have any depth acquisition.

To obtain Z distance, the user was located on the best manoeuvrability areas for Baxter right limb, based on experimental movements of Baxter arm with the cuff sensor pressed (the one located at its wrist). The result of these movements, was to conceive the distance and to measure it from Baxter's left camera (a value that will be really important for the euclidean equations already discussed).

To complete the mathematical representations of eq[4.3] and find the relevant constants, the procedure was to do measurements of Z_A, Z_B, Y_{ref_A} and Y_{ref_B} in meters, and repeat it through multiple iterations, to obtain the desired results, as follows:

Figure 4.22: Experiments to characterize Baxter left limb camera with respect of the image plane of the user. Own development.

After obtaining these scaling constant factors for mapping values from pixels to meters, the next step is to understand the transformations and their relevant translations and rotations. Keep in mind that all of these transforms are denoted in meters.

To transform the reference frame {T} into {CV}, the compact mathematical representation is done with the transformation matrix as follows:

$$ {}^{T}_{CV}\mathrm{T} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & P_x \\ r_{21} & r_{22} & r_{23} & P_y \\ r_{31} & r_{32} & r_{33} & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & -H/2 \\ -1 & 0 & 0 & W/2 \\ 0 & 0 & 1 & Z_{const} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4} $$

4.4: Transformation matrix from frame {T} to frame {CV}.

Additionally, the illustration shown in fig[4.23] represents the image plane for Baxter's left camera, specified with the corresponding frames {T} (Baxter left limb camera reference), {CV} (Computer Vision reference), and {F} (Face mouth position). One essential consideration to keep in mind, is that the already explained computer vision algorithms (Viola-Jones and Face Recognition), return a dictionary with the coordinates of the rectangle that circumscribes the user's face.

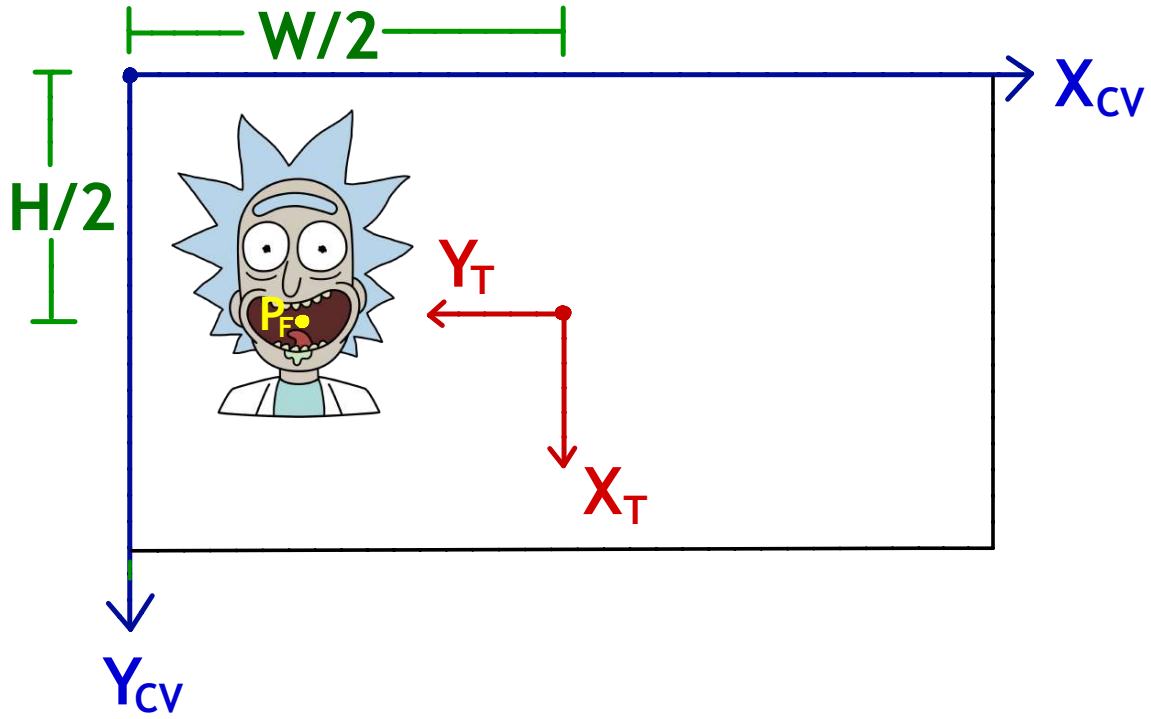


Figure 4.23: Relationship of frames {CV} and {T} in a plane for user's mouth detection. Own development.

Finally, the transformation matrix from the computer vision image plane {CV} into the specific face frame {F}, can be calculated with the following expression:

$$
{}^{T}_{CV}\mathrm{T} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & P_x \\ r_{21} & r_{22} & r_{23} & P_y \\ r_{31} & r_{32} & r_{33} & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & X_F \\ 0 & 1 & 0 & Y_F \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.5}
$$

4.5: Transformation matrix from frame {CV} to frame {F}.

The values specified on eq[4.5] (X_F and Y_F), are calculated based on the output of the face detection algorithm and the extra mappings explained in fig[4.19]. Notice that there are no rotations and the translation in the "Z" axis is already done in the previous transformation matrix to get to {CV}.

All the mathematical representations and the programmatic development on Python to transform these expressions into actual results, can be found with details in Baxter Bon Appetit GitHub repository:

- Baxter-Bon-Appetit Computer Vision Mappings (Source Code)

Chapter 5

Control Theory

A Robotic system is an multidisciplinary device that is designed on top of the pillars of engineering and computer science. The key goal of robotics is to create and supply, programmable machines, that can do different tasks with greater speed, safety and precision (Sharma, 2020). To properly obtain these benefits, robotic systems must be integrated with different control strategies that allow the system to perform different actions based on a desired conditions.

Baxter Bon Appetit project involves areas of knowledge such as robotics, computer vision, control theory and software development. The core component of this research is the development of a robust control system that aims to enable the active feeding of patients with motor disabilities on their upper limbs. To achieve this task, the system must be able to dynamically understand where the patient and the end-effector are located so it can move its arm towards the user's mouth.

In this chapter, it will be discussed the development, design and implementation of different control strategies that fulfill the requirements for actively feeding a patient.

5.1 Understanding the Control Problem

Every solution that uses a control strategy, must understand the inputs and outputs for a given dynamic system. Baxter Bon Appetit project has main actors that change dynamically in time: The user and the robot's arm.

The challenge that has to be solved in this research is to feed a patient by using one of the two arms of Baxter robot, based on the mouth's position of the patient.

Figure 5.1: Main actors that change in time during the feeding action.

This movement must be achieved satisfying the following conditions:

1. The orientation of the end-effector of Baxter robot must not change during the movement, this is an important statement because the robot must carry out food (solids and liquids) and they can not be spilled at any time of the trajectory.
2. The system must be able to track the current position of the user's mouth and move dynamically its arm towards it.
3. The system must reach the mouth's position in a smooth way, in order to perform a movement similar to the one executed by a human being performing this task.
4. The system must be able to perform the movement of the robot's arm correctly even with external perturbations.

5.2 PID Control Strategy

The PID controllers (Proportional-integral-derivative), is one of the most popular and common predefined control structures in real life applications.Their dynamics are known for decades and some of the great advantages of PID controllers are:

1. Its feasibility that allows an easy implementation.
2. Its well-known structure that enables a simple tuning approach for the control system.
3. Its effectiveness for SISO (Single Input - Single Output) systems.

5.2.1 Baxter joint controllers

By default, Baxter robot comes with a PID controller for each one of its joint articulations, implying that the overall behavior or Baxter's arm is defined by the consequence of the dynamics of each joint controller.

Figure 5.2: Baxter's internal joints structure. Taken from (Rethink Robotics, 2015a)

The default control structure for each joint of Baxter robot is a PID controller with the following expression:

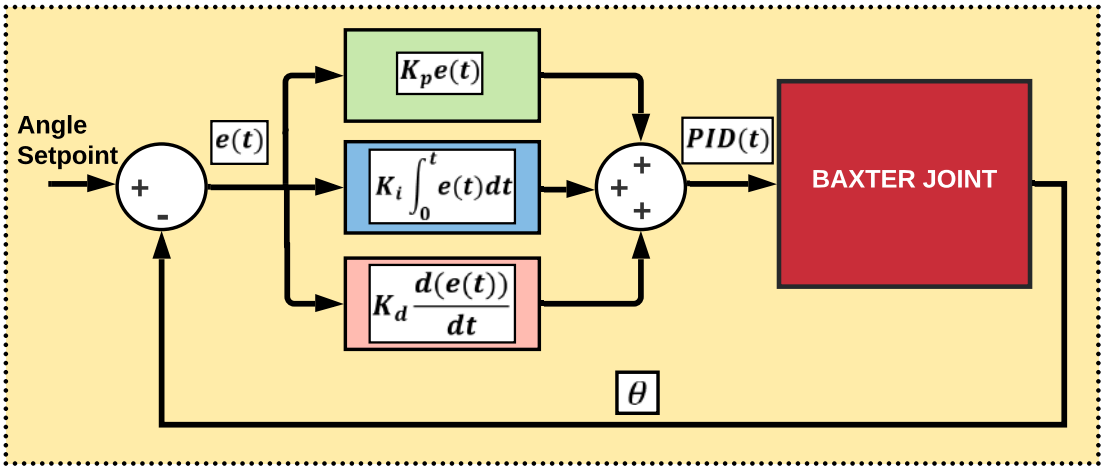


Figure 5.3: Block diagram of Baxter's joints PID controllers. Own development.

$$\text{PID}(k) = K_p \cdot e(k) + K_i(e(k) \cdot T_s + A_{sum}(k-1)) + K_d \cdot \left[\frac{e(k) - e(k-1)}{T_s}\right] \tag{5.1}$$
$$A_{sum}(k) = A_{sum}(k-1) + e(k) \cdot Ts$$

5.1: Baxter's joints PID control structure.

Where the $e(k)$ is the current error signal calculated from the difference between a reference angle value (radians) and the current measured value from the joint encoders angles. T_s correspond to the sample time in which the discrete controller takes the measurement and applies the control actions.

It is important to notice that this structure applies similarly to each one of the 7 degrees of freedom of Baxter robot.

The PID algorithms will be implemented using Baxter Interface SDK. These can be found at the Rethink Robotics' repository.

- Rethink Robotics baxter_interface PID controller implementation

5.2.2 Mouth tracking problem

Now that it is possible to apply a control signal to move each baxter joint to a desired angle, the question that must be answered to solve the mouth tracking problem exposed in this project, is how to find the right joint commands in order to move the end-effector to the user's mouth location.

To achieve this solution, an intermediate process must be implemented in which the current user's position is mapped from Cartesian coordinates to joint coordinates. This

analysis can be implemented using the inverse pose kinematics of Baxter robot, as seen in sec[1.3.2.1.3].

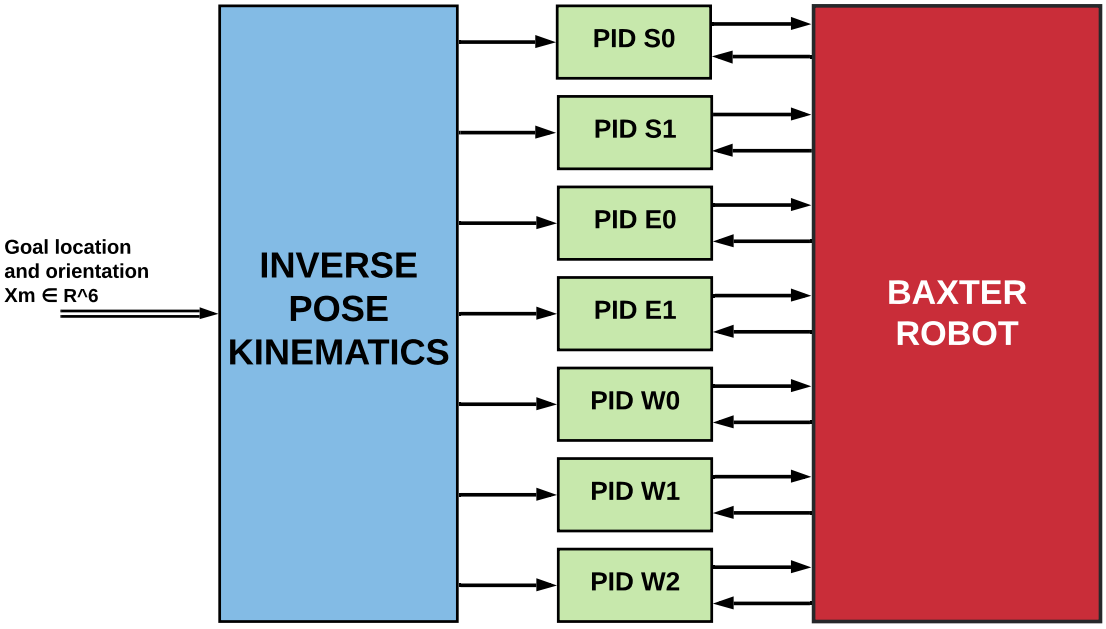


Figure 5.4: Mouth tracking solution using IPK and PID controllers. Own development.

