Universidade do Minho
Escola de Engenharia

Carlos Alberto Peixoto Borges

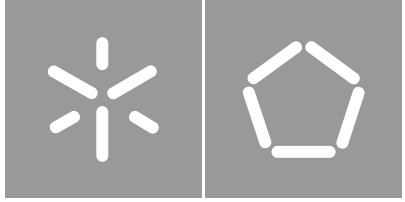
Inspection of Deformable Objects via Robotic Manipulation: a Learning from Demonstration Approach

June 2022

Inspection of Deformable Objects via Robotic Manipulation: a Learning from Demonstration Approach

Carlos Borges

UMinho | 2022



Universidade do Minho
Escola de Engenharia

Carlos Alberto Peixoto Borges

Inspection of Deformable Objects via Robotic Manipulation: a Learning from Demonstration Approach

Master's Dissertation
Integrated Master in Mechanical Engineering
Mechatronic Systems

Work developed under the supervision of:
Professor António Alberto Caetano Monteiro
and
Professor Estela Guerreiro da Silva Bicho Erlhagen

June 2022

ACKNOWLEDGMENTS

First, I would like to express my gratitude to my supervisors, Professor Caetano Monteiro, and Professor Estela Bicho, for the valuable guidance, encouragement, availability to help me when I most needed, and genuine concern for my well-being. Particularly to Professor Caetano I am grateful for the effort made during the past years to help me in my favorite field of study. Especially to Professor Estela I thank for all the "outside of the box" suggestions that were crucial to this work.

Although Professor Luís Louro and Professor Sérgio Monteiro were not officially my supervisors, they demonstrated a level of commitment and dedication to my work as if they were. As such, I would like to express my deepest gratitude to them too. I am also truly grateful to my laboratory colleagues, especially Paulo Vicente, for all the help in different areas.

If this dissertation could have a second author, it would undoubtedly be Miguel Maia. Almost every interesting aspect presented here started in discussion with him, and I am sure that without his ideas, many problems would remain unsolved until today. To you my friend, a very special thanks.

I thank Junchen Wang, from Beihang University, for the availability and promptness to help, mainly in topics related to singularities, and to Fabio Pando, from Kuka, for all the information related to the robotic arm used.

To my family, particularly to my parents and sister, I have to thank for all the investment in my education, for the unconditional support, comprehension, and inspiration. I sincerely hope you feel that all the efforts were worthwhile. I also cannot forget to express my gratitude to my friends, who were always available to comfort me and transmit confidence.

Lastly, I want to thank both Professor Estela Bicho and Professor Sérgio Monteiro for the opportunity to work on the exciting project IntVIS4Insp (ref.: POCI-01-0247-FEDER-042778) with the grant support provided by "Fundação para a Ciência e a Tecnologia" with the reference UMINHO/BIM/2021/35.

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

RESUMO

A aplicação generalizada da inspeção automática e o interesse crescente pelo controlo de qualidade impulsionam a procura de sistemas capazes de inspecionar objetos cada vez mais complexos. Neste âmbito surgiu o projeto de Investigação & Desenvolvimento "IntVIS4Insp", no qual esta dissertação se insere. As peças de couro são um exemplo importante do tipo de objetos que este projeto pretende cobrir, porque implicam manipulação no seu processo de inspeção. Uma abordagem que usa aprendizagem por imitação, baseada em dois sistemas, pode ser aplicada para automatizar os movimentos necessários para as inspecionar. Primeiro, um sistema de visão por computador capta as trajetórias descritas pela mão ativa de um técnico de qualidade. Segundo, um braço robótico (Kuka LBR iiwa) reproduz as trajetórias anteriormente adquiridas. Esta dissertação foca-se no segundo sistema.

Dois desafios relacionados têm de ser resolvidos no âmbito do seguimento de trajetórias: (i) determinar como comandar o manipulador, sem exceder limites de juntas ou atravessar singularidades, de modo que a garra siga a trajetória desejada, ou seja, resolver um problema de equivalência física; (ii) descobrir como aumentar a distância aos limites das juntas e singularidades por forma a aumentar a capacidade de o manipulador alterar arbitrariamente a pose da garra durante a execução da trajetória. A resolução deste segundo problema é essencial para, nas tarefas futuras do projeto IntVIS4Insp (e noutros), ser possível implementar um sistema de correção da trajetória baseado nas deformações induzidas nas peças de couro em cada instante. Este último sistema aumentará a robustez da abordagem utilizada.

Para resolver estes dois problemas é proposto um novo método para seguimento de trajetórias que considera a manipulabilidade. Fundamentalmente, como as trajetórias podem ser realizadas em diferentes zonas do espaço de trabalho do manipulador, esta liberdade é utilizada com as várias soluções do problema da cinemática inversa e com a redundância do cotovelo para aumentar a distância aos limites das juntas e singularidades. Abordagens estado da arte são utilizadas como ponto de partida, mas são alargadas para considerarem a continuidade do *arm angle* no limite do seu domínio.

O método proposto foi testado em trajetórias de inspeção reais (em simulação e com o manipulador real) e os resultados demonstram 3 pontos: (i) as trajetórias foram corretamente reproduzidas; (ii) conseguiu-se uma elevada manipulabilidade (distância média mínima de 42,6° aos limites das juntas e singularidades); (iii) a abordagem utilizada permite induzir as deformações desejadas nos objetos.

Palavras-Chave: Limites de Juntas, Manipulabilidade, Manipulador Robótico, Seguimento de Trajetórias, Singularidades.

ABSTRACT

The widespread application of automatic inspection and the increasing interest in quality control boost the search for systems capable of inspecting increasingly complex objects. In this scope emerged the Research & Development project "IntVIS4Insp", in which this dissertation is inserted. Leather pieces are an important example of the object type this project intends to address because their inspection process involves manipulation. A learning from demonstration approach, based on two systems, can be used to automate the necessary movements to inspect them. First, a computer vision system acquires the trajectories described by the active hand of a quality control technician. Second, a robot arm (Kuka LBR iiwa) reproduces these trajectories. This dissertation is focused on the latter system.

Two related challenges must be solved in a trajectory tracking problem: (i) determine how to command the manipulator so that its end-effector follows the desired trajectory without exceeding joint limits and crossing singularities, i.e., to solve a physical equivalence problem; (ii) find out how to increase the distance to joint limits and singularities to enhance the manipulator's capacity to change its tip's pose arbitrarily during the execution of the trajectory. Solving this second problem is essential so that, in future tasks of the IntVIS4Insp project (and others), it will be possible to implement a trajectory correction system based on the deformations induced in the leather pieces at each instant. The latter system will increase the robustness of the overall learning from the demonstration approach.

To solve both these problems, a novel trajectory tracking method considering manipulability is proposed. Fundamentally, as the trajectories can be executed in different zones of the manipulator's workspace, this freedom is used with the various possibilities for the inverse kinematics problem and the elbow redundancy to increase the distance to joint limits and singularities. State-of-the-art approaches are used as a starting point, but they are extended to consider the continuity of the arm angle in its domain boundary.

The proposed method was tested on real inspection trajectories (in simulation and with the actual manipulator), and the results demonstrate 3 points: (i) the trajectories were correctly reproduced; (ii) a high manipulability was obtained (minimum average distance of 42.6° to joint limits and singularities); (iii) the used approach allows to induce the desired deformations in the leather parts.

Keywords: Joint Limits, Manipulability, Robotic Manipulator, Singularities, Trajectory Tracking.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

Acronym	Description
AR	Augmented Reality
CAD	Computer Aided Design
CG	Center of Gravity
CPD	Coherent Point Drift
CSV	Comma Separated Values
DH	Denavit–Hartenberg convention
DoF	Degrees of Freedom
FK	Forward Kinematics
GC	Global Configuration
GUI	Graphical User Interface
HMI	Human Machine Interface
IK	Inverse Kinematics
IO	Input–Output communication
IP	Internet Protocol
LfD	Learning from Demonstration
PbD	Programming by Demonstration
RC	Redundancy Circle
ROS	Robot Operating System
RTU	Remote Terminal Unit
SE_vW	Shoulder–Elbow of the virtual manipulator–Wrist
SEW	Shoulder–Elbow–Wrist
S-R-S	Spherical–Rotational–Spherical
SW	Shoulder–Wrist
TPS–RPM	Thin Plate Spline–Robust Point Matching
TSM–RPM	Tangent Space Mapping–Robust Point Matching
UDP	User Datagram Protocol

LIST OF SYMBOLS

Symbol	Description	Used unit
$\boldsymbol{\alpha}$	Constant that controls the distance to the borders from where the repulsion begins, $\alpha \in \mathbb{R}^+$	-
$\boldsymbol{\alpha_i}$	DH parameter: angle between $\hat{z}_{i-1}$ and $\hat{z}_i$ about $\hat{x}_i$ (positive rotation direction is counter-clockwise)	rad
$\boldsymbol{\gamma}$	Angle formed by the shoulder–elbow, and elbow–wrist vectors	rad
$\boldsymbol{\delta}$	Arm angle safety margin for singularities	rad
$\boldsymbol{\theta_i}$	Angular value of joint i/DH parameter: angle between $\hat{x}_{i-1}$ and $\hat{x}_i$ about $\hat{z}_{i-1}$ (positive rotation direction is counter-clockwise)	rad
$\boldsymbol{\theta_i^v}$	Angular value of joint i in the virtual manipulator	rad
$\boldsymbol{\theta^l}$	Lower joint limit	rad
$\boldsymbol{\theta^u}$	Upper joint limit	rad
$\boldsymbol{\theta_{4\,mean}}$	Mean of the joint 4 distances to its limit/singularity for each path	rad
$\boldsymbol{\theta_{4\,min}}$	Minimum of the joint 4 distances to its limit/singularity for each path	rad
$\boldsymbol{\phi}$	Angle formed by the shoulder–wrist, and shoulder–elbow vectors	rad
$\boldsymbol{\psi}$	Arm angle	rad
$\boldsymbol{\psi_{(t)}}$	Arm angle to be determined in the current pose	rad
$\boldsymbol{\psi_{(t-1)}}$	Arm angle from the previous pose	rad
$\boldsymbol{\psi^l}$	Lower limit of the current pose feasible interval that contains $\psi_{(t-1)}$	rad
$\boldsymbol{\psi_{mean}}$	Mean of the shortest distances of the arm angles to the limits of the feasible intervals for each path	rad
$\boldsymbol{\psi_{min}}$	Minimum of the shortest distances of the arm angles to the limits of the feasible intervals for each path	rad
$\boldsymbol{\psi^u}$	Upper limit of the current pose feasible interval that contains $\psi_{(t-1)}$	rad