For Baxter Bon Appetit, the source code for this control strategy can be found at the GitHub repository:

- Baxter Bon Appetit mouth tracking approach based on PID and IPK algorithms

5.2.3 PID results

To analyze how is the behavior of the system with the control structure shown in fig[5.4], a experiment was designed in which the main task was to move Baxter's right arm to a specific constant location in space, so that the step response of the overall system could be analysed afterwards. The results are the following:

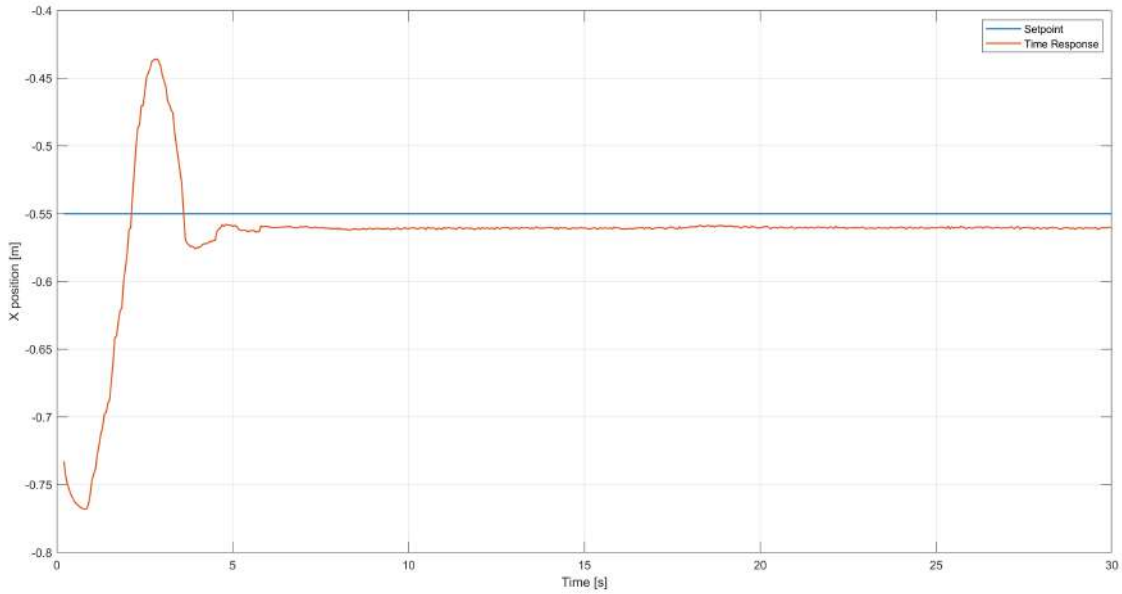


Figure 5.5: Baxter's end-effector x position during the experiment with the mouth tracking solution using IPK and PID.

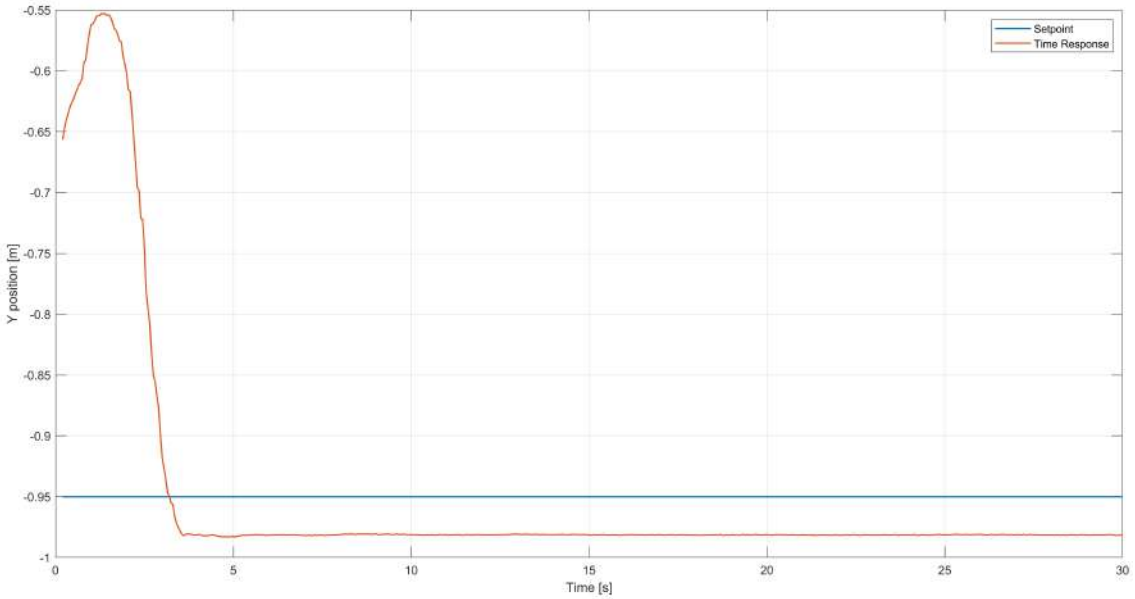


Figure 5.6: Baxter's end-effector y position during the experiment with the mouth tracking solution using IPK and PID.

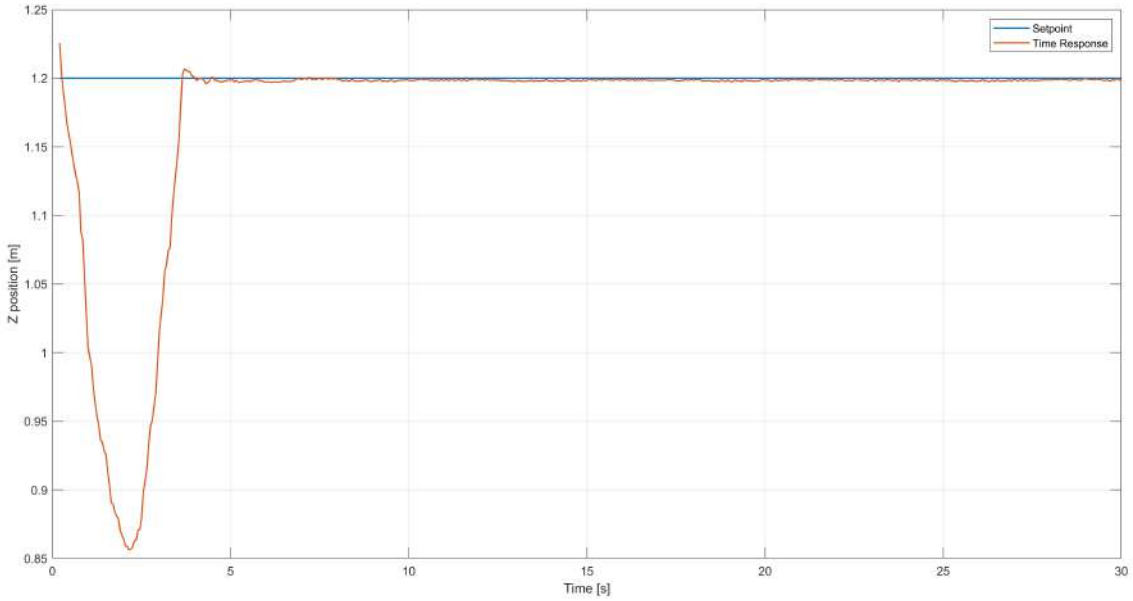


Figure 5.7: Baxter's end-effector z position during the experiment with the mouth tracking solution using IPK and PID.

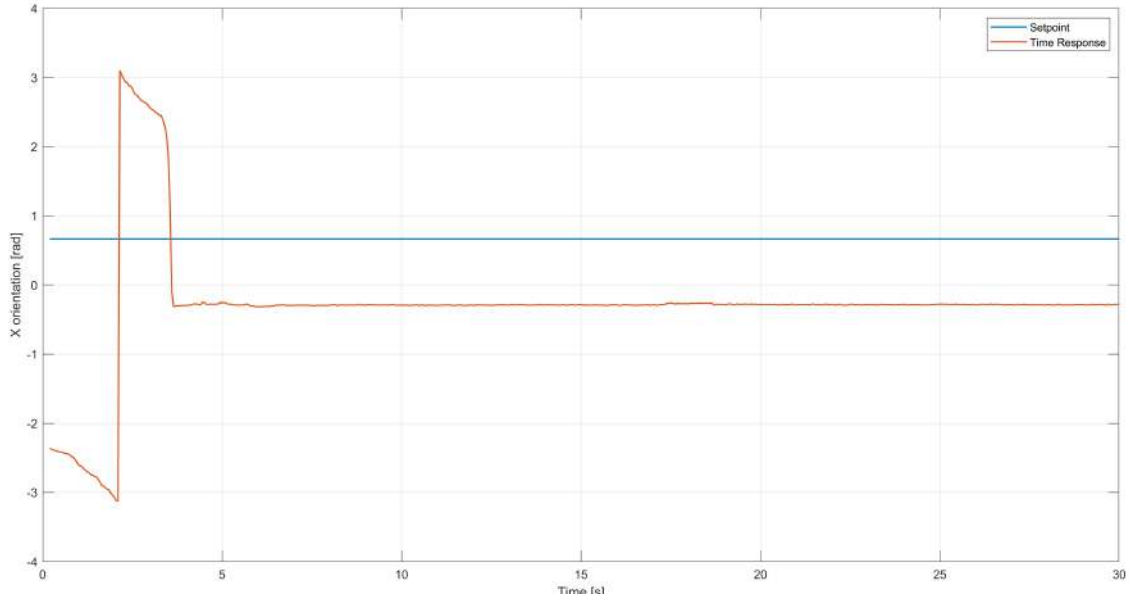


Figure 5.8: Baxter's end-effector x orientation during the experiment with the mouth tracking solution using IPK and PID.

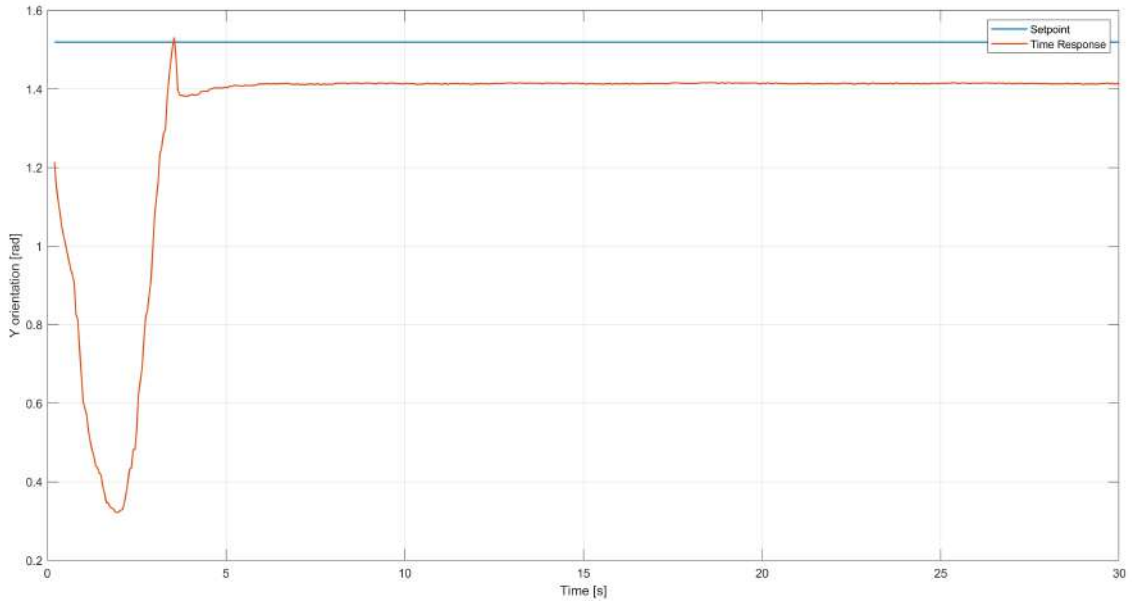


Figure 5.9: Baxter's end-effector y orientation during the experiment with the mouth tracking solution using IPK and PID.

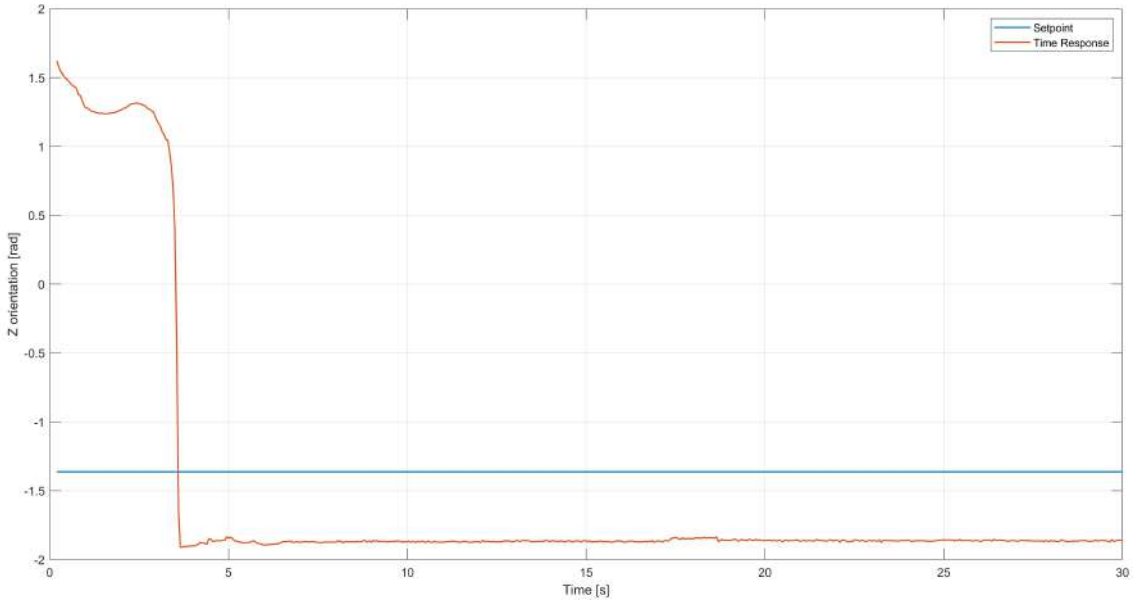


Figure 5.10: Baxter's end-effector z orientation during the experiment with the mouth tracking solution using IPK and PID.

After the results of the experiment were analyzed, some interesting insights were found:

- As seen in figs[5.5, 5.6], the time response of the system has a significant steady state error due to the mathematical simplifications of the 6 DOF IPK, explained in sec[3.3.2]

- The Cartesian orientation has relevant changes and even high-slope segments that are undesired for the feeding task.

5.3 MPC Control Strategy

In its beginnings the MPC (Model Predictive Control) was designed to be implemented in applications where the control of chemical processes was required (C et al., 1989). This control strategy was also widely used for the control of aerial vehicles due to its high performance (J et al., 2002). In recent years, the MPC has been strongly introduced and implemented in the vast world of robotics, with this novel control strategy, problems such as trajectory planning, compensation of the inertial forces of a manipulator and the approach in a three-dimensional space to a target, have been able to find solutions with a more efficient and optimal point of view.