Symbol	Description	Used unit
$\boldsymbol{A_s}$, $\boldsymbol{B_s}$, $\boldsymbol{C_s}$	Auxiliar matrices used in the calculation of ${}^0R_3$ with the Rodrigues' rotation formula, and also of ${}^4R_7$	-
$\boldsymbol{A_w}$, $\boldsymbol{B_w}$, $\boldsymbol{C_w}$	Auxiliar matrices used in the calculation of ${}^4R_7$	
$\boldsymbol{a_i}$	DH parameter: distance from frame $i-1$ to frame i measured along $\hat{\mathrm{x}}_i$	m
$\boldsymbol{a_n/a_d/a}$	Generic representation of an element of the matrices A_s or A_w. The subscript n stands for numerator or first input, the d for denominator or second input.	-
$\boldsymbol{a_{smn}}$, $\boldsymbol{b_{smn}}$, $\boldsymbol{c_{smn}}$	Element of the m row and n column of the auxiliar matrices A_s, B_s, C_s, respectively	-
$\boldsymbol{b_n/b_d/b}$	Generic representation of an element of the matrices B_s or B_w. The subscript n stands for numerator or first input, the d for denominator or second input.	-
$\boldsymbol{c}$	Notation simplification for cos	rad
$\boldsymbol{c_n/c_d/c}$	Generic representation of an element of the matrices C_s or C_w. The subscript n stands for numerator or first input, the d for denominator or second input.	-
$\boldsymbol{d_{bs}}$	Base–shoulder distance	m
$\boldsymbol{d_{ew}}$	Elbow–wrist distance	m
$\boldsymbol{d_i}$	DH parameter: distance from frame $i-1$ to frame i measured along $\hat{\mathrm{z}}_{i-1}$	m
$\boldsymbol{d_{se}}$	Shoulder–elbow distance	m
$\boldsymbol{d_{wt}}$	Wrist–tip distance	m
$\boldsymbol{GC_i}$	Configuration flags for the joints 2, 4, and 6.	-
$\boldsymbol{I_n}$	Identity matrix of size $\mathrm{n} \times \mathrm{n}$	-
$\boldsymbol{K}$	Constant that controls the strength of repulsion from the interval limits, $K \in [0,1]$	-
$\boldsymbol{ms}$	Score attributed to each path	-
${}^{\boldsymbol{i}}\boldsymbol{p_{jk}}$	Vector that goes from the origin of frame j to the origin of frame k, with coordinates relative to frame i	m

Symbol	Description	Used unit
$^{0}\boldsymbol{R}_{\psi}$	Rotation matrix resulting from the Rodrigues' rotation formula	-
$\boldsymbol{s}$	Notation simplification for sin	rad
$\widehat{\boldsymbol{v}}_{\boldsymbol{SEW}}$	Vector normal to the plane SEW	-
$\widehat{\boldsymbol{v}}_{\boldsymbol{SEW}}^{\boldsymbol{v}}$	Vector normal to the plane SE$_v$W	-
$\widehat{\boldsymbol{v}}\times$	Cross-product matrix for the unit vector $\hat{v}$	-

1. INTRODUCTION

The present work is part of the project IntVIS4Insp–Intelligent & Flexible Computer Vision System for Automatic Inspection (reference: POCI-01-0247-FEDER-042778). This project intends to develop automatic quality control systems capable of inspecting complex objects. Here, complex objects are considered the ones with specular or textured characteristics, complex tridimensional geometry, and features only visible through manipulation. This dissertation will mainly focus on the latter example, particularly in defects of small leather pieces with buckling strength (present resistance when one tries to approximate their extremities). Typically, these objects' inspection process involves movements to highlight the existing defects, e.g., cuts, folds, pokes, and glue spills. Giving a practical example, the reader (without looking to the next page) is challenged to identify all the cuts present in the leather piece of the figure below (Figure 1.1), which is flat on a surface. Then, the number of detected cuts can be compared with the one observed in Figure 1.2, where the object is bent.

Due to its complexity, this task is done manually, which raises the problem of musculoskeletal injuries. Highly skilled human operators are required to ensure good quality control, but these are hard to hire. Hence, to provide a healthier industry along with highly demanding customer expectations, it is of paramount importance to automate, at least in part, the task of leather pieces inspection. The work here reported makes several steps toward this objective. The ultimate goal is to develop a highly flexible robotic system that can emulate the inspection movements performed by human operators.

Figure 1.1 – Cut defects in a leather piece when it is flat on a surface.

Figure 1.2 – Cut defects in a leather piece when it is bent.

To automate the presented task, a Learning from Demonstration (LfD) approach[1], based on three systems, is used on the introduced project:

- First, the movements of a person's hands during the manipulation are acquired with a computer vision system. Other methods could be used to obtain the trajectories, such as teleoperation and kinesthetic teaching, but the allowed freedom of movement to the person performing the exemplification would be reduced.
- Then, a robotic system reproduces these trajectories. All the inspection movements considered are complementary between both hands, which means that to enforce a particular configuration on the object does not matter the movement of each hand independently but rather the movement relative to each other. Therefore, the inspection can be made with only one active robotic arm (with a 7 Degrees of Freedom (DoF) serial manipulator due to its dexterity), which is responsible for performing the inspection movements, and a fixed grasp support with a holding function (Figure 1.3).
- A second computer vision system analyzes the leather pieces during the manipulation.

It is crucial to note that, in the future, another computer vision system will be used to verify if the desired deformations are actually being induced in the leather pieces. If they are not, online corrections will need to be introduced in the trajectories. This will increase the robustness of the overall system and allow to deal, for example, with minor variations in the grasping points of the leather pieces.

[1] The main reasons why a learning from a demonstration approach must be used in the introduced project will be discussed in Section 2.

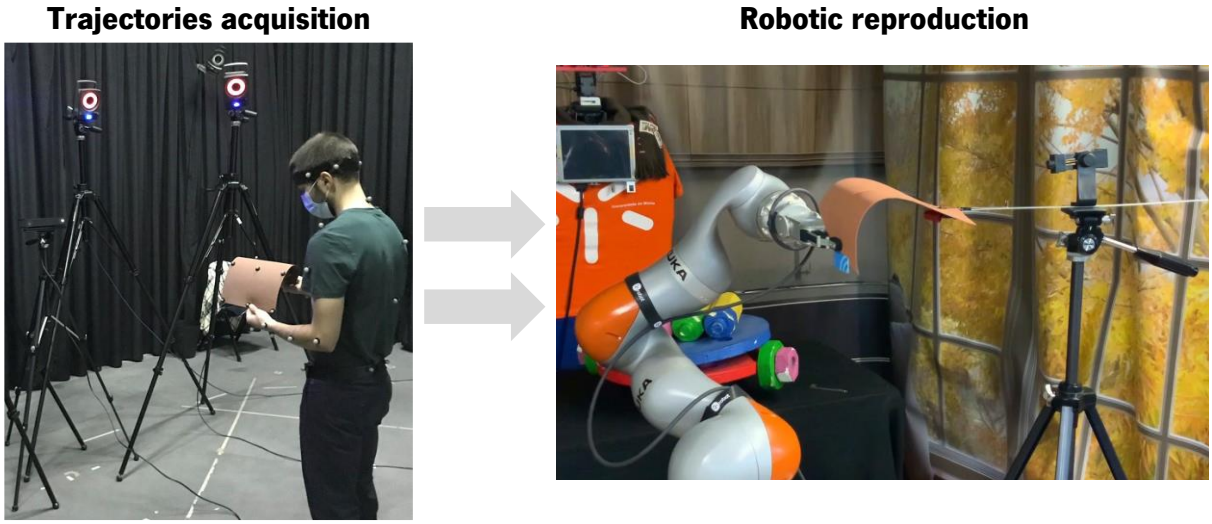


Figure 1.3 – Approach used to automate the necessary manipulation to inspect the leather pieces.

It was the goal of this dissertation to focus on the second system previously presented (i.e., the robotic one) and meet the following objectives:

- Obtain a trajectory tracking method capable of making the tip of a 7-DoF serial manipulator follow the acquired inspection trajectories, i.e., solve a physical equivalence problem. This method must consider the manipulability of the robotic arm at every via point of the trajectory, i.e., the capability to change the position and orientation of the end-effector arbitrarily. This capability will help ensure that the online corrections one intends to introduce in future works can be performed.
- Test the developed method with real inspection trajectories and analyze the results regarding manipulability.
- Validate the tracking of the trajectories and the safety of the generated movements using a robotics simulator.
- Mount a physical setup where the active manipulator must be the Kuka LBR iiwa 14 R820. All the other aspects should be studied and implemented, e.g., gripper to be used, and communication/command interfaces. All the decisions must consider the online corrections that will be implemented in the future. The available ROS (Robot Operating System) interfaces should be prioritized because they accelerate the component's integration process and the development of the necessary programs. With the obtained setup, it must be proved that the reproduced trajectories induce in the leather pieces the desired deformations.

The present dissertation is organized as follows: Section 2 mainly introduces learning from demonstration and presents the primary methods found in the literature that can be useful to solve a trajectory tracking problem considering manipulability. Section 3 uses approaches exposed in Section 2 to solve kinematic

problems related to a 7–DoF serial manipulator (e.g., forward kinematics and inverse kinematics). In Section 4, the two previous sections are used as a base to present a novel trajectory tracking method considering manipulability. Also in this chapter, the presented method is extensively tested using real inspection trajectories. Finally, Section 5 presents an overview of all the work done and extracts the main conclusions as well as the future prospects.

The most important contribution of this dissertation is the method exposed in Section 4, which gave rise to the paper "Trajectory tracking for the inspection of deformable objects considering manipulability of a 7-DoF serial manipulator", presented at the 2022 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC) (Borges et al., 2022).

2. LEARNING FROM DEMONSTRATION

This chapter first presents the essential aspects of learning from demonstration. Then, because of its importance to this dissertation, the physical equivalence problem is explored in detail. Finally, the generalization of demonstrations in the scope of deformable objects is studied, and the chapter's conclusions are presented.

2.1 Fundamental aspects of learning from demonstration

In robotics, Learning from Demonstration[2] (LfD) represents a paradigm in which robots learn by imitating experts, thereby acquiring new skills (Ravichandar et al., 2020). One of the significant advantages of this method is that it can be used by those who have no programming capabilities. Users only have to exemplify how to perform the task they want to automate because the robotic system can collect a set of data and use it later to replicate the exemplified task (Billard et al., 2016; Zhu & Hu, 2018). In this dissertation, this advantage is essential because people who know how to highlight the defects in leather pieces do not know how to program the robots. In addition, LfD is also advantageous in the sense that, unlike other methods used to manipulate deformable objects, avoids modeling (see, for example, Sanchez et al. (2020)) and motion planning (see, for instance, Saha and Isto (2007)) (McConachie et al., 2020). Despite the advantages presented above, LfD also has disadvantages. In this method, the actions generated may be limited by the demonstrator's capabilities, not by the robot. The latter may be, for example, faster than the former. Thus, to take full advantage of the robotic system capabilities, there may be a need to resort to specific techniques such as Reinforcement Learning (see Sutton and Barto (2018) for an introduction to the topic). Furthermore, if a task is to be performed under different conditions than those in which the demonstrations were performed (with pieces of leather with different sizes or properties, for example), they can become inadequate. It may be necessary to generalize the exemplifications or perform new ones (Argall et al., 2009; McConachie et al., 2020). Due to its importance, the first option will be detailed in subchapter 2.3.

[2] In addition to the term learning from demonstration, there are other common terminologies in the literature. Examples are Programming by Demonstration (PbD), Imitation Learning, and Apprenticeship Learning (Billard et al., 2016).

Using LfD, it is necessary to decide how the exemplifications of the task to be automated will be performed. Three different approaches can be chosen:

- **Kinesthetic teaching:** in this method, while the person performing the exemplifications physically moves the manipulator, data for learning is collected from sensors that equip the robot itself (Figure 2.1–a)). Thus, this is a user-friendly method that does not require any equipment other than the arm. Moreover, it does not require transferring movements between different systems because the demonstrations are performed in the robot that will be used in the desired task. On the other hand, a disadvantage of this approach is that the quality of the demonstrations is dependent on the dexterity and smoothness of the agent exemplifying the task (Billard et al., 2016; Ravichandar et al., 2020; H. Zhang et al., 2020).
- **Teleoperation:** here, the agent performing the demonstration sends the commands to the robot from a device (such as the one in Figure 2.1–b)), which can be haptic (as in Evrard et al. (2009)) or combined with virtual reality interfaces (as in T. Zhang et al. (2018)). In teleoperation, the robot collects data from sensors, similarly to kinesthetic teaching, and, as such, there is no need to transfer movements here either. Although, in this method, the exemplification process may not be user-friendly (Ravichandar et al., 2020).

a) b) c)

Figure 2.1 – Different ways of performing the demonstrations: a) Kinesthetic teaching (adapted from Tang (2020)); b) Teleoperation (adapted from Ravichandar et al. (2020)); c) Human movements (adapted from Vogt et al. (2017)).

- **Human movements:** in this approach, a person exemplifies the task that is to be performed, and her movements are captured, for example, by vision-based systems (as in Vogt et al. (2017)) or by motion sensors[3] (as in Edmonds et al. (2017)). Subsequently, it is necessary to understand how the captured movements will be transferred to the robots since they have differences from humans (known as the body correspondence problem). Despite this, the approach allows ample freedom of movement to the agent performing the exemplification. As already mentioned in

[3] There are situations in which it may be useful to use sensors that can also collect force information, such as when one wants to open medicine bottles with child safety mechanisms (Edmonds et al., 2017).

Section 1, this was the reason why this method was selected for this dissertation (Figure 2.1–c)) (Billard et al., 2016).

Two relevant questions in LfD are "What to imitate?" and "How to imitate?" (Billard et al., 2016):

- **"What to imitate":** answering this question consists of understanding which aspects of the demonstrations must be reproduced and which can be safely ignored. For example, when a manipulator grasps an object to transport it to another location, the grasping of the object may have to be done in a specific way, for instance, in the zone of its Center of Gravity (CG), so it does not flip, or it may also make no difference. The distinction between essential and irrelevant aspects can be made in various ways. The simplest is to use statistical methods and consider that what appears consistently over several exemplifications is more critical. In this dissertation, the answer to this question is simple: the trajectories described by the demonstrator's hands are what has to be imitated.
- **"How to imitate":** after determining which aspects of the demonstrations must necessarily be reproduced, it remains to be determined how the robot will imitate these aspects in practice. For this imitation to be successful, generally, it is crucial to ensure the correspondence between the exemplifying and the reproducing agent (robot) at two levels:
    - **Perceptual equivalence:** it is guaranteed if the information necessary to perform the desired task is accessible to both agents. For example, in a later phase of the project in which this dissertation is integrated, a vision system will control if the deformations inflicted in the leather pieces are the desired ones. This information will be processed to correct the robot's trajectory if necessary. The same function is performed by the demonstrator's eyes and brain while performing the task.
    - **Physical equivalence:** here, one wants to determine how the robot will physically perform the exemplified task. In this dissertation, there is a need to follow trajectories in Cartesian space with a 7-DoF S-R-S manipulator. Interestingly, this is not a simple challenge, mainly because the joint movable ranges of these robots are highly restricted to avoid self-collisions (Shimizu et al., 2008). Thus, solving a physical equivalence problem, i.e., figuring out how to command the manipulator so that its end-effector follows the desired trajectory without exceeding joint limits[4], will be a central challenge of

[4] Other constraints, such as singularities, must be considered. However, in this brief introduction to the problem, only the joint limits are highlighted because they represent the main challenge.

this dissertation. The following subchapter will present the existing methods essential to solving this problem, also referred to as a trajectory tracking problem.

2.2 Relevant methods to solve the physical equivalence problem

First, this subchapter subdivides the physical equivalence problem into subproblems and details them. Then, it presents the strategies available in the literature to solve these subproblems. Finally, other methods are presented despite not being essential, because they can enrich this dissertation.

2.2.1 Problem detailing

A manipulator can be represented in configuration space, using the joint values, or in the Cartesian space, using the position and orientation of the end-effector (the position is described by three variables and the orientation by another three). The correspondence between these two spaces gives rise to two classical problems in manipulator kinematics:

- **Forward Kinematics (FK):** one wants to determine the position and orientation, i.e., pose, of the manipulator's end-effector, knowing the values of the joints.
- **Inverse Kinematics (IK):** as the name suggests, this is the inverse problem of forward kinematics. The goal is to determine the values of the joints, knowing the (desired) position and orientation of the manipulator's tip.

In order to solve the trajectory tracking problem in Cartesian space, as presented at the end of the last subchapter, a proper joint motion must be commanded to the manipulator. As such, a solution to the IK problem will be required. This problem is complex because the kinematics equations are nonlinear. In addition to this, while in the FK problem there is always one and only one solution in the inverse kinematics, there may be multiple or no results (Craig, 2014). For an easier understanding of these multiple possible solutions, one could consider that the manipulator to be used has its third joint frozen. By solving the inverse kinematics problem with this restriction, it is possible to conclude that, for a given pose, there can be at most eight possible configurations (Wang et al., 2021). For simplicity, only 4 of these configurations are represented in Figure 2.2, but each could generate another one. This can be accomplished by changing the values of the last three joints according to the following equations, where θ'_i is the new value for the joint i, and θ_i is the original value:

$$\theta'_4 = \theta_4 + \pi, \tag{2.1}$$

$$\theta'_5 = -\theta_5, \tag{2.2}$$

$$\theta'_6 = \theta_6 + \pi. \tag{2.3}$$

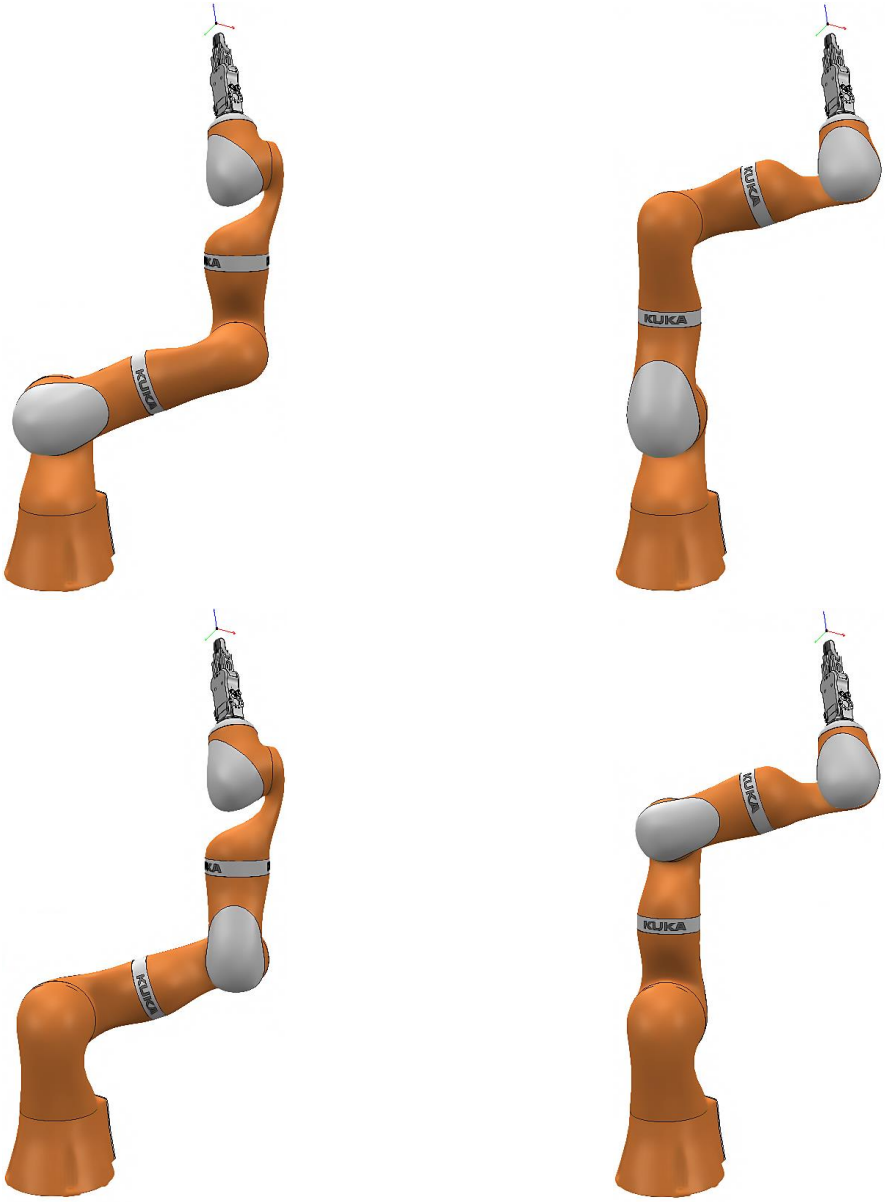


Figure 2.2 – Four solutions of Kuka LBR iiwa for a single pose.

It should be noted that each of these eight configurations is defined by a particular combination in the signals of joints 2, 4, and 6 (Faria et al., 2018; Wang et al., 2021).

Since the manipulator has seven DoF and the target poses are entirely defined by six (three for position and three for orientation), the inverse kinematics problem becomes under-constrained. This means that the manipulator will still have, at most, the eight solutions already presented, but each of them, in turn, will be able to give rise to infinite possibilities. These constitute the so-called null spaces or self-motion

manifolds of the robot, and with this in mind, it makes sense that these manipulators are known as redundant. In practice, a null-space movement consists of the elbow free rotation about a circle[5], whose normal vector is parallel to the axis from shoulder to wrist. In this movement, the end-effector retains its position and orientation (Faria et al., 2018; Silva, 2011; Wiedmeyer et al., 2021). The representation of the null space for the first configuration represented in Figure 2.2 can be seen in Figure 2.3.

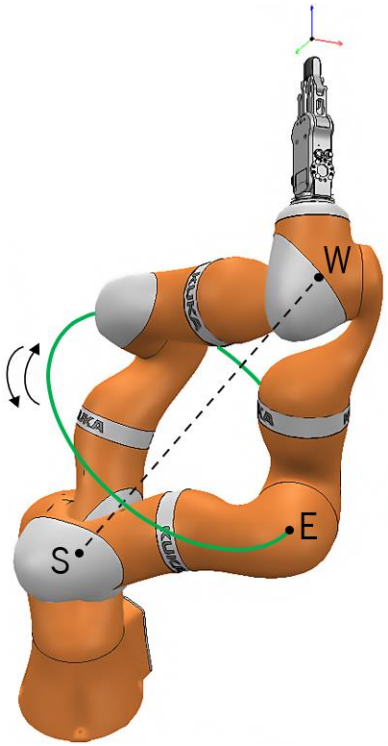


Figure 2.3 – Kuka LBR iiwa redundancy representation (S – Shoulder; E – Elbow; W – Wrist).