Baxter Bon Appetit MPC strategy is inspired from two contemporary robotic researches that integrated successfully modern control solutions with real-life problems (Park et al., 2020) (Jain et al., 2013).
The controller to be developed for the Baxter Bon Appetit project uses a linear MPC, with a prediction and control horizon which will be chosen in the tuning stage. This controller is based on the discrete-time model:

$$x(k+1) = Ax(k) + Bu(k) \tag{5.2}$$

5.2: General discrete-time model of the robot.

Where $x(k)$ are the states of the system and $u(k)$ are the control inputs.

This control strategy calculates a series of control inputs every time-step k, having as main goal to minimize an objective function which depends on the states and inputs of the system, this optimization problem must consider a series of constraints which map the physical limitations of the robot and its degrees of freedom. N and M are the prediction and control horizon of the model predictive controller respectively. This control strategy defines a quadratic algorithm, that is solved based on a convex optimization problem at each sample time k based on N prediction steps in the future. After that, it saves and applies M calculated control inputs in the system and reformulates the quadratic problem after the already calculated control inputs are applied.

The MPC algorithms and conditions for Baxter Bon Appetit research can be found at the GitHub repository:

- Baxter Bon Appetit MPC algorithms
- Baxter Bon Appetit MPC implementation

5.3.1 Overview of the one-step model predictive controller

As mentioned above, the model predictive controller implemented in Baxter Bon Appetit project aims to move the robot's end-effector to a position close to the patient's mouth, considering different physical constraints of the manipulator.

This controller has the following important parameters:

- Goal location and orientation ($x_m \in \mathbb{R}^6$): This is a vector with the desired Cartesian position and orientation that represents where and how the controller attempts to move the end-effector of Baxter robot. The position depends on the user's mouth location.
- Optimal control inputs ($\Delta\phi \in \mathbb{R}^7$): This a vector with the solution of the quadratic problem, these are the optimal values that the Model Predictive controller calculated given the goal position and orientation parameters.
- Robot joint commands ($\phi \in \mathbb{R}^7$): This is a vector with the direct joint commands that the robot is going to receive, these values can be understood as the control signal of the controller.
- Robot states ($\theta \in \mathbb{R}^7$): This is a vector with the current values of the joints of the robot. These values are given by the robot's encoders.

5.3.2 Control structure

The figure fig[5.11] shows the block diagram of the model predictive controller and its structure. This MPC implementation runs in Python programming language on a conventional desktop computer at a control frequency of approximately 100 Hertz.

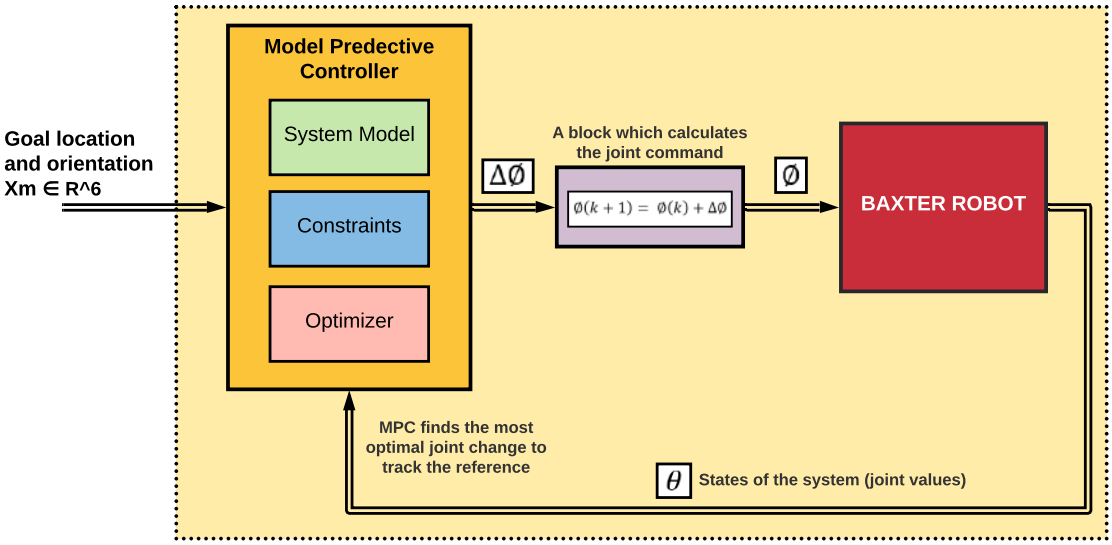


Figure 5.11: Block diagram of the Model Predictive Controller to implement in Baxter Bon Appetit. Own development.

The setpoint of this model predictive controller is a desired Cartesian position and orientation represented mathematically by means of a 6-component vector, where, the first three components represent the Cartesian coordinates of the x, y and z positions, and the others, represent the Cartesian orientation in absolute angles representation (Craig, 2004). This controller uses the measurements of the articulations given by the encoders of the robot as feedback to perform the dynamic control of the system.

5.3.3 Linear discrete-time model

One of the most important elements for the correct design and implementation of a Model Predictive Controller is the system model or "digital copy" of the robot. Based on this model, the controller can somehow predict the future behavior of the real system.

For this specific application, mathematical model of the system is given by:

$$\theta(k+1) = A\theta(k) + B\Delta\phi(k) \tag{5.3}$$

5.3: Discrete-time model of Baxter robot's arms.

One important remark from eq[5.3], is that matrices A and B are considered to be identity matrices, due to the fact that the changes in the articulations of Baxter robot are considered to be homogeneous based on the inputs. Additionally to this simplification, it is valuable to keep in mind that this is a linear model and the real behavior could have slight non-linear performances.

5.3.4 Quadratic program to compute $\Delta\phi$

A quadratic optimization problem is a mathematical expression in which the goal is to find the best possible and feasible solution based on some given criteria (ByungSoo Ko, 2017).

For the Model Predictive Controller of Baxter Bon Appetit project, the optimization problem is related to the eq[5.3], in which $\Delta\phi$ is the optimization variable that governs the behavior of the system. In order to find $\Delta\phi$ a quadratic objective function must be minimized with convex optimization algorithms. For this research, the open source cvxpy library will be implemented to solve this mathematical problems:

- cvxpy: A Python-embedded modeling language for convex optimization problems

The objective function has a general structure as follows:

$$\sum_i \sigma_i q_i \tag{5.4}$$

5.4: MPC objective function structure.

In eq[5.4], q_i represents a series of quadratic functions that depends on the optimization variable $\Delta\phi$ and σ_i are tweaked experimental constants for the weights of the quadratic equation to solve. For Baxter Bon Appetit, σ_i will be considered as one. The structure of q_i will be discussed below.

5.3.4.1 Move to the goal position

The first component of the quadratic objective function described on eq[5.4], has as main goal to reach the robot's arm to a goal position (user's mouth location). This expression has the following structure:

$$q = ||\Delta x_d - \Delta x_h||^2 \tag{5.5}$$

5.5: MPC objective function structure.

Where it is important to notice that $\Delta x_h = x_h(k+1) - x_h(k)$, is the predicted position and orientation of Baxter's end-effector and Δx_d is the desired delta expected after one sample time of those values.

For this implementation it will be assumed that the changes of the robot's joint angles will be small enough to satisfy the equation:

$$\Delta x_h = J_h \Delta\theta \tag{5.6}$$

5.6: Relationship between the Cartesian space and joint space.

The jacobian matrix shown on eq[5.6], is the one discussed on sec[3.3.3], which is Baxter's jacobian of the end-effector, where $J_h \in \mathbb{R}^{6\times7}$. The term $\Delta\theta = \theta(k+1) - \theta(k)$ corresponds to the change in the joint angles predicted by the eq[5.3].

The objective function for Baxter Bon Appetit Model Predictive Controller will be expressed as follows:

$$q = ||\Delta x_d - J_h \Delta\theta||^2 \tag{5.7}$$

5.7: Objective function of the Model Predictive Controller.

For this research, an intermediate controller will provide the unit vector a distance in which the end-effect will move towards the goal in a straight line. This controller enables and guarantees the small changes in position and orientation, as well as the safety of the patient when the arm is reaching his mouth.

For Baxter Bon Appetit, this process is called Cartesian increment and it is defined as follows:

$$\Delta x_d = \begin{cases} d_c \frac{x_g - xh}{||x_g - x_h||} & if||x_g - x_h|| > d_c \\ x_g - x_h & if||x_g - x_h|| \leq d_c \end{cases} \tag{5.8}$$

5.8: Simple-wise function for the Cartesian increments.

Where x_g is a vector with the goal position and orientation that the robot must reach, and d_c is a small constant value that indicates the magnitude of change of the position and orientation towards the goal.

5.3.4.2 Joint limits

One of the greatest advantages of MPC controllers is the possibility of adding custom constraints for the variables of the system model, this ability allows the controller to predict scenarios that don't exceed the physical limitations of the given variables. For Baxter Bon Appetit project, these constraints are going to be essential for maintaining each joint angle inside the lower and upper limit shown in tab[3.1].

These constraints can be expressed mathematically as follows:

$$\theta_{min} \leq \theta(k+1) \leq \theta_{max} \tag{5.9}$$

5.9: MPC constraints for joints physical limitations.

Where θ_{min} and θ_{max} are the vectors that correspond to the minimum and maximum values for each one of the joints of Baxter robot respectively

5.3.5 MPC Tuning

As exposed in the MPC expressions, there are some variables that have a direct impact on the behavior of the system, such as the prediction horizon (N) and control horizon (M). For this research, these variables will be found in an experimental way.

In order to find the best possible values of N and M, multiple experiments were developed. The main task of those tests was to move Baxter's right arm to a specific constant location in space, so that the step response of the overall system could be analysed afterwards. In this experiments, different combinations of these parameters were applied to the MPC controller, with the objective of finding the controller with the best dynamic performance.

The experiments were developed with the following combinations of N and M:

Experiment Nº	N	M
1	1	1
2	3	1
3	5	1
4	10	1
5	3	3
6	5	3
7	10	3
8	5	5
9	10	5
10	10	10

Table 5.1: Combinations of N and M for MPC tests.

During the experiments, the main variables of the system (Joints of the robot and the Cartesian location and orientation of the end-effector) were monitored constantly. The collected data of the Cartesian coordinates taken at the experiments will be shown in the following plots.

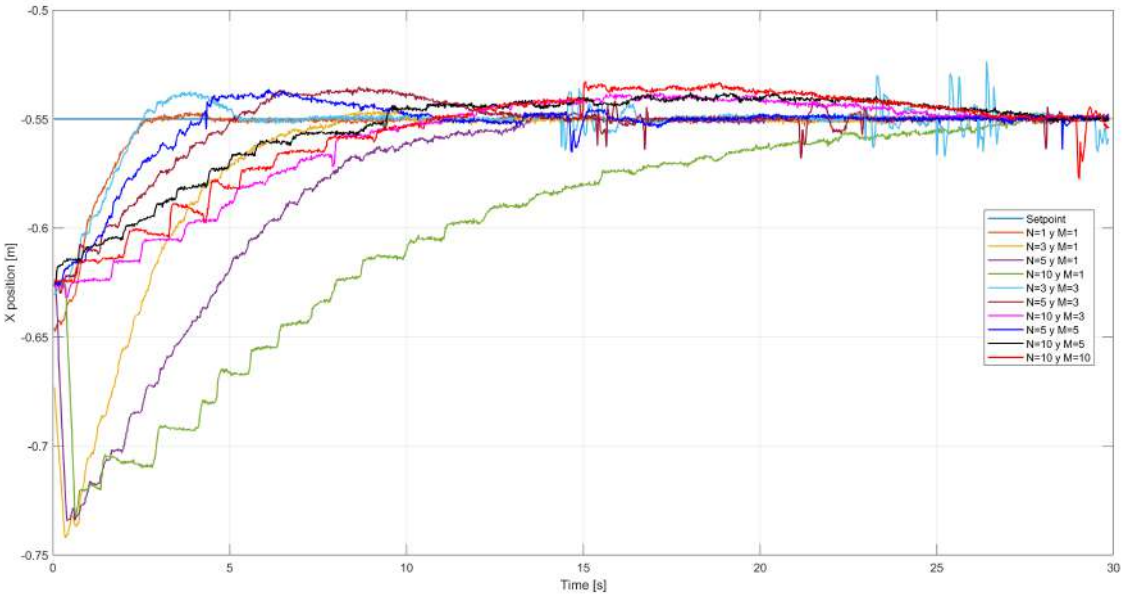


Figure 5.12: Baxter's end-effector x position during the MPC experiments.

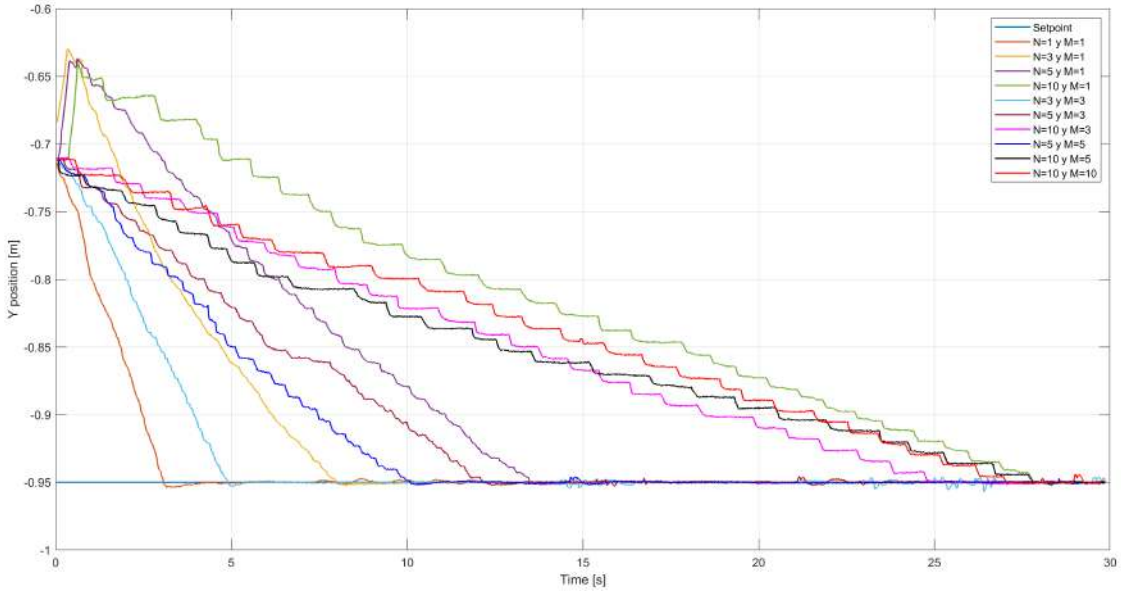


Figure 5.13: Baxter's end-effector y position during the MPC experiments.

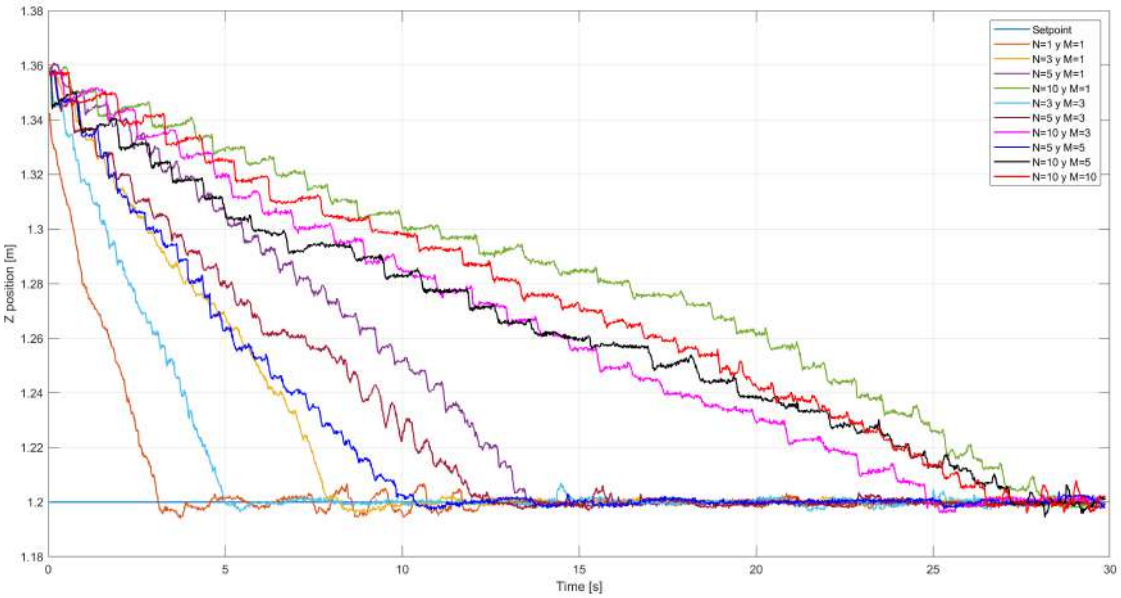


Figure 5.14: Baxter's end-effector z position during the MPC experiments.

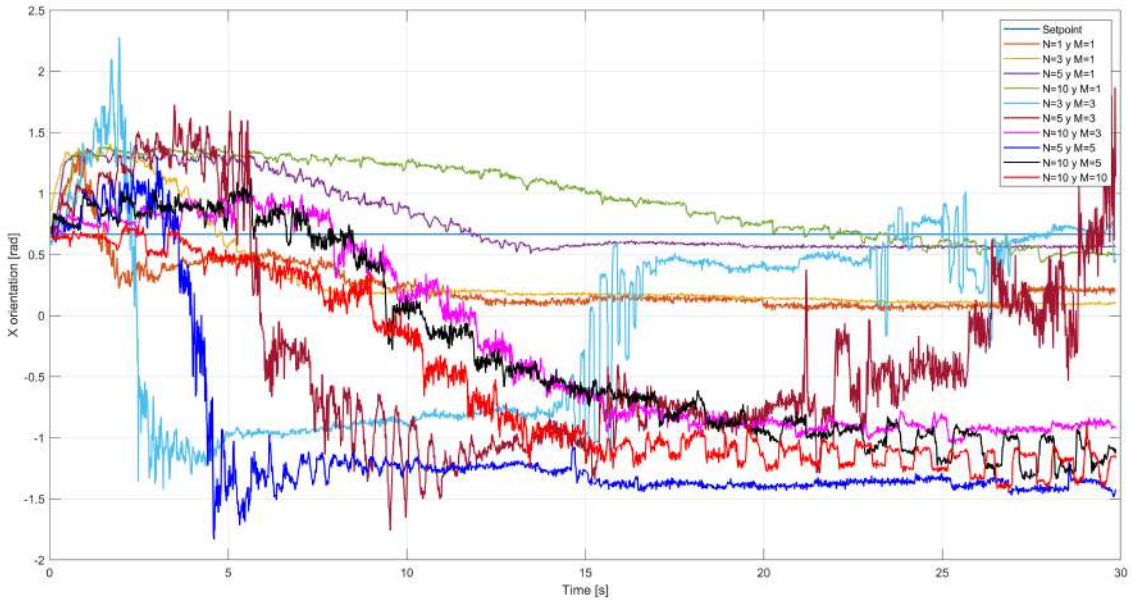


Figure 5.15: Baxter's end-effector x orientation during the MPC experiments.

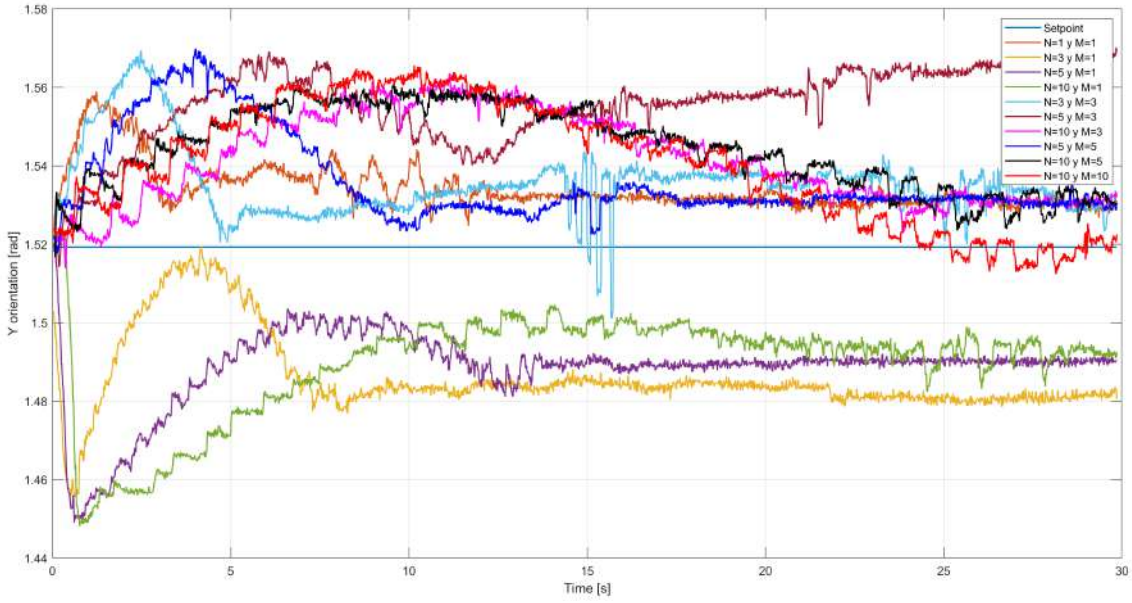


Figure 5.16: Baxter's end-effector y orientation during the MPC experiments.

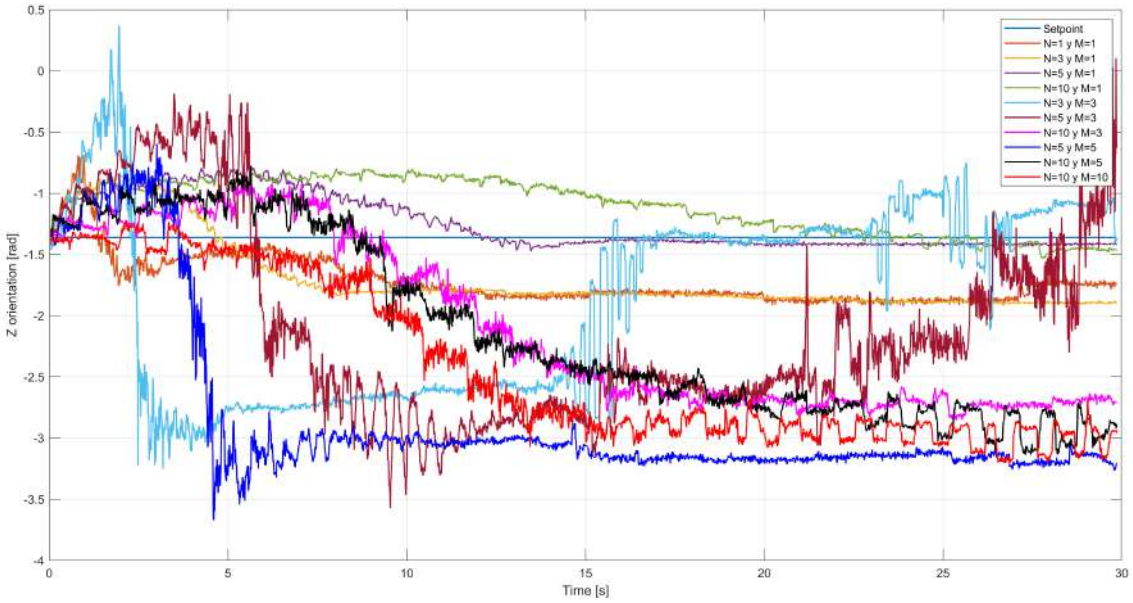


Figure 5.17: Baxter's end-effector z orientation during the MPC experiments.

After analysing all the possible configurations of the MPC controller implemented in the experiments, the best performance of the system was achieved with the values of $N = 1$ and $M = 1$. Some of the reasons for selecting this combination of the MPC parameters are:

- The rise time and the settling time were smallest ones compared to the other combinations.
- Time response has the smoothest behavior, making the movements of the end-effector optimal accurate for the task of feeding.
- As seen in figs[5.15, 5.16, 5.17] this controller has the steadiest behavior for the end-effector's orientation.

Further more, the experiments revealed some additional valuable information about the MPC. The step-wise behavior seen in some combinations such as the one that involves $N = 10$ and $M = 1$, indicate that computational resources are close to their performance limits and due to this, the horizontal parts of the step-wise response are a consequence of the processing time needed to compute these algorithms.

5.4 PID vs MPC Strategies Results Discussion

For Baxter Bon Appetit project, it is important to choose the best control strategy based on tangible conclusions that aim to execute the feeding task as safe and smooth as possible.

The previous sections showed the process of development of two control strategies (IPK and PID based controller and MPC controller). Although both approaches had great general results, it is crucial to distinguish which one satisfies better the project requirements.

To validate the best controller, the finest MPC controller (N = 1, M = 1) and the IPK-PID based controller were plotted in the same graph to get a better insight of the comparison of their behavior. The results were the following:

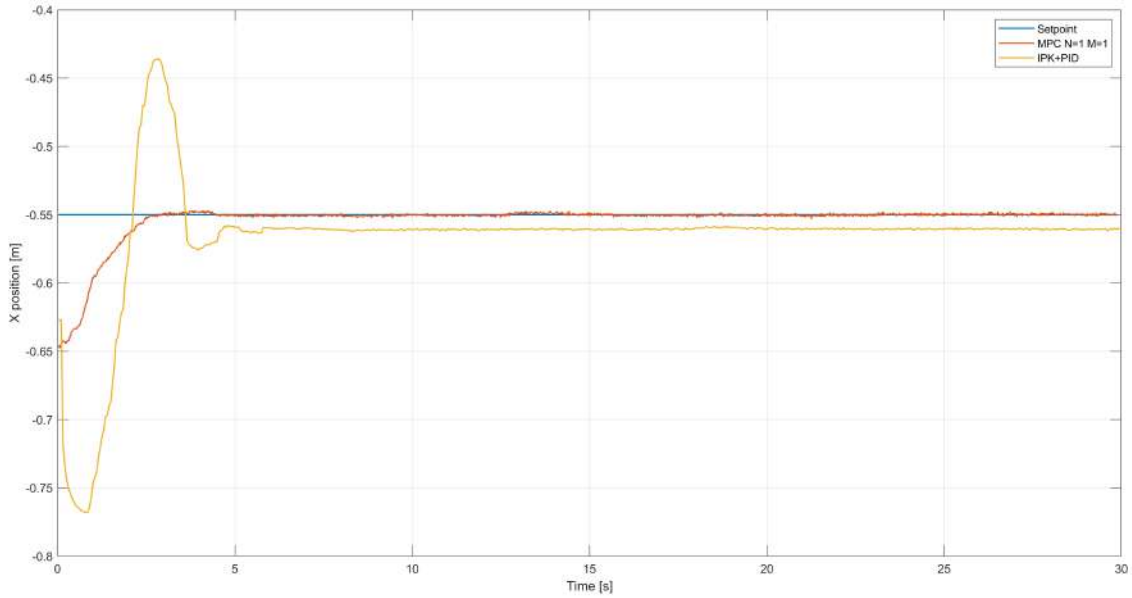


Figure 5.18: Baxter's end-effector x position for MPC and IPK-PID controllers.