Null-space movements are important because they allow the execution of secondary tasks, aside from reaching the desired poses: singularities (Nenchev et al., 2004) and joint limits avoidance (Faria et al., 2018), human-like motions (Kim & Rosen, 2015), collision avoidance (Vu, 2012), and minimization of the arm movement (Moradi & Lee, 2005). As outlined before, avoiding the robot's joint limits will present a critical adversity to the trajectory tracking problem, and, as such, the redundancy must be used to deal with this difficulty. Thus, associated with the algorithm to solve the inverse kinematics problem, a redundancy resolution strategy, i.e., selecting a proper solution from the null space, is needed. In addition to this, a suitable selection, among the eight solutions that at most the inverse kinematics can have, is also necessary. Therefore, works exploring these topics will be analyzed in the following section. However,

[5] This circle is normally known in the literature as Redundancy Circle (RC) (Moradi & Lee, 2005).

before doing so, a quick review of the main types of methods utilized to solve the inverse kinematics problem will be made.

2.2.2 Inverse kinematics and redundancy resolution

There are two major approaches that one can use to obtain a solution to the inverse kinematics problem. These are the position-based and velocity-based approaches. The first option, in turn, can be divided into numerical or closed-form[6] solutions, but only the latter alternative will be considered (Dou et al., 2022). This is because, although they are usually not general, closed-form solutions are faster and readily identify all possibilities. Consequently, numerical solutions are only recommended when there is not or is difficult to find a solution in a closed-form way, which, as it will be seen, is not the case in this dissertation (Waldron & Schmiedeler, 2016). Thus, a decision must be made between velocity-based and closed-form solutions. The first option is generalizable to different kinematic structures and can enforce joint position, velocity, and acceleration limits. Despite this, closed-form solutions present some critical advantages (Faria et al., 2018; Wiedmeyer et al., 2021):

- All solutions can be obtained and in a computationally more efficient way.
- There are no cumulative errors.
- It is easier to map joint limits, and a null-space value can be directly specified. This makes the redundancy resolution easier, regardless of its purpose, and it can be fully analytic.

For the reasons presented above, this dissertation focuses on closed-form solutions for the inverse kinematics problem. It should be noted that approaches considering machine learning algorithms are also presented in the literature (Sariyildiz et al., 2012). Despite this, they will be disregarded because there is no need to teach a system to obtain the solutions that an analytical method can already accurately provide in all circumstances.

Lee and Bejczy (1991) were the first authors to propose a closed-form solution for the inverse kinematics problem of redundant arms. The proposed algorithm parametrized redundant joints, transforming a redundant arm into a parametrized non-redundant version of itself. As such, this algorithm was denominated parameterization method and has reinforced the theory that position-based are superior to velocity-based approaches. Furthermore, the method proposed by Lee and Bejczy (1991) can be applied

[6] As Craig (2014) notes, closed-form "means a solution method based on analytic expressions or on the solution of a polynomial of degree 4 or less, such that noniterative calculations suffice to arrive at a solution" (p.106).

to any kinematic structure. Unfortunately, the quality of the solutions this approach generates is highly dependent on the correct selection of the redundant joints, and it can be argued that it would be difficult to apply this algorithm to joint limit analysis. More recently, Zaplana and Basanez (2018) presented a similar approach to the one proposed by Lee and Bejczy (1991), but this method cannot represent the elbow redundancy in a simple manner.

Kreutz-Delgado et al. (1992) introduced an explicit way to represent the elbow position in the null space/redundancy circle. These authors used an angle, which they called arm angle (represented in Figure 2.4 with the Greek letter ψ), that is defined between two planes:

- **Arm plane:** is spanned by the shoulder, elbow, and wrist of the manipulator, as shown in Figure 2.4 **in blue**.
- **Reference plane:** is defined by a fixed vector (for example, vector $\vec{v}$ in Figure 2.4) and the vector from shoulder to wrist, as shown in Figure 2.4 **in grey**.

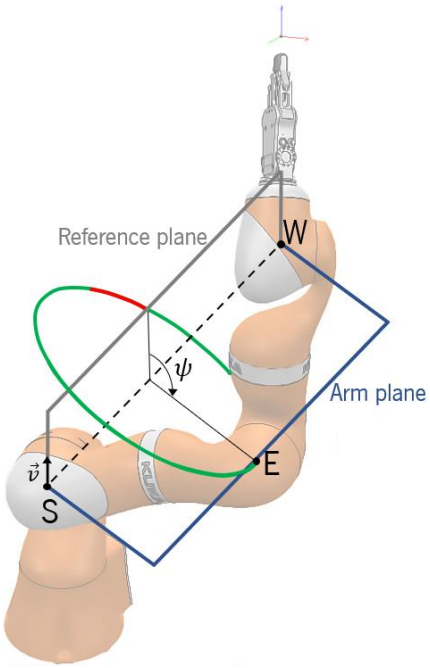


Figure 2.4 – Arm angle representation (ψ) and joint limits mapping into the redundancy circle.

Despite the usefulness of this representation, the authors warn that there are situations in which it can fail. The most important happens when the vectors used to define the reference plane are collinear, making this plane undefined (algorithmic singularity).

Dahm and Joublin (1997) took two crucial first steps toward solving the inverse kinematics and redundancy problem. Firstly, these authors geometrically solved the inverse kinematics considering an elbow redundancy parameter. Secondly, they presented an initial approach to map the joint limits into

the redundancy circle. In practice, this means that, as depicted in Figure 2.4, one can know which zones of the redundancy circle put the joints inside (represented with the **green** color) and outside their limits (represented with the **red** color). Despite this, the inverse kinematics algorithm does not consider the different solutions, and the joint limits mapping is only achieved for one joint. Moradi and Lee (2005) made minor corrections to the inverse kinematics algorithm proposed by Dahm and Joublin (1997) and introduced a more extensive method to map joint limits into de redundancy circle. These authors also presented a redundancy resolution approach in which the arm movement was reduced with the minimization of the elbow position variation. Nevertheless, as the work of Dahm and Joublin (1997), this paper does not consider the different inverse kinematics solutions.

Shimizu et al. (2008) presented a solution to the previously mentioned situation in which the arm angle presented by Kreutz-Delgado et al. (1992) became undefined. To achieve this, they proposed that the reference plane stopped being spanned by a fixed vector and the vector from shoulder to wrist, starting being defined by the SEW (Shoulder–Elbow–Wrist) plane of a virtual manipulator, refered as SE_VW. The virtual manipulator is a non-redundant image of the original arm (has its third joint always in its zero position)–Figure 2.5.

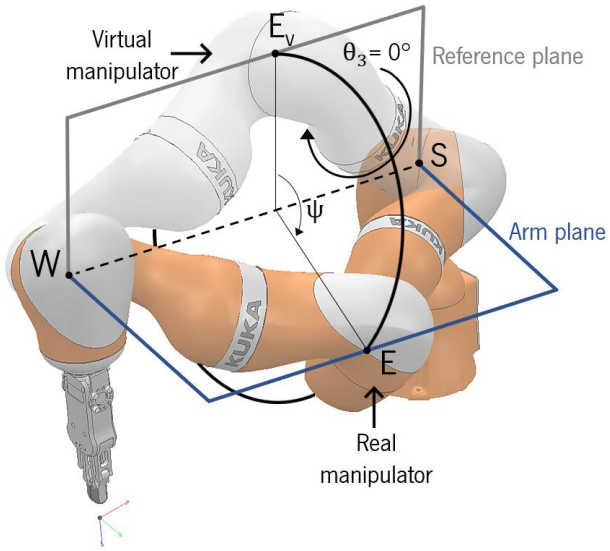


Figure 2.5 – Arm angle (ψ) representation with the reference plane being given by a virtual manipulator, as proposed by Shimizu et al. (2008).

Moreover, these authors presented a new method to solve the inverse kinematics problem, and they derived expressions that allow mapping not only joint limits but also singularities into the redundancy circle. Finally, a redundancy resolution method is presented to increase the distance to joint limits. Despite being an extensive and detailed work, which considers the main problems of this dissertation, the paper of Shimizu et al. (2008) has the same problem as others already presented: it does not consider the multiple inverse kinematics solutions. Furthermore, when the root of the wrist (intersection of the last

three joints' axes of the manipulator) is in the direction of the joint 1 axis, the angular value for this joint becomes indeterminate at an early stage of the algorithm. Moreover, this condition can lead to abrupt variations in trajectory tracking. To solve the first problem, the authors use a convention and set the value of the first joint to zero, but the second was not analyzed.

Xu et al., 2016 agreed that the alternative way of defining the arm angle proposed by Shimizu et al. (2008) could deal with the algorithmic singularity presented in the Kreutz-Delgado et al. (1992) approach. Despite this, they pointed out that the reference plane used by Shimizu et al. (2008) is not absolute, i.e., it varies between different solutions of the inverse kinematics, making the approach neither user-friendly nor convenient for practical applications. Therefore, Xu et al., 2016 proposed a method that uses two arm angles. These are defined similarly to what was seen in the Kreutz-Delgado et al. (1992) approach but in a twofold way. Two fixed vectors and the vector from shoulder to the wrist are used to spawn two reference planes, which define the two arm angles with the SEW plane. As the vector from shoulder to the wrist cannot be collinear simultaneously with the two fixed vectors, at least one reference plane will always be defined. Thus, the algorithmic singularity is avoided, and both reference planes are absolute. These authors also present a method for solving the inverse kinematics and consider the eight different solutions that exist at most. As previously presented, these eight solutions can be defined by the combinations of signals used in joints 2, 4, and 6. Thus, these authors suggest placing three configuration flags in these three joints to make the selection among the various possible solutions. Kuhlemann et al. (2016) also solved the inverse kinematics problem considering an elbow redundancy parameter and configuration flags. Despite everything, in both works, joint limits and kinematic singularities were not considered, and a redundancy resolution approach was not proposed as in Shimizu et al. (2008).

Oh et al. (2017) defined the arm angle similarly to Kreutz-Delgado et al. (1992) and used the Shimizu et al. (2008) approach to solve the inverse kinematics problem and map joint limits into the redundancy circle. Moreover, they determined the feasible arm angle range to avoid self-collisions and presented a redundancy resolution strategy. Despite this, it is not adequate for a trajectory tracking problem because it does not ensure a continuous arm angle variation.

The work proposed by Faria et al. (2018) brought two significant developments, which made this the state-of-the-art approach to solving the inverse kinematics and redundancy problem (Wang et al., 2021; Wiedmeyer et al., 2021):

- These authors used configuration flags to make the inverse kinematics algorithm proposed by Shimizu et al. (2008) return the eight possible solutions (at most). For these solutions, they also used the strategy of Shimizu et al. (2008) to map joint limits and singularities to the redundancy

circle. In this particular work, the authors call the flags Global Configuration parameter and divide it into three variables, GC_2, GC_4, and GC_6, which are the flags for the shoulder, elbow, and wrist, respectively.