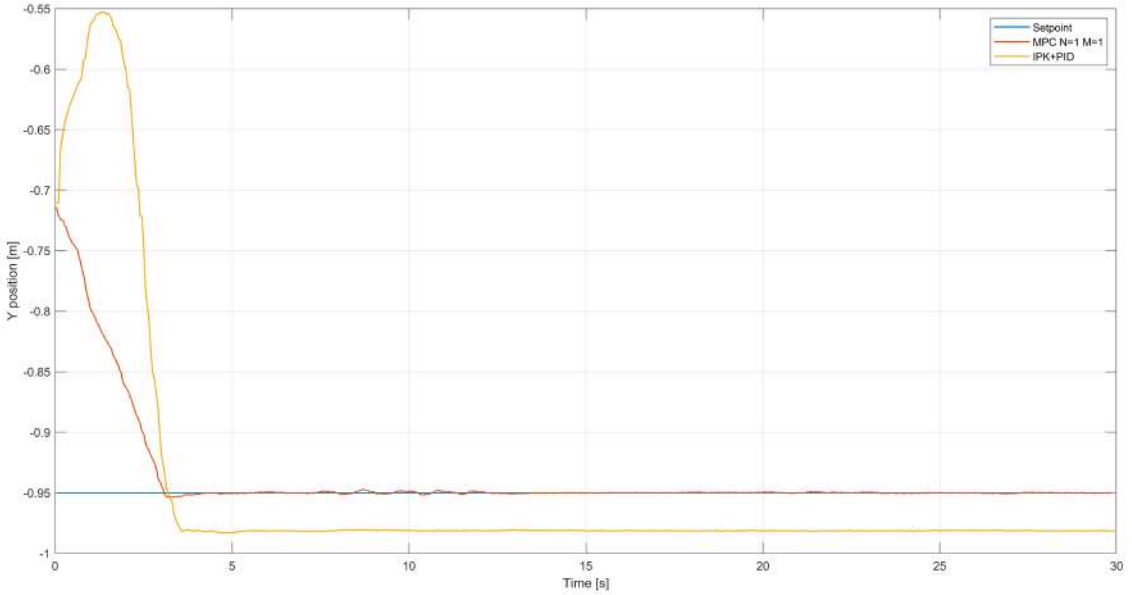


Figure 5.19: Baxter's end-effector y position for MPC and IPK-PID controllers.

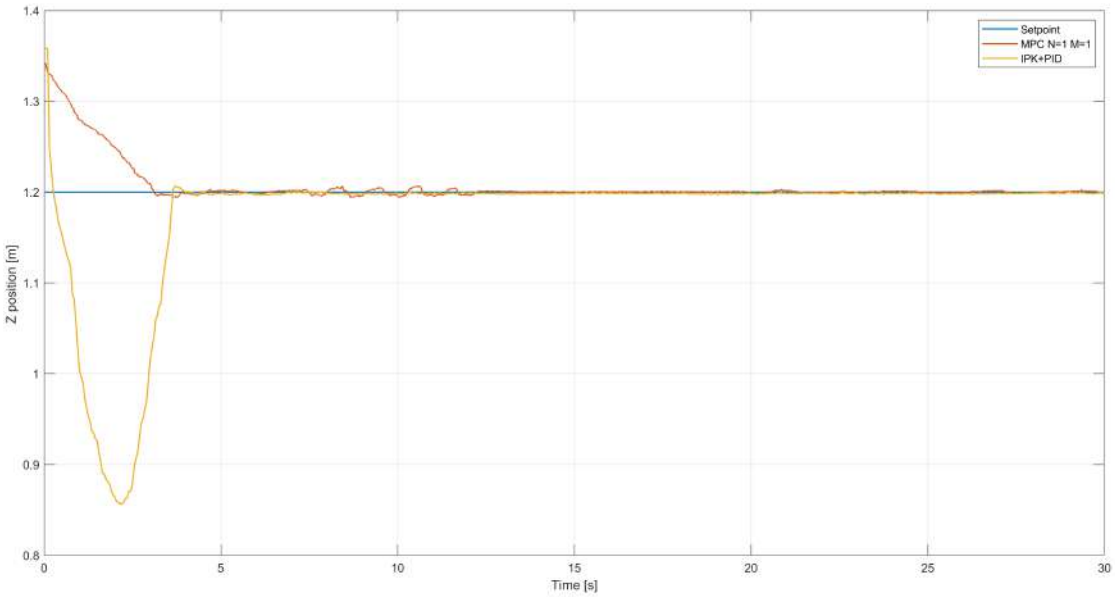


Figure 5.20: Baxter's end-effector z position for MPC and IPK-PID controllers.

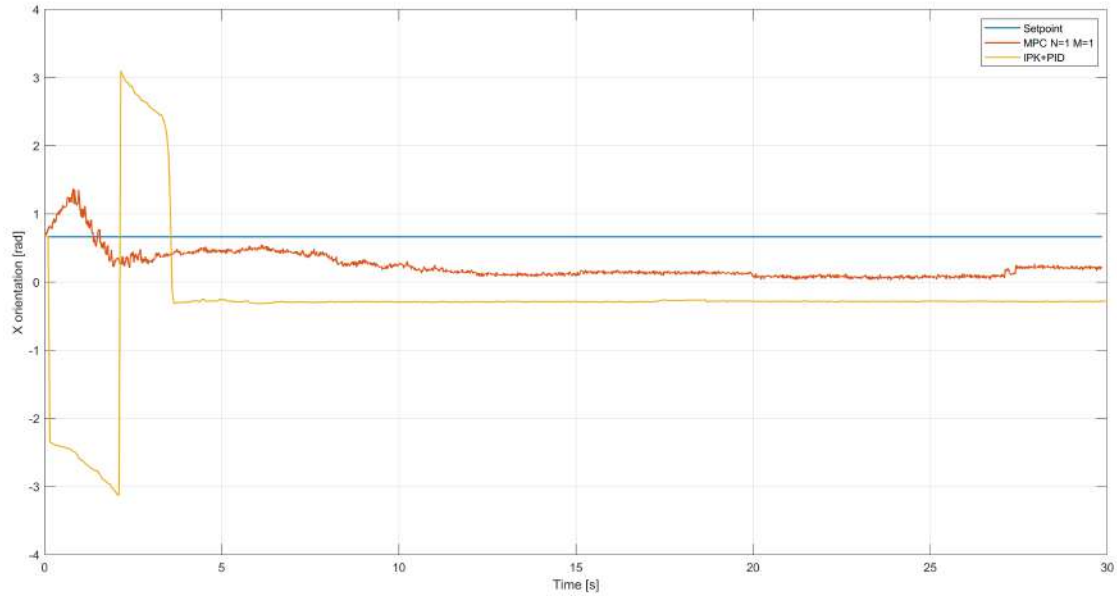


Figure 5.21: Baxter's end-effector x orientation for MPC and IPK-PID controllers.

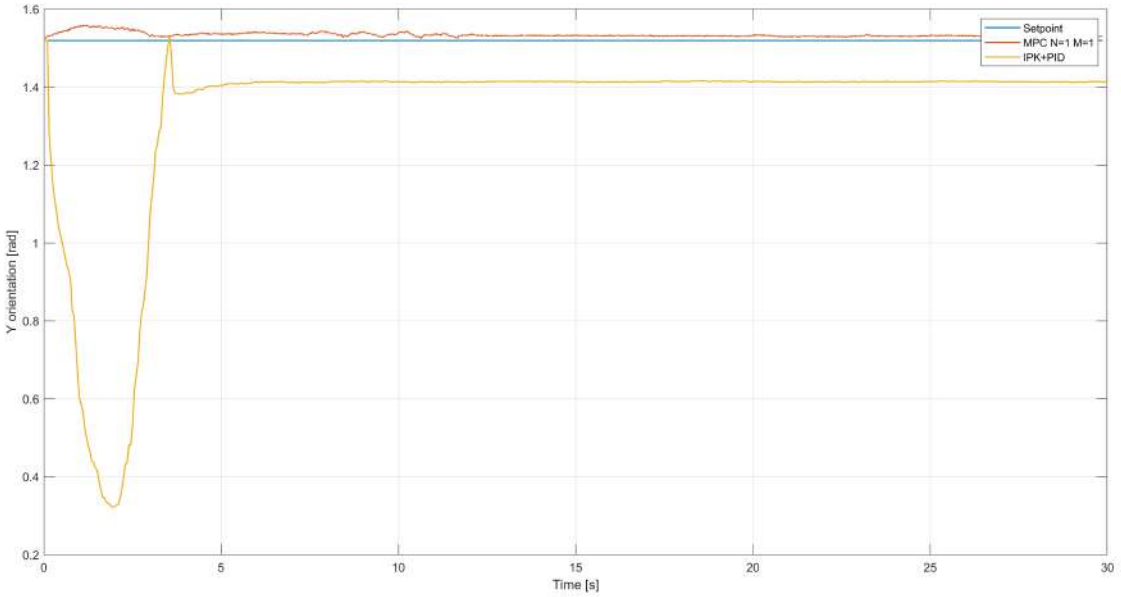


Figure 5.22: Baxter's end-effector y orientation for MPC and IPK-PID controllers.

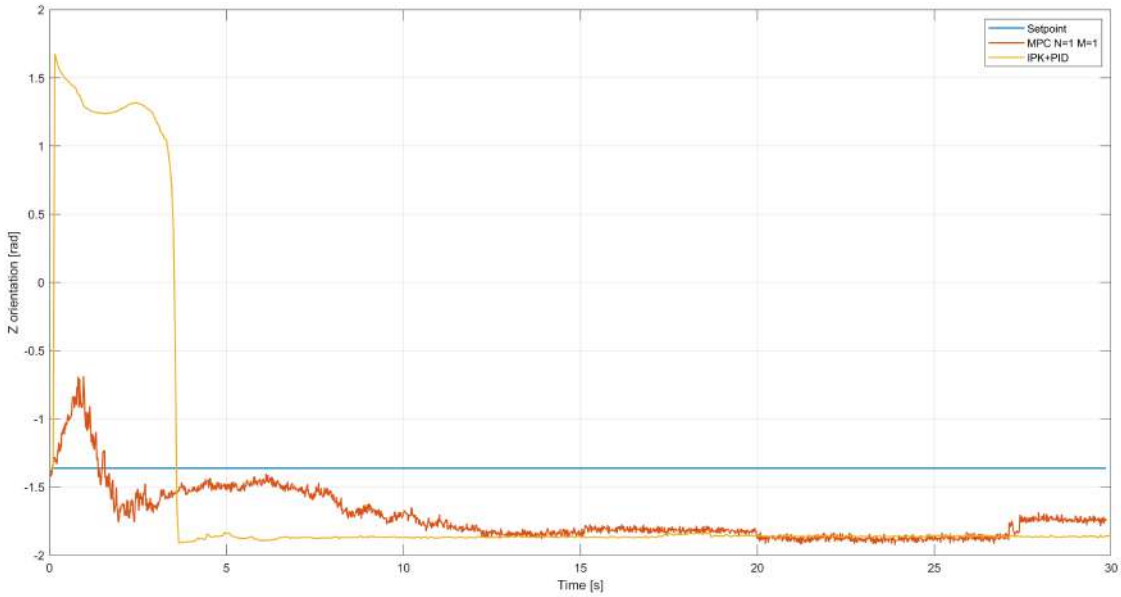


Figure 5.23: Baxter's end-effector z orientation for MPC and IPK-PID controllers.

As it can be inferred in figs[5.18, 5.19, 5.20], the best performance for the absolute Cartesian position was achieved with the MPC control strategy, because it had the smoothest time response and an smaller steady state error. It can be notice that IPK-PID control strategy has high-slope changes that can affect directly the acceleration and jerk of the robot's end-effector.

It is crucial to understand that one of the most important requirements for the feeding task is to maintain the end-effector's orientation constant during the robot's movements.

This condition is a complex challenge due to the high related dynamic behavior of the position and orientation of the robot and this can impact the other one. As is can be observed in figs[5.21, 5.22, 5.23], the MPC strategy can regulate better the end-effector's orientation during the movement of the robot, which fulfills that the robot gives correctly the food to the patient without dropping it during the feeding task.

Based on these analyses, the selected control strategy that is going to be implemented for Baxter Bon Appetit project is the MPC controller with $N = 1$ and $M = 1$ parameters.

5.5 MPC performance analysis

The two most important features that a controller must guarantee are trajectory tracking and setpoint regulation. Trajectory tracking is concerned with the design of control laws that force a system to reach and follow a time parameterized reference (Aguiar and Hespanha, 2007). Setpoint regulation is the control condition that aims to maintain a dynamic variable of a system on a reference value.

To evaluate the behavior of the MPC strategy, it is necessary to develop some tests that allow to understand how the system would perform under specific conditions. The tests were developed for the tracking and the regulation tasks.

5.5.1 Tracking task

The tracking task experiment consisted of executing the MPC strategy while the user moved to different locations. The goal of this test was to determine how effective the control strategy is under trajectory tracking circumstances.

The user's movement for the tracking test can be seen on fig[5.24].

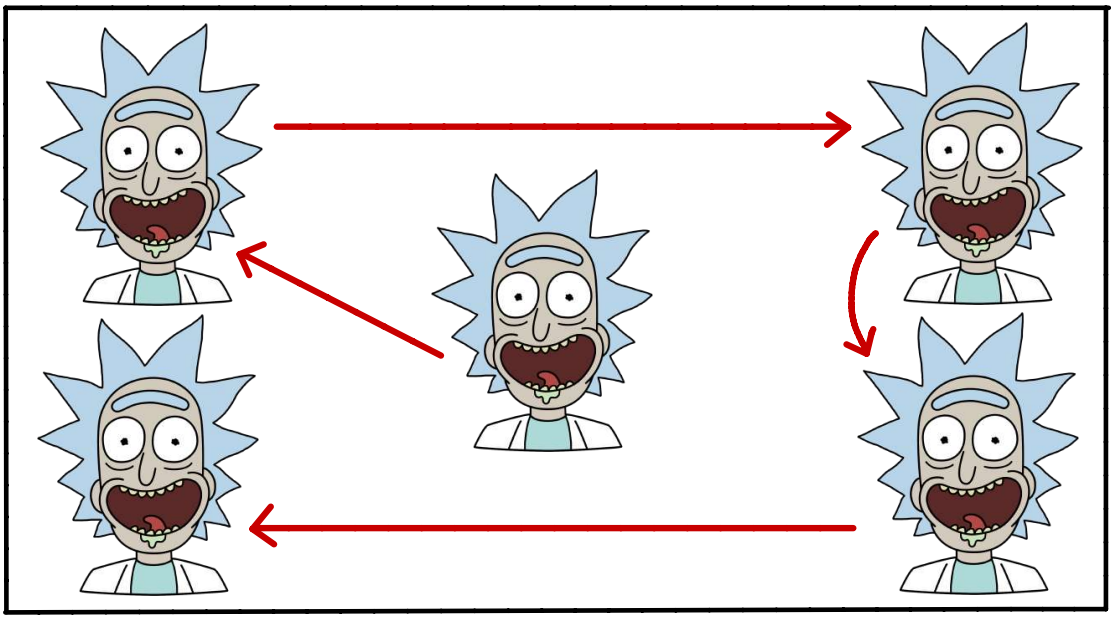


Figure 5.24: Illustration of patient's movement for mouth tracking experiment. Own development.

The results of the tracking experiment are going to be shown in the following figures:

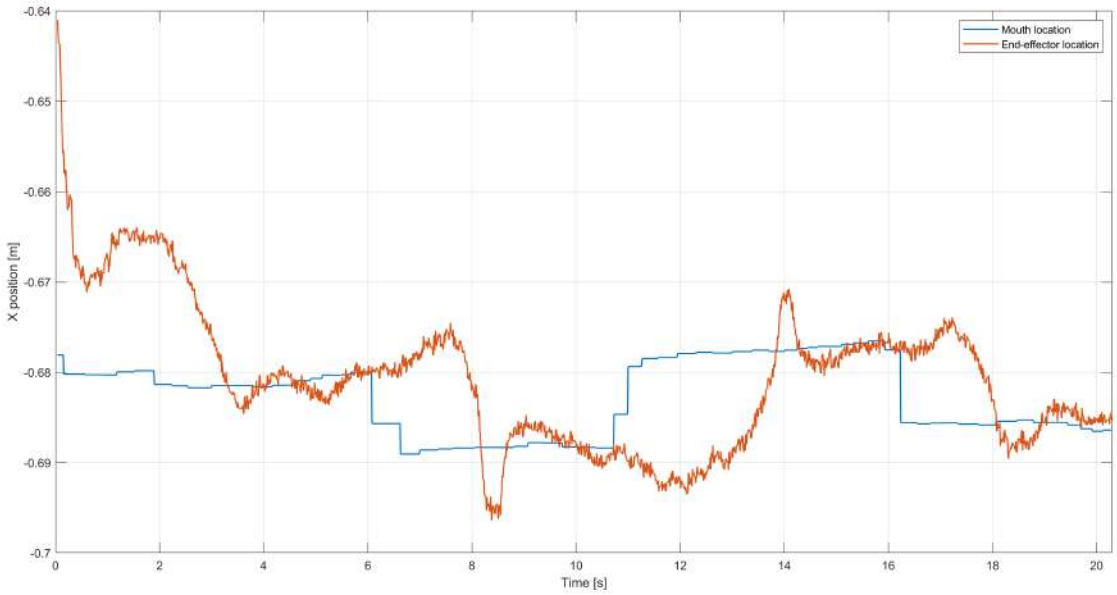


Figure 5.25: Baxter's end-effector x position during the mouth tracking test.

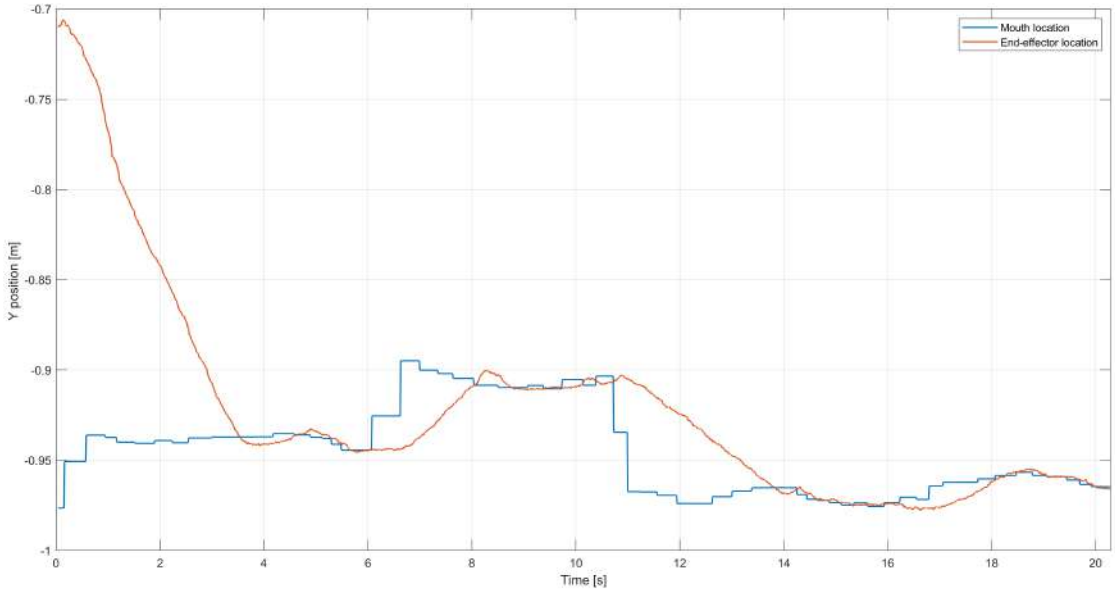


Figure 5.26: Baxter's end-effector y position during the mouth tracking test.

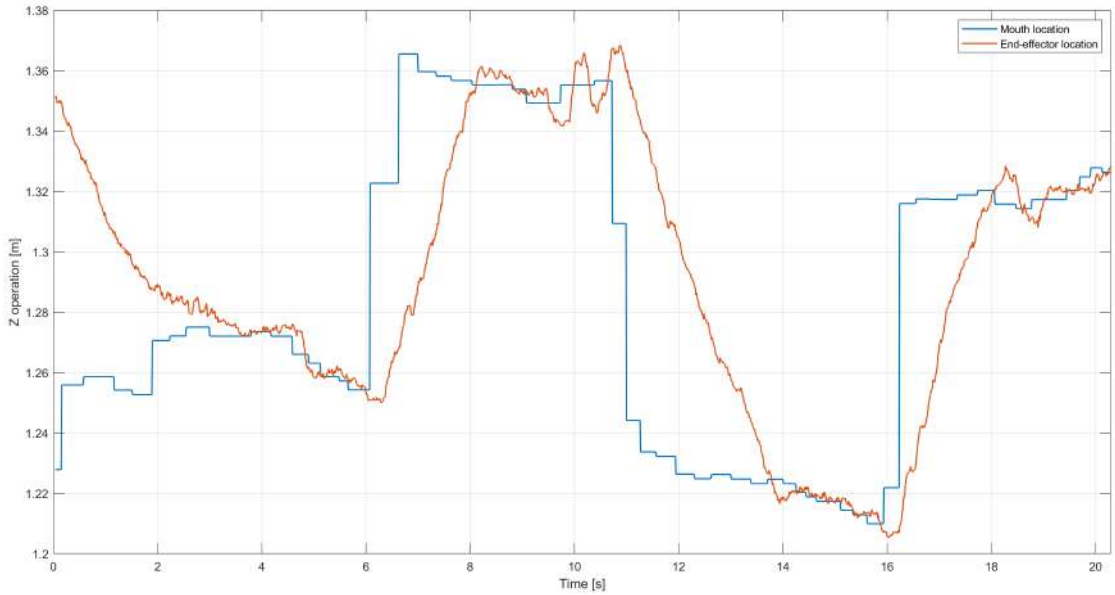


Figure 5.27: Baxter's end-effector z position during the mouth tracking test.

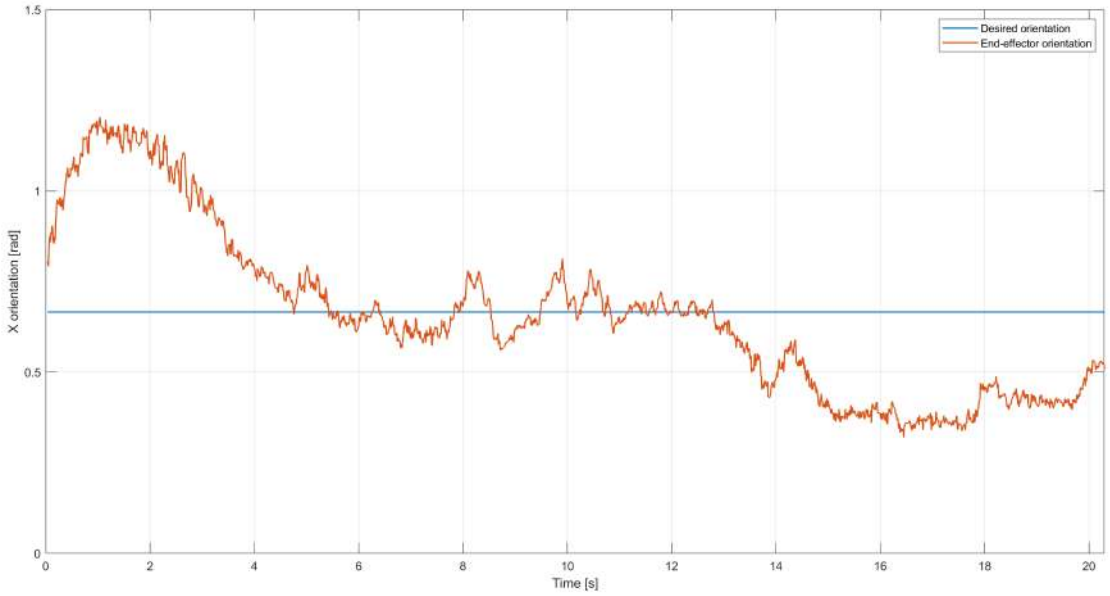


Figure 5.28: Baxter's end-effector x orientation during the mouth tracking test.

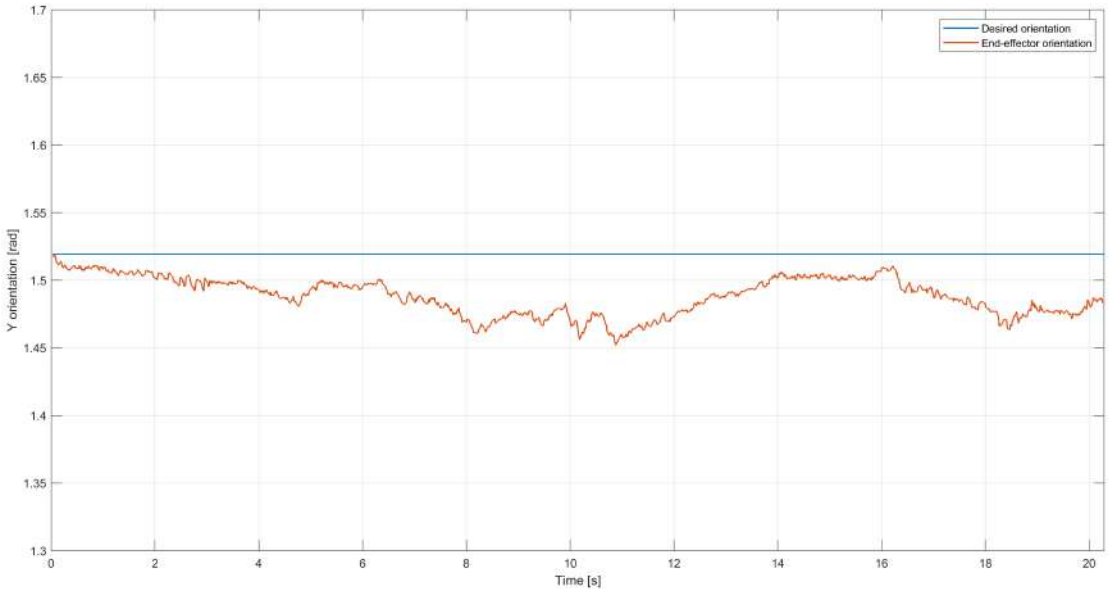


Figure 5.29: Baxter's end-effector y orientation during the mouth tracking test.

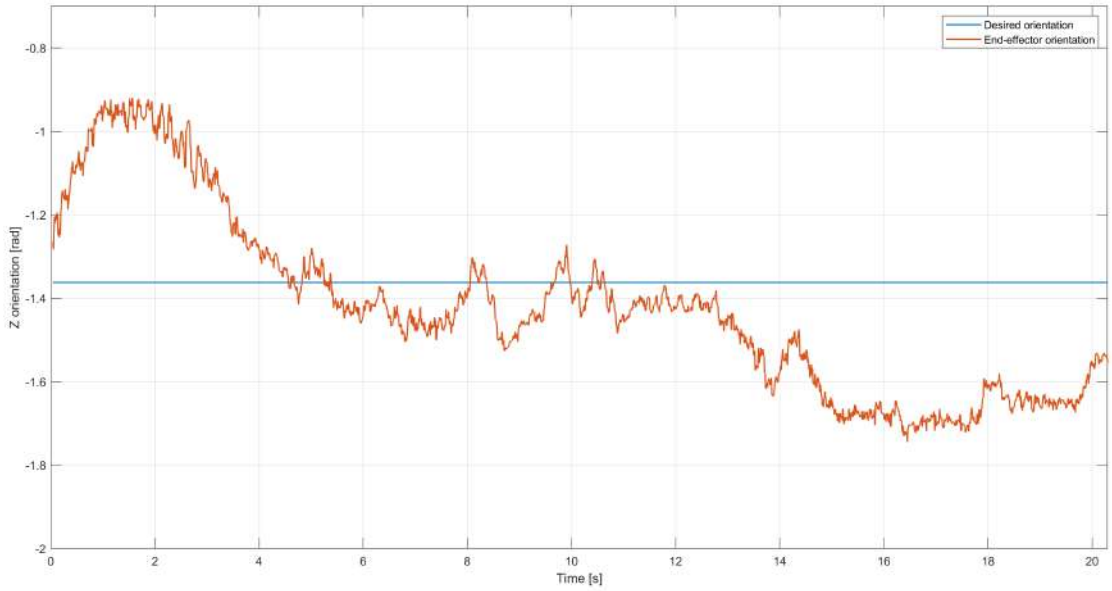


Figure 5.30: Baxter's end-effector z orientation during the mouth tracking test.