- An analytical redundancy resolution approach was proposed, allowing joint limits and singularities avoidance, and real-time performance. To better understand this method, one should consider Figure 2.6. In this figure, it is possible to see the arm angle feasible intervals (represented in grey) for two consecutive poses of a given trajectory (to a specific solution of the inverse kinematics). The chosen arm angle for pose 1 was ψ_{t-1} and the arm angle for pose 2 is to be determined (ψ_t). Here, the method proposed by Faria et al. (2018) first looks for which interval in the current pose, pose 2, contains the arm angle of the previous pose (ψ_{t-1}), pose 1. In this case, this interval would be the second (marked in **green**). From this interval and with ψ_{t-1}, it is possible to determine ψ_t from an equation with two parameters that can be tuned. The first controls the strength of repulsion of the arm angle from the limits of the feasible interval. The second controls the distance to the limits from where the repulsion starts. Thus, the second parameter controls the tendency of the arm angle to move towards the interval center or to preserve its position, as long as it is not overly close to limits. In conclusion, the next arm angle is calculated depending on the previous, originating smooth transitions between poses and avoiding joint limits and singularities.

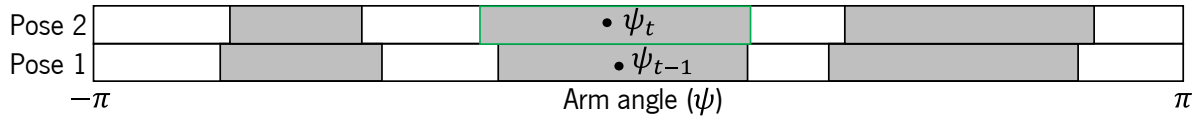


Figure 2.6 – Redundancy resolution approach proposed by Faria et al. (2018).

Another great advantage of the article proposed by Faria et al. (2018) is that the authors make the approach implementation in Matlab available, which speeds up the understanding of the code, the implementation of changes, and the testing phase (Faria, 2021). Regardless of this, some problems still arise:

- The redundancy resolution method does not consider the continuity of the arm angle at $-\pi/\pi$, i.e., it does not consider that at one instant of time one can have a value slightly below π, and at the next instant a value slightly above $-\pi$ (Figure 2.7), or vice versa.

- The resolution of the singularity that happens when the root of the wrist lies in the direction of the joint one axis is solved identically to Shimizu et al. (2008), and the problems that can arise from here related to trajectory tracking are also not analyzed.
- The method proposed by Faria et al. (2018) does not suggest any way to choose one of the several solutions available in inverse kinematics or to select the arm angle to be used in the first pose of a trajectory.

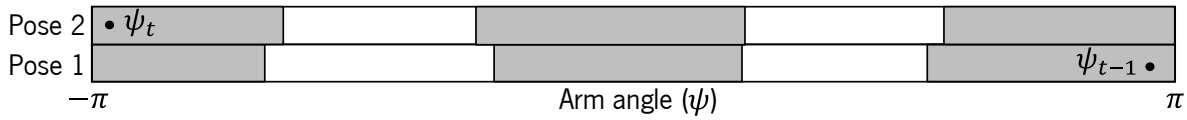


Figure 2.7 – Continuity of the arm angle in $-\pi/\pi$ in redundancy resolution.

Gong et al. (2019) proposed an alternative way to solve the inverse kinematics problem, also using configuration flags and defining the arm angle similarly to Shimizu et al. (2008). They discussed the situations in which the configuration flags could be switched in the middle of a movement without causing abrupt changes in the joint values. In addition to this, these authors also mapped into the redundancy circle joint limits and collisions with obstacles. Nevertheless, it can be argued that this approach will not be as suitable for trajectory tracking as the one of Faria et al. (2018) because singularities are not taken into account in such a detailed way, and a suitable redundancy resolution approach is not suggested.

Tian et al. (2021) solved the inverse kinematics problem similarly to Faria et al. (2018). Although, they defined the reference plane used to define the arm angle through the vectors from base to shoulder and from shoulder to wrist. As Gong et al. (2019), these authors discuss the situations in which it is possible to switch between solutions of the inverse kinematics, in the middle of a movement, without abrupt changes in the joint values. To solve the redundancy problem, an iterative approach was proposed that optimizes a cost function, resulting in the arm angle that maximizes the avoidance of joint limits.

Wiedmeyer et al. (2021) used the algorithm of Faria et al. (2018) to solve the inverse kinematics problem and map joint limits and singularities to the redundancy circle. Despite this, they proposed a new method for solving the redundancy problem but considered more than one objective. In addition to avoiding joint limits and singularities, higher end-effector velocities and smoother trajectory execution were sought and achieved. As the approach of Faria et al. (2018), the strategy of these authors can also work in real-time, and its implementation in C++ is public too (Wiedmeyer, 2020). Regardless, it is not as user-friendly as the implementation of Faria et al. (2018).

One of the goals of this dissertation is to create a system that is prepared to be subjected to online corrections. To achieve this, the robot must have the capability to change the position and orientation of

the end-effector arbitrarily. This requirement is usually evaluated in the literature by the concept of manipulability (Yoshikawa, 1985), which is negatively affected by the proximity to joint limits and singularities (Vahrenkamp et al., 2012). For this reason, Wiedmeyer et al. (2021) highlight that the middles of the feasible arm angle intervals, for each pose, increase manipulability. In this approach, two aspects must be taken carefully. The first is that the feasible intervals do not consider the fourth joint because it is independent of the arm angle. Thus, singularities and joint limits related to it will not be considered. The second is that the arm angle is an abstraction for six joints and, as such, may sometimes not represent the real proximity to the boundaries of all joints or singularities. Nevertheless, this approach is suitable to increase the manipulability in real-time performance because the absolute value of the fourth joint will be explicitly defined by the pose[7], and, as such, one can not change it, i.e., it can not be used to influence the manipulability.

In cases where there is the possibility of changing the poses values, the fourth joint angle can already be exploited and, as such, can be used to increase the manipulability. More than this, the pose with larger feasible intervals can be searched. Such an approach is presented in the doctoral thesis Faria (2018). In this work, the author was able to change the orientation of a pose and used this possibility to obtain the one with the highest manipulability. Despite this, the distance to joint limits in the fourth joint was not considered, only its singularity.

Faria et al. (2018) detected singularities by comparing a discriminant with zero through a threshold. Wang et al. (2021) argued that this is not a good practice because the threshold will be inconsistent across different poses, which will lead to wrong feasible arm angle intervals. To solve this numerical issue, the authors clearly defined that a 7-DoF S-R-S manipulator is in a singularity when joint 2 or 6 is at 0° or ±180° (in this dissertation, only the first case is possible because the used arm's joints do not have a maximum angular limit of ±180°). From this point and with the expressions that come from the works of Shimizu et al. (2008) and Faria et al. (2018), Wang et al. (2021) were able to detect singularities in a numerically stable way.

Dou et al. (2022) proposed a strategy to solve the redundancy problem similar to Faria et al. (2018). Still, they simplified the calculations of the equation presented in the latter paper. Despite this, they continued to use two equivalent tunable parameters and the arm angle from the previous pose. Furthermore, these authors divided the feasible intervals into "working areas" and "dangerous areas" to increase safety.

[7] It will be seen in Section 3.2.1 that the joint 4 will have the same distance to joint limits and singularities independently of the arm angle and inverse kinematics solution.

Finally, the introduced redundancy resolution strategy was used to successfully solve a trajectory tracking problem.

Works as Asfour and Dillmann (2003), Tondu (2006), and Kim and Rosen (2015) are not explored in this literature review because they focus on human likeness[8] and, as such, are difficult to use in general-purpose approaches. Furthermore, this section mainly presented closed-form inverse kinematics algorithms and redundancy resolution strategies of 7-DoF S-R-S manipulators. Still, specific trajectory tracking strategies can contribute to the success of this work. As such, the next section will expose some of these strategies and works that use them.

2.2.3 Complementary strategies

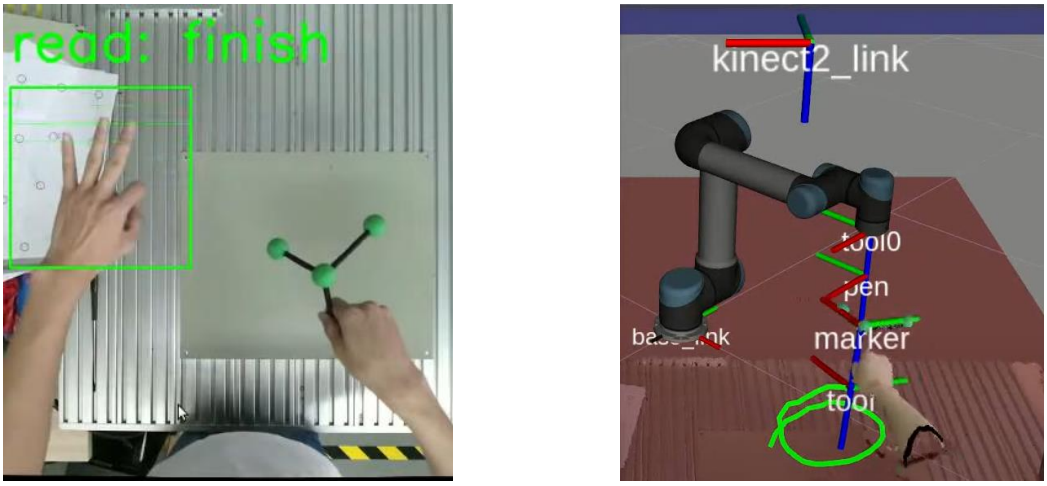
Despite the importance of inverse kinematics algorithms and the strategies for redundancy resolution, other aspects have been highlighted in the literature that can add value to this dissertation. Gutzeit et al. (2018) pointed out that when one wants to solve a trajectory tracking problem, like the one faced in this dissertation, the workspace zone where the manipulator will describe the trajectory can vary. By following the trajectory, the effects on the material to be inspected will be the same. Still, there will be restrictions associated with the passive manipulator reachable zone and the areas that the inspection camera will capture. Despite this, they will never be highly limiting because the passive manipulator reachable zone can be, at least, equal to that of the active one. Moreover, there may be several cameras or only one camera attached to the passive manipulator. Considering that this dissertation pretends to solve a trajectory tracking problem, increasing the manipulability of the arm in each pose, the place in which these requirements are fulfilled should be chosen.

One of the most significant advantages of learning from demonstration is that it allows people not specialized in robotics to program a robot. Thus, a system that works by LfD should have a set of interfaces with the user, which allow more than just collecting trajectories. Nowadays, these interfaces enable interactions such as:

- Command and collect information about the manipulator intuitively through a Human Machine Interface (HMI)/Graphical User Interface (GUI).
- Perform commands through gestures (Figure 2.8 – a)).

[8] Despite not being in the scope of this thesis, the human-like movements are of great importance in taks that envolve human-robot collaboration, because they make the movements more predictable and enhance the collaboration (Garcia et al., 2019).

- Visualize acquired trajectories and simulations in Augmented Reality (AR) (Figure 2.8 – b)). Also, in AR, obstacles can be designed online with gestures to be included in simulations, and trajectories can be modified.



a) b)

Figure 2.8 – User-friendly interfaces used in LfD: a) Commands through gestures; b) Trajectory capturing with online representation in an AR environment (adapted from Zhang (2019)).