The results demonstrated that the MPC controller was able to follow the trajectory successfully. Some relevant metrics about the experiment are:

Performance metric	**X position**	**Y position**	**Z position**
Rise time (t_r) [sec]	3.17	1.96	2.03
Settling time (t_s) [sec]	4.87	2.84	3.01
Peak time (t_p) [sec]	3.39	2.2	2.32
Dead time (t_d) [sec]	0.24	0.43	0.29

Table 5.2: Performance metrics for mouth tracking experiment.

5.5.2 Regulation task

Chapter 6

ROS Integration

In order to create a robust software/hardware architecture for Baxter Bon Appetit project, it is necessary to design a proper message-passing middle-ware suit that enables a framework to operate Baxter robot. To fulfill these requirements it will be implemented a Robotic Operating System (ROS) architecture, which is an open-source robotic flexible framework (ROS Community, 2021a).

Throughout the article, multiple scripts and software developments have been explained in an isolated way (showing their own functionality without mentioning how they interact with the rest of the system elements). In this section, it will be explained how those functionalities are going to be integrated for Baxter Bon Appetit project.

6.1 ROS definitions

To understand how Baxter Bon Appetit ROS architecture is designed, it is mandatory to have a basic understanding of the following concepts:

- **ROS node**: is a process with emphasis in a single computational task. They are conceived as simple micro-services, that combined together, can create powerful robot functionalities (ROS Community, 2021c).
- **ROS message**: is a simple data structure with parameterized typed fields. Messages enable nodes to communicate with each other to topics (ROS Community, 2021b).
- **ROS topic**: is communication bus that helps nodes to exchange massages. Topics are design for unidirectional, streaming communication and have publisher/subscriber semantic, that isolate ROS nodes micro-services (ROS Community, 2021f).
- **ROS launch**: is a tool that allows launching multiples ROS nodes and some other useful functionalities such as setting parameters and automatically re-spawning dead nodes (ROS Community, 2021e).

6.2 Baxter Bon Appetit nodes architecture

In this section, it will be explained the general functionalities if the ROS nodes, that coupled together via ROS topics, create Baxter Bon Appetit architecture. The goal of

these explanations is to show how these nodes were developed with the best practices of a decoupled-system composed of multiple micro-services.

The ROS architecture is the way in which the nodes and the topics interact with each other in a micro-services approach. This structure can be seen in fig[6.1]:

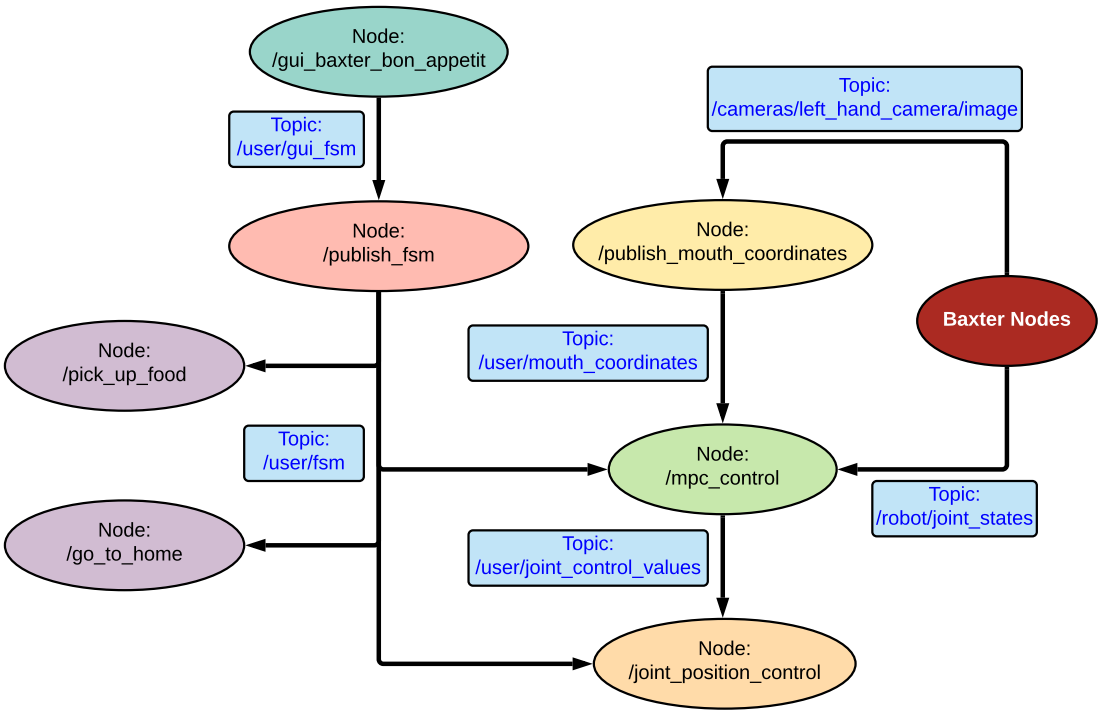


Figure 6.1: ROS architecture for Baxter Bon Appetit project. Own development.

- **Publish mouth coordinates**: this node communicates with the underlying Baxter's camera message and executes the mouth detection algorithms explained on sec[4.6]. The goal of this node is to publish the mouth location to a topic at a given frequency.
- **Go to home**: the objective of this node is to simultaneously move Baxter's limbs to the home position, that is the neutral setup for Baxter Bon Appetit operations as it is shown in fig[3.20]. This is the starting node for properly executing the feeding and scooping tasks.
- **MPC control**: this node is in charge of executing the model predictive control strategy explained on sec[5.3], in order to find the best possible configurations of the joint angles at a given time. The solution angles are published to a topic in a desired control frequency.
- **Joint position control**: the goal of this node is to send joint control commands to the robot based on the solution of the MPC control node.
- **Pick up food**: this node has the propose of moving Baxter's end-effector close to the food location and scooping the food in order to be ready for the feeding task.

- **Finite state machine**: this node aims to orchestrate all possible states of Baxter robot for Baxter Bon Appetit project, so that the application is conceived as a multi-state approach with different independent functionalities explained on sec[2.7].
- **Baxter GUI**: this nodes correspond to the graphical user interface that operates the finite state machine node in a friendly user experience approach. It is going to be explained on sec[6.3].
- **Baxter nodes**: they corresponds to a series of nodes that are implemented by default in Baxter robot. These nodes publish valuable topics with important information about the robot sensors and states.

6.3 Orchestrate Nodes with Graphical User Interface

In order to have a reliable and friendly user experience, it is useful to design a graphical user interface that enables the functionalities of Baxter Bon Appetit to any non-technical patient. To do so, the main features were abstracted and implemented based on the finite state machine explained on sec[2.7].

The major version of the GUI was designed with the help of a professional UI/UX tool called Figma, a collaborative interface web-based design tool specialized in the development of graphical applications (Figma Community, 2021). The main reason to implement Figma, is that it can easily generate different layouts using drag and drop functionalities without the need of using a programming language.

After the visual approach of the GUI was developed, it was converted into a Python-based Tkinter desktop application with the help of an open-source repository called "Tkinter-Designer", that using an API request, generates the Python code implementing native libraries (ParthJadhav, 2021).

The result of this procedure was the following:

Figure 6.2: Baxter Bon Appetit Graphical User Interface. Own development.

The way in which the GUI works, is with multiples Python methods, that when the buttons are pressed, they run callbacks to orchestrate the finite state machine for Baxter Bon Appetit.

One important remark is that the "START button" has the purpose of activating the ROS-launch tool for starting, orchestrating and automatically re-spawning the following nodes of Baxter Bon Appetit architecture:

- /go_to_home
- /publish_mouth_coordinates
- /mpc_control
- /joint_position_control

It is also relevant to notice that the "SHUTDOWN button" not only has the purpose of stopping the movements, but it also disables the internal joint controllers of the robot, replicating the functionalities of the physical emergency button of the system.

Another important functionality of the GUI is to make sure that the user only executes the absolutely necessary ROS nodes for Baxter Bon Appetit, avoiding the possibility of calling other programs that could interfere with each other. This is an extra layer of protection and compliance for the safety of the patient.

To dive deeper into the source code for the implementation of the Graphical User Interface and the underlying ROS-launch tool for orchestrating the nodes, feel free to analyze these Baxter Bon Appetit GitHub repository files:

- Baxter Bon Appetit ROS-launch tool implementation.
- Baxter Bon Appetit GUI node

Chapter 7

Conclusion

Mechatronics engineering seeks to assertively integrate multiple technological and scientific pillars through integrated solutions for modern problems. This research project aims to creatively integrate some of these pillars, such as robotics, computer vision, modern control and software integration, in a possible solution to a tangible necessity in today's society.

This study developed an active robotic solution designed through Baxter robot for the assistance of patients with upper limb motor disabilities.

There are medical devices for feeding patients on the market today, but none of them (currently) have active functionalities through systems that monitor the current position of the user's mouth and have a manipulator with anomaly detection capabilities.

Baxter Bon Appetit offers a breakthrough in active patient feeding systems as a theoretical and practical contribution to modern solutions of increasing problems in the world's population. This project is fully open-source and seeks a commitment towards promoting these developments for social inclusion and the continuous improvement of projects that support people with motor disabilities.

Finally, this mechatronics project manages to implement multiple spheres of knowledge in an innovative way, to meet a relevant need of society and thus, continue aiming to encourage the study of these active manipulative systems for the benefit of humanity.

Chapter 8

Ethical Considerations

Since ancient times, there have been several technological advances that have changed the world in positive and negative ways. It is our job as future Mechatronics Engineers from EIA University, to generate a positive impact on the development of society, where technological, social, environmental, cultural and legal areas, in our country and in the world, can be enhanced.

As students of the EIA University, we have always had as a main pillar the motto of "being, knowing and serving", so this project, from its conception to the final development, is a step towards the pursuit of a more equitable society for a target population that is highly affected by their physiological conditions. From our mechatronics scope, our main purpose will always be to help others and generate knowledge that contributes to improve the quality of life of these individuals with motor disabilities in the upper limbs.

Likewise, it is our great responsibility and duty, to abide by each and every one of the existing rules in the country and in the world, for the development of technological solutions that have people as direct elements in the performance tests, so we must rigorously consider the legal conditions related to this ethical reality of utmost importance and should seek to be viable from Ethics Committees around the world.

It is a great pleasure for us to be able to generate a path towards scientific dissemination where a positive social impact is generated, while integrating some of the nuclei of knowledge from the area of Mechatronics Engineering. Therefore, carrying out this project was not just a challenge for us, but also a space for contemplation and continuous projection towards a significant contribution to the society in which we live. These results are a step towards science and the development of a positive influence on society itself.

Finally, the aim of this research project was to seek a benefit for people with motor disabilities in the upper extremities and not to remain as a simple proposal, but to be the starting point to generate future projects and disclosures, which will continue and take up the methodologies presented in this article, with the aim of improving the quality of life of these individuals.

Chapter 9

Further investigation

Baxter Bon Appetit research project is a starting point for implementing a wide-variety of assisted robotics solutions. The internal algorithms for each of the subsystems that compose the architecture of the project, are subject to possible optimizations and upgrades.

One great advantage of the chosen software design, is that the main functionalities are divided into a series of independent micro-services that, gathered together, create a robust system that is most reliable. In addition to that, the architecture supports individual improvements for each one on the nodes, in order to make them faster, more optimal and adding new features.

As any investigation project, there are always possible branches and fields to develop new researches on top of the insights obtained from Baxter Bon Appetit project. Some possible areas of improvement are:

- The development of a web server application that expands the graphical user interface to remote devices such as smartphones, tablets and computers, allowing them to execute Baxter's functionalities remotely.
- Gathering, processing and analysis of soft real-time data in order to build new models that are tweaked for the feeding tasks based on the collected data and its insights.
- Creation of an interactive dashboard that works simultaneously with the graphical user interface in order to monitor and watch the desired variables of the system in a pseudo real-time approach.
- Test, train and implement new plausible anomaly detection and classification algorithms that are built with models based on data that is collected from Baxter Bon Appetit tasks and are constantly improving their performance.
- Explore new modern control techniques that are designed based on data and smart system models that seek a better performance,

Finally, to develop similar autonomous assisted robotics feeding solutions, it was developed a "Manual to replicate Baxter Bon Appetit to other robotic platforms", which is a general abstraction guide to systematically design and implement the functionalities of Baxter Bon Appetit similar robots. This guide can be found at:

- Baxter Bon Appetit: guide to replicate in other robots.

References

ACS Publications (1996). Limitations and countermeasures of pid controllers. https://pubs.acs.org/doi/abs/10.1021/ie960090+.

Aguiar, A. P. and Hespanha, J. P. (2007). Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty. 52(8):1362–1379. http://users.isr.ist.utl.pt/~pedro/CONAV/publications/IEEETACSupervisory.pdf.

Autodesk (2021). Fusion 360: Integrated cad, cam, cae, and pcb software. https://www.autodesk.com/products/fusion-360/overview.

Barrientos, A., Peñín, L. F., and Balaguer, C. (2007). *Fundamentos de robótica*. Second Edition. McGraw-Hill. https://books.google.com.co/books/about/Fundamentos_de_rob%C3%B3tica.html?id=ArEMPAAACAAJ&redir_esc=y.

Brilliant.org (2021). Finite state machines. https://brilliant.org/wiki/finite-state-machines/.

Burge, S. (2009). Pugh matrix analysis (pm). https://www.burgehugheswalsh.co.uk/uploaded/1/documents/pugh-matrix-v1.1.pdf.

ByungSoo Ko (2017). Convex optimization problem. https://kobiso.github.io/research/research-convex-optimization/.

C, G., D, P., and M, M. (1989). Model predictive control: Theory and practice-a survey. https://www.sciencedirect.com/science/article/abs/pii/0005109889900022.

Camacho, E. and Bordons, C. (2007). *Model Predictive Control*. 1. Springer-Verlag. https://www.springer.com/gp/book/9781852336943.

Canonical and Ubuntu Community (2014). Ubuntu 14.04.6 lts (trusty tahr). https://releases.ubuntu.com/14.04/.

Canudas, C., Siciliano, B., and (Eds), G. B. (1996). *The Theory of Robot Control*. 1. Springer. https://www.springer.com/gp/book/9781447115038.