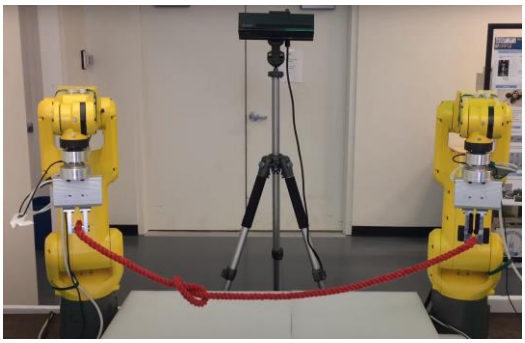
A recent and exciting work that addresses these topics is Zhang et al. (2020).

2.3 Generalization

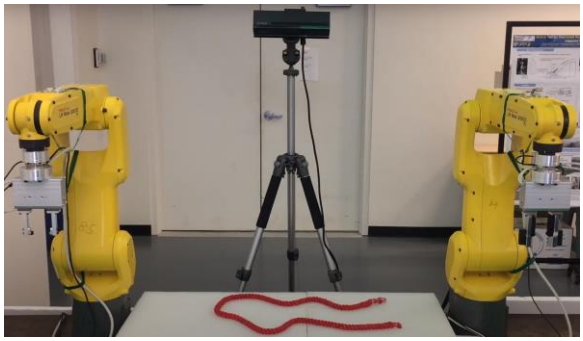
Although LfD takes a task demonstration as a starting point, it is not easy to exemplify all the conditions under which the desired task can be performed. Imagine, for example, the case in which one wants to tie a knot in a rope with a bimanual robotic system (Figure 2.9 – a)). In the situation in which the desired task is to be performed (Figure 2.9 – b)), no matter how many exemplifications are performed, the rope can always be in different initial configurations and poses (Figure 2.9 – c)), or ropes of different sizes may appear. These modifications imply changes in the way the task is performed. Thus, it is considered that the generalization/adaptation ability is important in LfD. Deep down, it will distinguish systems that can learn a task from the systems that can only reproduce demonstrations (Ravichandar et al., 2020). In the case of deformable materials, when LfD is used, the adaptability of systems has been achieved based on non-rigid registration methods[9]. These methods allow transforming a point cloud into another, and, to do so, they solve two problems (Tang et al., 2016):

[9] Examples of methods used in task modeling and generalization, outside the scope of deformable object manipulation, are: Hidden Markov Models (HMMs), Gaussian Mixture Models (GMMs), and Dynamic Movement Primitives (DMPs) (Billard et al., 2016; Zhu & Hu, 2018). More information about these methods can be found in Chernova and Thomaz (2014) and Xie et al. (2020).

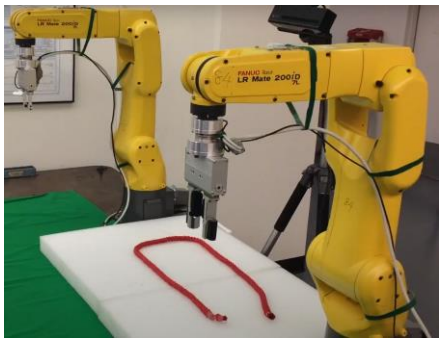
1. Find the correspondence between the points of the two clouds.
2. Determine a transformation function that minimizes the distance between the previously considered correspondent points when applied to the cloud to be transformed.



a)



b)



c)

Figure 2.9 – Bimanual manipulation of linear deformable objects: a) Two manipulators finish the task of tying a knot; b) Initial configuration of a rope for which the system is intended to perform the task; c) Initial configuration of a rope in the exemplification performed, that only approximates the configuration in which the task is intended to be performed (adapted from Tang (2020)).

This type of method is called "non–rigid" because to transform a point cloud into another, the first is "warped", not only translations or rotations are applied (Crum et al., 2004). This aspect can be seen in Figure 2.10, particularly in the grids of a) and b), as they allow to infer the effect of the “warp” function needed to transform the point cloud of a) (which corresponds to the rope configuration presented in Figure 2.10 – c)) into the one of b) (which corresponds to the configuration presented in Figure 2.10 – d)). Some examples of non-rigid registration methods are the algorithms: Thin Plate Spline-Robust Point

Matching (TPS–RPM) (Chui & Rangarajan, 2003); Tangent Space Mapping–Robust Point Matching (TSM–RPM) (Tang et al., 2016); and Coherent Point Drift (CPD) (Myronenko & Song, 2010).

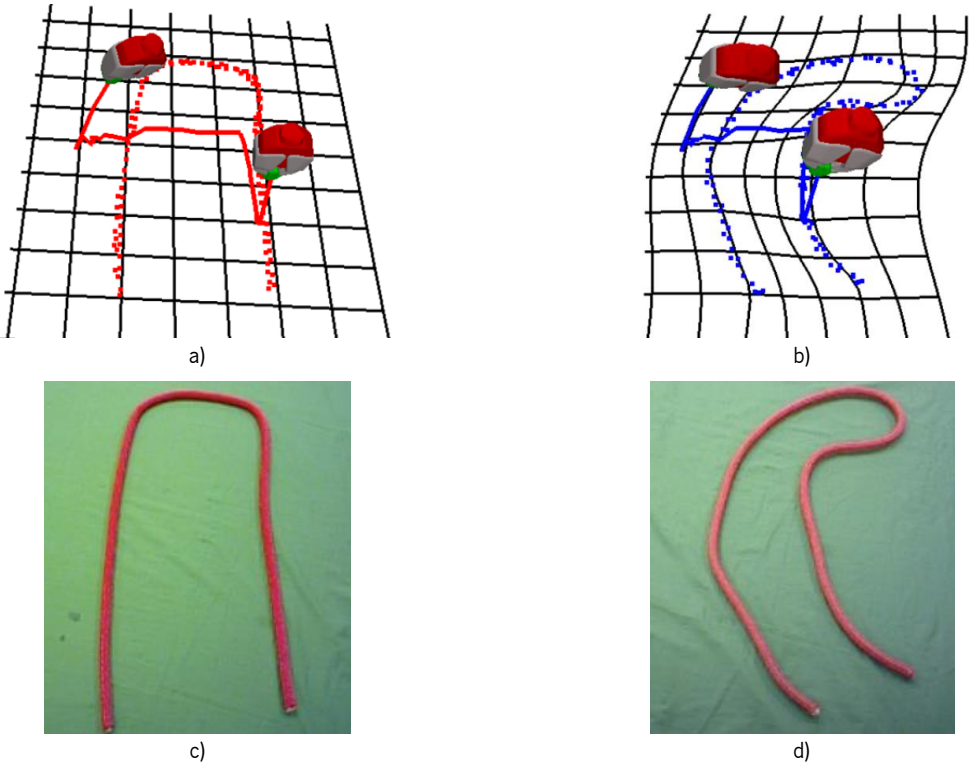


Figure 2.10 – Generalization of trajectories, in the manipulation of deformable linear objects, based on non-rigid registration methods: a) Point cloud corresponding to the configuration of the rope present in c) and respective manipulation trajectory, obtained from an exemplification; b) Point cloud corresponding to the configuration of the rope present in d) and respective manipulation trajectory, generalized from the trajectory present in a); c) Initial configuration of the rope before performing the exemplification that provides the trajectory present in a); d) Initial configuration of the rope when the desired task is to be performed (adapted from Schulman et al. (2016)).

Schulman et al. (2016) were the first to use non-rigid registration methods to generalize exemplified trajectories. They studied tasks like knot tying and folding clothes, in which the initial pose, configuration, and size of the object may change between the exemplification and reproduction situations. To adapt the trajectories between these two scenarios, these authors assumed that the function that transforms the objects point clouds from one situation to another, is the same that will allow the adaptation of the demonstrated trajectories to the new case. Thus, these authors use a non-rigid registration method (TPS-RPM) in order to determine the necessary transformation function ($\mathbb{R}^3 \rightarrow \mathbb{R}^3$), and then apply this function to the trajectory collected in the exemplification, obtaining a new one, suited for the new initial situation. For this reason, the authors have called the method trajectory transfer. In the case of Figure 2.10 the transformation function obtained allows to "warp" the point cloud of a) into the one of b), and applying this function to the exemplified trajectory, present in a), one obtains its generalization, shown in b).

Still in the work of Schulman et al. (2016) it is also relevant to highlight the following three points:

- In tasks that involve gripping and releasing the object several times, such as knot tying and folding clothes, the trajectory transfer method is applied several times to each of the individual trajectories between gripping and releasing the object.
- The results will be better if there are several exemplifications for each task/task step. This is because one can choose the exemplification whose initial state has more similarities to the current situation, considering the configuration, pose, and object size.
- The exemplification of trajectories that allow changing the object state in situations from which one will not be able to finish the task allows reducing the number of times the method fails.

After the work done by Schulman et al. (2016), several others appeared with the same scope (generalization of trajectories in the manipulation of deformable objects, using non-rigid registration methods), bringing different improvements. Three of the most recent of these works are Huang et al. (2015)[10], Tang et al. (2016), and Tang et al. (2018). Huang et al. (2015) improved the correspondence between the point clouds of the exemplification and reproduction situations. To do this, they identified specific zones of ropes and towels, using convolutional neural networks[11], and incorporated the information from there into the TPS-RPM algorithm used by Schulman et al. (2016). Tang et al. (2016) proposed an alternative method, the TSM-RPM, which has application in tangent space (see Figure 2.11 (a) and (b)), whereas TPS-RPM had in Cartesian space.

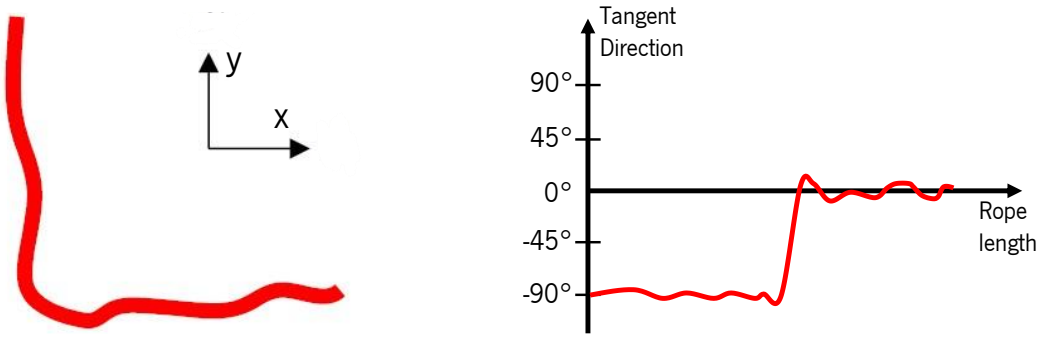


Figure 2.11 – Tangent space used in the application of the TSM-RPM algorithm, proposed by Tang et al. (2016): a) Schematic representation of a rope arranged on a surface; b) Representation of the rope presented in a) in the tangent space: the horizontal axis represents the length of the rope, and the vertical axis represents the direction of a unit vector tangent to it (adapted from Tang et al. (2016)).

[10] Although this paper is indicated as more recent than Schulman et al. (2016), it is not, because the latter was first published in 2013.

[11] Convolutional neural networks allow, among others, to solve complex problems in computer vision, particularly in object recognition (Rivas, 2020). In the scope of this thesis, this method can be considered as a black box, but detailed information can be found in Nielsen (2015) and Aggarwal (2018).

This new method achieves better results, especially in obtaining specific rope configurations and preventing overstretching. Tang et al. (2018), in place of the TPS-RPM non-rigid registration method, used CPD, achieving greater robustness in situations where occlusions exist.

2.4 Chapter conclusions

From this chapter, it was possible to draw a set of conclusions from which the following are highlighted:

1. Within the scope of LfD, this dissertation will focus on the physical equivalence problem. Solving this problem using a 7-DoF S-R-S manipulator is not straightforward because this type of robot has its joint movable ranges severely limited to avoid self-collisions. Having this problem solved, one will be able to manipulate flexible objects in the same conditions in which the demonstrations were done.
2. To solve the physical equivalence problem, it is necessary to use an inverse kinematics algorithm and a strategy to solve the redundancy of the manipulator. The inverse kinematics solving algorithm must be position-based and closed-form. The redundancy resolution strategy should have the avoidance of joint limits as its central focus.
3. Several works have been dedicated to at least one of these points or aspects related to them. However, the work of Faria et al. (2018) is currently considered the state-of-the-art approach (Wang et al., 2021; Wiedmeyer et al., 2021) for solving both, and as such, will be used as a basis in this dissertation. Furthermore, these authors provide the implementation of the proposed solution in Matlab, which will increase the solution prototyping speed.
4. Some works have introduced improvements in the Faria et al. (2018) approach. Despite this, only the changes proposed by Wang et al. (2021) are considered essential to this dissertation. The reason is that they are related to singularities and may compromise the safety of the movements. The main changes presented in the other works (e.g., Dou et al. (2022) and Wiedmeyer et al. (2021)) will not be considered, but their implementation in future work will remain possible.
5. These works (Faria et al. (2018) and Wang et al. (2021)) would probably be enough to solve the trajectory tracking problem in most cases and include means to increase the manipulability[12], but

[12] It possible to determine the arm angle feasible intervals and one can choose the value for the latter that is farther from the limits, in which there will be a joint limit or singularity; The different IK solutions can be determined and, as such, one can choose the most convinent one, for instance, the one with the wider feasible intervals.

they are not focused on this subject at the level this work needs. Thus, they can be used as a base, but the manipulability aspects have to be considered with more attention giving rise to a trajectory tracking method with increased manipulability. This will allow the implementation of an online correction system in the future. To achieve this, a method that uses the redundancy, the different places where the trajectory can be tracked, and the IK's various solutions to increase the distance to joint limits and singularities must be developed.

6. To generalize the trajectories of a person's hands, one will probably need to use some non–rigid registration method. More than this, and despite the evident importance of the generalization capabilities to the LfD, first, a method to follow trajectories in the Cartesian space robustly has to be obtained.

With these conclusions drawn, the next chapter, i.e., Chapter 3, explores the points outlined here related to the arm's kinematics, such as the IK and the mapping of joint limits and singularities into the redundancy circle. Then, in Chapter 4, the developed trajectory tracking method with increased manipulability is presented.

3. KINEMATICS OF A 7–DOF SERIAL MANIPULATOR

This chapter uses some of the approaches presented in the previous section (mainly the ones of Faria et al. (2018) and Wang et al. (2021)) to solve the kinematics problems that will arise in the following: forward kinematics, inverse kinematics, and joint limits and singularities mapping into the redundancy circle.

3.1 Forward kinematics

This subchapter first explains the fundamental ideas behind a typical resolution of a FK problem and then solves it for a 7–DoF serial manipulator. Some of these aspects will be useful throughout the whole chapter. Recalling the definition presented in subchapter 2.2.1, in FK one wants to determine the position and orientation, i.e., pose, of the manipulator's end-effector, knowing the values of the joints. Here, for a set of joint values, there always exists a solution for the end-effector pose, and that solution is unique. It should be noted that, since the manipulator to be used has seven DoF, the determination of the elbow position in the resolution of the FK problem can also be useful.

3.1.1 Forward kinematics fundamental aspects

Typically, the following strategy is used to obtain the pose of the end-effector, knowing the angular positions of the joints[13]:

1. A coordinate frame is attached to the base of the arm and other to its end-effector.
2. Since the length of the links and the joint rotations are known, one can describe the transformation between the base coordinate frame and the end-effector one using transformation matrices.
3. The obtained transformation matrix will contain information about the position and orientation of the end-effector relative to the base coordinate frame. As such, a conversion from the joint space to the cartesian space is achieved.

[13] This subchapter is based on Craig (2014), McKerrow (1991), and Siciliano et al. (2009).

As both position and orientation information in three-dimensional space is relevant, 4x4 transformation matrices are used, which combine translations and rotations (homogeneous transformation matrices). These are given in the following form:

$$T = \left[\begin{array}{ccc|c} & rotation & & translation \\ \hline 0 & 0 & 0 & 1 \end{array}\right], \tag{3.1}$$

where a general translation by a vector $p_x\hat{x} + p_y\hat{y} + p_z\hat{z}$ is given by:

$$Trans(p_x, p_y, p_z) = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.2}$$

and the rotations about x, y, and z by an angle ϕ:

$$Rot(x, \phi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) & 0 \\ 0 & sin(\phi) & cos(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.3}$$

$$Rot(y, \phi) = \begin{bmatrix} cos(\phi) & 0 & sin(\phi) & 0 \\ 0 & 1 & 0 & 0 \\ -sin(\phi) & 0 & cos(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.4}$$

$$Rot(z, \phi) = \begin{bmatrix} cos(\phi) & -sin(\phi) & 0 & 0 \\ sin(\phi) & cos(\phi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.5}$$

It should be noted that rotations about a general axis will not be needed to solve the direct kinematics problem. To better understand what rotation matrices represent and how they are obtained, one could consider only the first three rows and columns of the matrix 3.3:

$$Rot(x, \phi)[1{:}3, 1{:}3] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi) & cos(\phi) \end{bmatrix}. \tag{3.6}$$

In the matrix presented above (3.6), column 1 has in each row, respectively, the x, y, and z projections of the x-axis of the frame resulting from the given rotation (Figure 3.1: frame $\hat{x}^1, \hat{y}^1, \hat{z}^1$) in the reference frame (Figure 3.1: frame x^0, y^0, z^0). The second and third columns correspond, respectively, to the y and z-axis projections. Thus, each column represents a unit vector that corresponds to an axis of the new frame in relation to the reference one (Figure 3.1). Moreover, these rotation matrices have the following interesting properties:

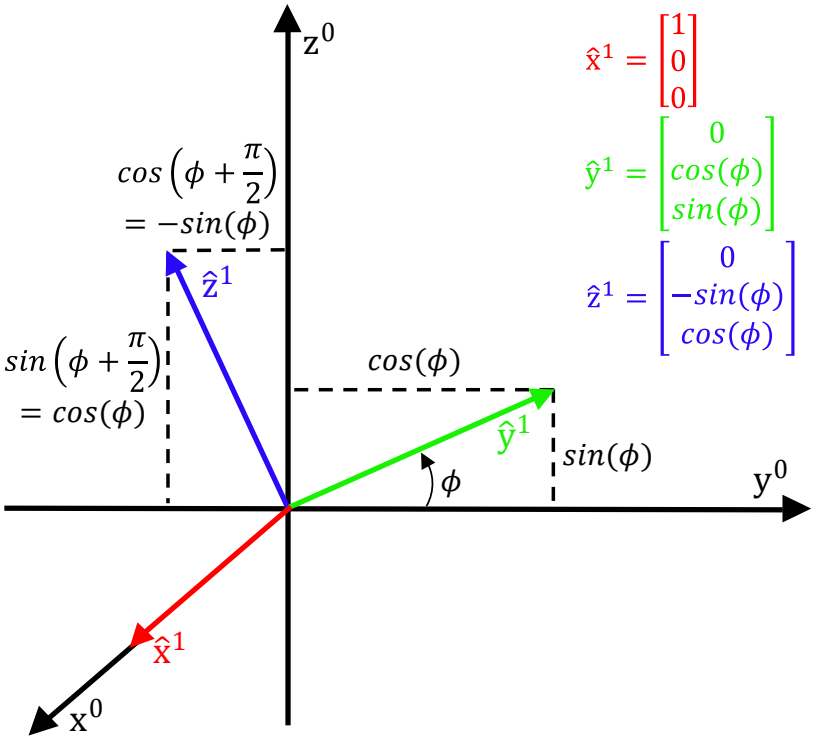


Figure 3.1 – Orientation of frame 1 relative to frame 0.

- The inverse matrix is equal to its transpose because the rotation matrices have orthonormal columns. As such, if ${}^{i}R_{j}$ is a rotation matrix which describes the rotation that transforms the frame i into j:

$$({}^{i}R_{j})^{-1} = ({}^{i}R_{j})^{T}. \tag{3.7}$$

- Rotation matrices can be "chained together" (composition of rotation matrices), as in the following example:

$${}^{0}R_{3} = {}^{0}R_{1}\,{}^{1}R_{2}\,{}^{2}R_{3}. \tag{3.8}$$

- Being ${}^{j}q$ a coordinate vector represented in frame j, if one applies to it the rotations that are needed to transform the frame i into j (${}^{i}R_{j}$) the coordinates of the vector q relative to the i frame will be obtained (${}^{i}q$) (because it will be as if the frame i and the vector q suffered the same rotation, so the relative coordinates do not change):

$${}^{i}q = {}^{i}R_{j}\,{}^{j}q. \tag{3.9}$$

Returning to the homogeneous transformation matrix (3.1), it is now possible to understand better how the rotation and translation terms relate, why these matrices can also be "chained together", and how they represent the position and orientation of a frame relative to another. With this purpose, one could consider two homogenous transformation matrices: ${}^{0}T_{1}$ and ${}^{1}T_{2}$, which describe the transformation from frame 0 to 1 and from 1 to 2, respectively. Considering a symbolic representation it can be written:

$$ {}^{0}T_{1} = \left[\begin{array}{ccc|c} & {}^{0}R_{1} & & {}^{0}p_{01} \\ \hline 0 & 0 & 0 & 1 \end{array}\right], \tag{3.10} $$

$$ {}^{1}T_{2} = \left[\begin{array}{ccc|c} & {}^{1}R_{2} & & {}^{1}p_{12} \\ \hline 0 & 0 & 0 & 1 \end{array}\right], \tag{3.11} $$

where ${}^{i}p_{jk}$ is the vector that goes from the origin of frame j to the origin of frame k, with coordinates relative to frame i. Maintaining the symbolic representation, the multiplication between the two matrices can be presented as:

$$ {}^{0}T_{1}\,{}^{1}T_{2} = \left[\begin{array}{ccc|c} & {}^{0}R_{1}\,{}^{1}R_{2} & & {}^{0}p_{01} + {}^{0}R_{1}\,{}^{1}p_{12} \\ \hline 0 & 0 & 0 & 1 \end{array}\right]. \tag{3.12} $$

From the matrix 3.12 is possible to draw three conclusions:

1. The overall constitution of the matrix remains the same: the first three rows and columns keep having information about orientation (rotation matrix); the last column in the first three rows continues to have position information; the last row remains the vector [0 0 0 1]. Thus, the multiplication between two consecutive homogenous transformation matrices is a homogenous matrix.
2. In the first three rows and columns will occur the multiplication of rotation matrices. As such, these terms will comply with the chaining of transformations, and the obtained rotation matrix will give the orientation of a frame relative to another. In this case, the orientation of frame 2 relative to frame 0.
3. The equation that gives rise to the first three rows of the last column (${}^{0}p_{01} + {}^{0}R_{1}\,{}^{1}p_{12}$) allows one to obtain the position of frame 2 relative to the base (frame 0). This is because ${}^{0}R_{1}\,{}^{1}p_{12}$, as shown in the equation 3.9, is equal to ${}^{0}p_{12}$, and ${}^{0}p_{01} + {}^{0}p_{12}$ gives ${}^{0}p_{02}$ (which are the coordinates of frame 2 origin relative to the base). Thus, also the translation terms of the homogenous transformation matrix comply with the chaining of transformations, and the obtained position vector will represent coordinates of a frame's origin relative to the base.