Chen, T., Ciocarlie, M., Cousins, S., Grice, P., Hawkins, K., Hsiao, K., Kemp, C., King, C.-H., Lazewatsky, D., Leeper, A., Nguyen, H., Paepcke, A., Pantofaru, C., Smart, W., and Takayama, L. (2013). Robots for humanity using assistive robotics to empower people with disabilities. *Robotics & Automation Magazine, IEEE*, 20:30–39. https://www.researchgate.net/publication/260700777_Robots_for_Humanity_Using_Assistive_Robotics_to_Empower_People_with_Disabilities.

Cheshire, D. (2010). How to use a morphological matrix to generate ideas. https://innovationmanagement.se/2010/03/10/how-to-use-a-morphological-matrix-to-generate-ideas/#:~:text=A%20morphological%20matrix%20is%20a,been%20around%20for%20a%20while.

Cmglee (2021). Summed-area table. https://en.wikipedia.org/wiki/Summed-area_table#/media/File:Integral_image_application_example.svg.

Craig, J. J. (2004). *Introduction to Robotics: Mechanics and Control (3rd Edition)*. Third Edition. Pearson Education. https://www.amazon.com/Introduction-Robotics-Mechanics-Control-3rd/dp/0201543613.

Creality (2021). Ender-3 v2 3d printer. https://www.creality.com/goods-detail/ender-3-v2-3d-printer.

Davison, E. J., Aghdam, A. G., and Miller, D. E. (2020). *Centralized Control Systems*. Springer US.

Decker, M., Fischer, M., and Ott, I. (2017). Service robotics and human labor: A first technology assessment of substitution and cooperation. *Robotics and Autonomous Systems*, 87:348–354. https://www.sciencedirect.com/science/article/pii/S0921889016306285.

Dieter, G. and Schmidt, L. (2012). *Engineering Design*. Science Engineering & Math. https://www.amazon.com/-/es/George-Dieter-ebook/dp/B008K9XWQM.

Drgoňa and Ján (2017). Model predictive control with applications in building thermal comfort control. https://www.researchgate.net/publication/323219837_Model_Predictive_Control_with_Applications_in_Building_Thermal_Comfort_Control.

E, King et al (2021). Github repository: dlib c++ library. https://github.com/davisking/dlib.

Edward, J., Wannasuphoprasit, W., and Peshkin, M. (1999). Cobots: Robots for collaboration with human operators. https://www.researchgate.net/publication/2808147_Cobots_Robots_For_Collaboration_With_Human_Operators.

Fattal, C., Leynaert, V., Laffont, I., Baillet, A., Enjalbert, M., and Leroux, C. (2019). Sam, an assistive robotic device dedicated to helping persons with quadriplegia: Usability study. *International Journal of Social Robotics*, 11(1):89–103. https://doi.org/10.1007/s12369-018-0482-7.

Figma Community (2021). Figma: the collaborative interface design tool. https://www.figma.com.

Gama, J. and Brazdil, P. (2000). Cascade generalization. https://link.springer.com/article/10.1023/A:1007652114878.

Geitgey, A. (2016). Machine learning is fun! part 3: Deep learning and convolutional neural networks. https://medium.com/@ageitgey/machine-learning-is-fun-part-3-deep-learning-and-convolutional-neural-networks-f40359318721.

Geitgey, A et al (2021). Github repository: Face recognition. https://github.com/ageitgey/face_recognition.

Genesis Systems (n.d.). Robots in automotive manufacturing: Top 6 applications. https://www.genesis-systems.com/blog/robots-automotive-manufacturing-top-6-applications.

GNU Operating System (n.d.). What is bash? https://www.gnu.org/software/bash/manual/html_node/What-is-Bash_003f.html.

Guizzo, E. and Ackerman, E. (2012). How rethink robotics built its new baxter robot worker. https://spectrum.ieee.org/robotics/industrial-robots/rethink-robotics-baxter-robot-factory-worker?utm_source=robots.ieee.org.

Harvard Business Review (1988). The house of quality. https://hbr.org/1988/05/the-house-of-quality.

ICBF (2016). Guía Ténica del Componente de Alimentación y Nutrición para la Población con Discapacidad. www.icbf.gov.co/sites/default/files/procesos/g7.pp_guia_tecnica_de_alimentacion_y_nutricion_para_poblacion_en_discapacidad_v1.pdf.

IEEE Robots (2012). Baxter. https://robots.ieee.org/robots/baxter/?gallery=photo3.

J, B., A, R., and J, H. (2002). Receding horizon control of autonomous aerial vehicles. https://www.semanticscholar.org/paper/Receding-horizon-control-of-autonomous-aerial-Bellingham-Richards/a5d00bf04c0dcb71157ea4211cbad9a0e3c975de#citing-papers.

J. Denavit and R. Harterberg (1955). A kinematic notation for lower-pair mechanisms based on matrices. https://home.konkuk.ac.kr/~cgkang/courses/robotics/courseMaterials/DHpaper.pdf.

Jaffe, D. L., Nelson, D., and Thiemer, J. (2012). Perspectives in assistive technology engr110/210. http://web.stanford.edu/class/engr110/2012/04b-Jaffe.pdf.

Jain, A., Killpack, M. D., Edsinger, A., and Kemp, C. C. (2013). Reaching in clutter with whole-arm tactile sensing. *RThe international Journal of Robotics Research*. https://journals.sagepub.com/doi/10.1177/0278364912471865.

Jalani, N. H., Jalani, J., and Abdullah, J. (2008). Position control for linear motion servo system via nominal characteristic trajectory following (nctf) controller. https://www.researchgate.net/publication/267211142_POSITION_CONTROL_FOR_LINEAR_MOTION_SERVO_SYSTEM_VIA_NOMINAL_CHARACTERISTIC_TRAJECTORY_FOLLOWING_NCTF_CONTROLLER.

Klančar, G., Zdešar, A., Blažič, S., and Škrjanc, I. (2017). *Wheeled Mobile Robotics From Fundamentals Towards Autonomous Systems*. Elsevier Inc. https://www.sciencedirect.com/book/9780128042045/wheeled-mobile-robotics#book-info.

Kumar, A. (2020). The ultimate guide to adaboost algorithm — what is adaboost algorithm? https://www.mygreatlearning.com/blog/adaboost-algorithm/.

Mathworks (2021). Integral image. https://la.mathworks.com/help/images/integral-image.html.

Mayo Clinic (2020). Parkinson's disease. https://www.mayoclinic.org/diseases-conditions/parkinsons-disease/symptoms-causes/syc-20376055.

Mehta, B. and Reddy, Y. (2015). *Industrial Process Automation Systems Design and Implementation*. Elsevier Inc. https://www.sciencedirect.com/book/9780128009390/industrial-process-automation-systems.

Montalvo M, M. (2010). Técnicas de visión estereoscópica para determinar la estructura tridimensional de la escena. https://eprints.ucm.es/id/eprint/11350/1/T%C3%A9cnicas_de_visi%C3%B3n_estereosc%C3%B3pica_para_determinar_la_estructura_tridimensional_de_al_escena.pdf.

Open Source Computer Vision (2021). Cascade classifier. https://docs.opencv.org/4.5.2/db/d28/tutorial_cascade_classifier.html.

OpenJS Foundation (2021). Build cross-platform desktop apps with javascript, html, and css. https://www.electronjs.org.

Park, D., Erickson, Z., Bhattacharjee, T., and Kemp, C. C. (2016). Multimodal execution monitoring for anomaly detection during robot manipulation. pages 407–414. https://ieeexplore.ieee.org/document/7487160.

Park, D., Hoshi, Y., and Kemp, C. C. (2018). A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder. *IEEE Robotics and Automation Letters*, 3(3):1544–1551. https://ieeexplore.ieee.org/document/8279425.

Park, D., Hoshi, Y., Mahajan, H. P., Kim, H. K., Erickson, Z., Rogers, W. A., and Kemp, C. C. (2020). Active robot-assisted feeding with a general-purpose mobile manipulator: Design, evaluation, and lessons learned. *Robotics and Autonomous Systems*, 124:103344. https://www.sciencedirect.com/science/article/pii/S0921889018307061.

Parkinson's Foundation (2020). Statistics. https://www.parkinson.org/Understanding-Parkinsons/Statistics.

ParthJadhav (2021). ParthJadhav/Tkinter-Designer: Create Beautiful Tkinter GUIs by Drag and Drop. https://github.com/ParthJadhav/Tkinter-Designer.

Pertuz, S., Llanos, C., and Muñoz, D. (2018). Simulation and implementation of impedance control in robotic hand. https://www.researchgate.net/publication/324363084_Simulation_and_Implementation_of_Impedance_Control_in_Robotic_Hand.

Poe, W. A. and Mokhatab, S. (2017). Internal model control. https://www.sciencedirect.com/topics/engineering/internal-model-control.

Poltawski, L., Allison, R., Briscoe, S., Freeman, J., Kilbride, C., Neal, D., Turton, A. J., and Dean, S. (2016). Assessing the impact of upper limb disability following stroke: a qualitative enquiry using internet-based personal accounts of stroke survivors. *Disability and Rehabilitation*, 38(10):945–951. https://doi.org/10.3109/09638288.2015.1068383.

Python Software Foundation (2021). Python. https://www.python.org.

Python Software Foundation [US] (2021). tkinter — python interface to tcl/tk. https://docs.python.org/3/library/tkinter.html.

Rensselaer Polytechnic Institute (2017). Mobile human-friendly assistive robot. https://patentimages.storage.googleapis.com/c2/f4/81/2125f3800c4e4b/US20170095382A1.pdf.

Rethink Robotics (2015a). Arms. https://sdk.rethinkrobotics.com/wiki/Arms.

Rethink Robotics (2015b). Baxter research robot hardware specifications. https://sdk.rethinkrobotics.com/wiki/Hardware_Specifications.

Rethink Robotics (2015c). Baxter research robot wiki. https://sdk.rethinkrobotics.com/wiki/Home.

Rethink Robotics (2015d). Baxter research robot wiki. https://sdk.rethinkrobotics.com/wiki/Home.

Rethink Robotics (2015e). Baxter research robot workspace guidelines. https://sdk.rethinkrobotics.com/wiki/Workspace_Guidelines.

Rethink Robotics (2015f). Cameras. https://sdk.rethinkrobotics.com/wiki/Cameras.

Rethink Robotics (2015g). Hardware specifications. https://sdk.rethinkrobotics.com/wiki/Hardware_Specifications.

Rethink Robotics (2015h). Input and output. https://sdk.rethinkrobotics.com/wiki/Input_and_Output.

Rethink Robotics (2015i). Robot arm calibration. https://sdk.rethinkrobotics.com/wiki/Arm_Calibration.

Rethink Robotics (2020). Rethink robotics meets german engineering. https://www.rethinkrobotics.com.

Robot Academy (n.d.). Robot workspace. https://robotacademy.net.au/lesson/robot-workspace/.

Robots Done Right (2021). Abb robots. https://robotsdoneright.com/ABB-Robots.html.

ROS Community (2019). Understanding ros nodes. http://wiki.ros.org/ROS/Tutorials/UnderstandingNodes.

ROS Community (2021a). About ros. https://www.ros.org/about-ros/.

ROS Community (2021b). Messages. http://wiki.ros.org/Messages.

ROS Community (2021c). Nodes. http://wiki.ros.org/Nodes.

ROS Community (2021d). Robot operating system. https://www.ros.org.

ROS Community (2021e). roslaunch. http://wiki.ros.org/roslaunch.

ROS Community (2021f). Topics. http://wiki.ros.org/Topics.

ROS organization (2014). Ros indigo igloo. http://wiki.ros.org/indigo.

Rosenfeld, A. (1988). Computer vision. 27:265–308. https://www.sciencedirect.com/science/article/pii/S0065245808602612?via=ihub.

Sharma, M. (2020). Introduction to robotic control systems. https://towardsdatascience.com/introduction-to-robotic-control-systems-9ec17c8ac24f.

Singh, S. and Prasad, S. (2018). Techniques and challenges of face recognition: A critical review. *Procedia Computer Science*, 143:536–543. 8th International Conference on Advances in Computing & Communications (ICACC-2018).

Talend (n.d.). What is middleware? technology's go-to middleman. https://www.talend.com/resources/what-is-middleware/.

Tang, S., Zhu, Q., Chen, W., Darwish, W., Wu, B., Hu, H., , Chen, M., and Li, J. (2016). Enhanced rgb-d mapping method for detailed 3d indoor and outdoor modeling. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5087378/.

United Nations (2012). Take action for the sustainable development goals. https://www.un.org/sustainabledevelopment/sustainable-development-goals/.

University EIA (2021). Mechatronics engineering. https://www.eia.edu.co/ingenieria-mecatronica/.

Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features.

Vuze (2021). Vuze+ 3d 360 vr camera. https://vuze.camera/camera/vuze-plus-camera.

Williams, R. L. (2017). Baxter humanoid robot kinematics. https://www.ohio.edu/mechanical-faculty/williams/html/PDF/BaxterKinematics.pdf.

Winston, P. H. (2014). Artificial intelligence: Learning: Boosting. https://www.youtube.com/watch?v=UHBmv7qCey4&list=PLUl4u3cNGP63gFHB6xb-kVBiQHYe_4hSi&index=18.

Yavuz, S. C. (2009). Kinematic analysis for robot arm. https://www.researchgate.net/profile/Mohamed_Mourad_Lafifi/post/how_to_calculate_the_angular_velocity_of_end_effector_of_two_link_robot_arm/attachment/59d6487c79197b80779a326c/AS%3A467097353494532%401488376095512/download/KINEMATIC+ANALYSIS+FOR+ROBOT+ARM+Thesis.pdf.

YuichiAraki, NobutakaShimada, and YoshiakiShirai (2001). Face detection and face direction estimation using color and shape features. pages 229–234.

Zaghetto, C., Aguiar, L. H., Zaghetto, A., Ralha, C. G., and de Barros Vidal, F. (2016). Agent-based framework to individual tracking in unconstrained environments. https://www.researchgate.net/publication/317607006_Agent-based_framework_to_individual_tracking_in_unconstrained_environments.

ZED Cameras (n.d.). Zed 2 stereo camera. https://chironix.com/products/zed-stereo-camera.