With this conclusion drawn, the matrix 3.12 can be rewritten as:

$$ {}^{0}T_1\,{}^{1}T_2 = \left[\begin{array}{ccc|c} & {}^{0}R_2 & & {}^{0}p_{02} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] = {}^{0}T_2. \tag{3.13} $$

Now that it is possible to understand better how the homogeneous transformations work, a simple example of the three-step procedure presented at the beginning of this subchapter will be presented. This example will consider a simple 2-DoF planar manipulator, like the one presented in Figure 3.2.

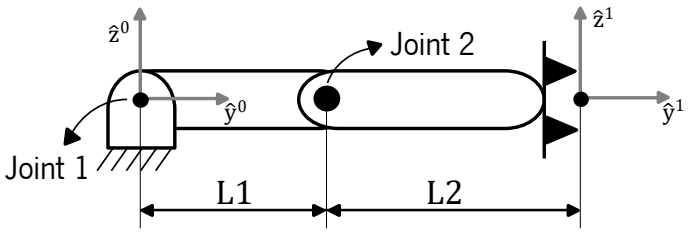


Figure 3.2 – Schematic representation of a 2-DoF manipulator with a base and end-effector frame. L1 and L2 are the lengths of links 1 and 2, respectively.

Considering now that joints 1 and 2 are submitted to a rotation θ_1 and θ_2, respectively, the arm is positioned in the configuration presented in Figure 3.3.

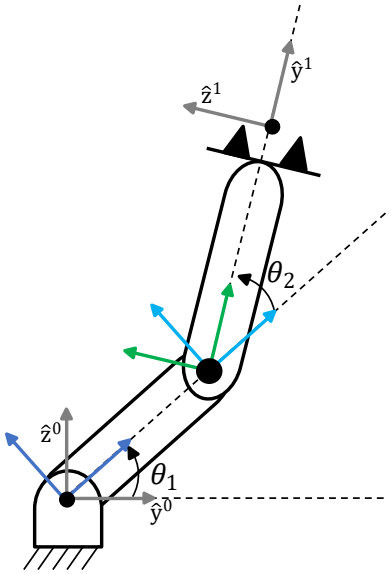


Figure 3.3 – Schematic representation of a 2-DoF manipulator with a θ_1 and θ_2 rotations in joints 1 and 2, respectively.

The position and orientation of frame 1 relatively to frame 0 can be obtained by (colors of equation 3.14 should be related to the frames of Figure 3.3):

$$ {}^{0}T_1 = Rot(x,\theta_1)Trans(0,L1,0)Rot(x,\theta_2)Trans(0,L2,0), \tag{3.14} $$

which allows to obtain,

$$ {}^{0}T_{1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\theta_1+\theta_2) & -sin(\theta_1+\theta_2) & L2cos(\theta_1+\theta_2)+L1cos(\theta_1) \\ 0 & sin(\theta_1+\theta_2) & cos(\theta_1+\theta_2) & L2sin(\theta_1+\theta_2)+L1sin(\theta_1) \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (3.15) $$

The latter can easily be verified by the projections represented in Figure 3.4 (and the colors related).

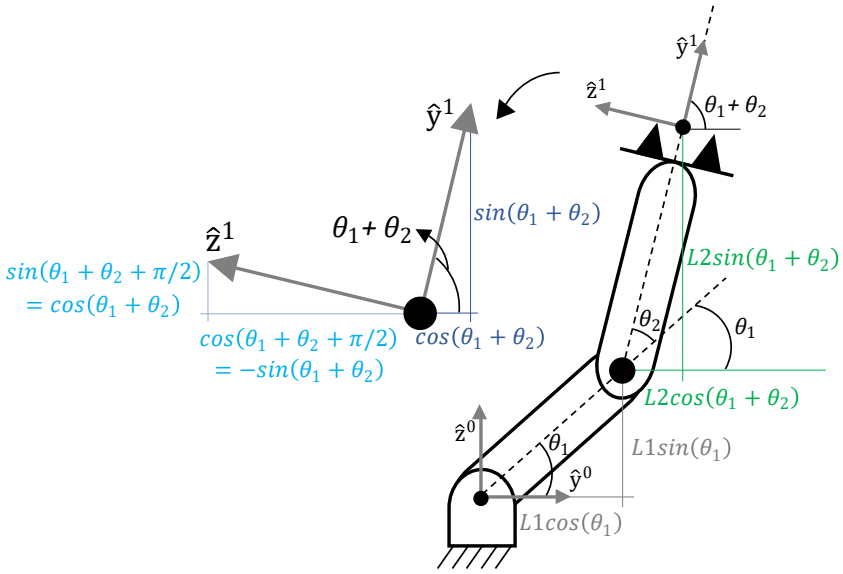


Figure 3.4 – Schematic representation of a 2-DoF manipulator with the elbow and tip projections in the base frame.

The previously presented topics and procedure are of great importance to understanding the fundamental aspects of the forward kinematics problem. Despite this, a systematic and general method called the Denavit-Hartenberg convention (DH) is typically followed[14]. In this convention, a coordinate frame is assigned to each link of the manipulator. It is considered that an arm has a total of $\mathrm{n}+1$ links, connected by n joints. The link 0 is conventionally fixed to the robot's base. As such, for a n DoF manipulator, there will be $\mathrm{n}+1$ coordinate frames, numbered from 0 to n. The assignment of the frames should be done following the steps presented next (particular considerations about the frames 0 and n will need to be made before):

1. Choose the z-axis of the i (goes from 0 to n) frame ($\hat{\mathrm{z}}_i$) along the axis of joint $i+1$. The direction of the z-axis is determined by the joint's positive direction of rotation.
2. Locate the origin of the frame i (O_i) at the intersection of the axis $\hat{\mathrm{z}}_i$ with the common normal to axes $\hat{\mathrm{z}}_i$ and $\hat{\mathrm{z}}_{i-1}$ (line containing the minimum distance segment between the two axes). If $\hat{\mathrm{z}}_i$ and $\hat{\mathrm{z}}_{i-1}$are parallel, then O_i should be located so that the distance between the current and

[14] There are different variations of the Denavit-Hartenberg convention (Waldron & Schmiedeler, 2016). In this thesis is used the one known as "classical" (particulary the steps and rules presented in Siciliano et al. (2009) and McKerrow (1991)) because it is the one that is used in the papers that constitute the foundations of this work.

the previous frame, measured in the z-axis of the latter, becomes 0. If the axes intercept, then O_i should be at the intersection of the axes axes $\hat{z}_i$ and $\hat{z}_{i-1}$.

3. Choose $\hat{x}_i$ axis along the common normal to axes $\hat{z}_i$ and $\hat{z}_{i-1}$ with direction from joint i to joint $i+1$. If $\hat{z}_i$ and $\hat{z}_{i-1}$are parallel, $\hat{x}_i$ is made coincident with the centerline of the link. If they intersect $\hat{x}_i$ can be made parallel, or anti-parallel, with the cross product of $\hat{z}_i$ and $\hat{z}_{i-1}$.
4. Choose $\hat{y}_i$ according to the right-hand rule.

These steps do not specify everything about frames 0 and n. As such, one should consider the following:

- The origin of frame 0 is located on the axis $\hat{z}_0$, and the axes $\hat{x}_0$ and $\hat{y}_0$ are chosen arbitrarily, respecting the right-hand rule. This frame can be made coincident with the robot's base frame if possible.
- The origin of frame n is attached to the robot's tip, and generally, it can be made equal to frame $n-1$.

Once the frames have been assigned, the position and orientation of the frame i concerning the frame $i-1$ can be described entirely using the following parameters:

- a_i–distance from frame $i-1$ to frame i measured along $\hat{x}_i$.
- d_i–distance from frame $i-1$ to frame i measured along $\hat{z}_{i-1}$.
- α_i–angle between $\hat{z}_{i-1}$ and $\hat{z}_i$ about $\hat{x}_i$ (positive rotation direction is counter-clockwise).
- θ_i–angle between $\hat{x}_{i-1}$ and $\hat{x}_i$ about $\hat{z}_{i-1}$ (positive rotation direction is counter-clockwise).

The parameters for all the links are generally summarized in a table as in Table 3.1.

Table 3.1 – DH parameters table example.

Link	a_i	d_i	α_i	θ_i
1	a_1	d_1	α_1	θ_1
		(...)		
n	a_n	d_n	α_n	θ_n

With each line of Table 3.1 is possible to obtain the homogeneous transformation matrix, which describes the position and orientation of the frame i relatively to frame $i-1$ (considering the notation simplification $cos = c$ and $sin = s$),

$$
{}^{i-1}T_i = Rot(\hat{z}_{i-1}, \theta_i)Trans(\hat{z}_{i-1}, d_i)Trans(\hat{x}_i, a_i)Rot(\hat{x}_i, \alpha_i) =
$$

$$
\begin{bmatrix} c(\theta_i) & -s(\theta_i) & 0 & 0 \\ s(\theta_i) & c(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c(\alpha_i) & -s(\alpha_i) & 0 \\ 0 & s(\alpha_i) & c(\alpha_i) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.16)
$$

$$
= \begin{bmatrix} c(\theta_i) & -s(\theta_i)c(\alpha_i) & s(\theta_i)s(\alpha_i) & a_i c(\theta_i) \\ s(\theta_i) & c(\theta_i)c(\alpha_i) & -c(\theta_i)s(\alpha_i) & a_i s(\theta_i) \\ 0 & s(\alpha_i) & c(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}.
$$

With the homogenous transformation matrices that relate all the frames that were assigned, two by two, it is possible, by multiplying all the matrices sequentially, to obtain the homogenous transformation matrix that relates the tip with the base, i.e., which gives the position and orientation of the robot's tip in relation to the base frame,

$$
{}^{0}T_n = {}^{0}T_1 {}^{1}T_2 \dots {}^{n-2}T_{n-1} {}^{n-1}T_n. \quad (3.17)
$$

3.1.2 Forward kinematics resolution

In Figure 3.5–a) is possible to observe the locations and positive directions of rotation for each joint of the manipulator. With this information, and following the Denavit–Hartenberg classical convention presented in the last section, it is possible to obtain the assignments of Figure 3.5–b) and the parameters of Table 3.2. It should be noted that, because it is possible (there are configurations in certain robots that cannot be described following the rules of DH, e.g., PUMA 560 with all links in a vertical direction) and convenient (allows one to deal with a single set of joints values), the kinematic zero-angle configuration of the arm is made coincident with the joint controller's zero-angle configuration.

Table 3.2 – Kuka LBR iiwa 14 R820 classic DH parameters table.

Link	a_i **(mm)**	d_i **(mm)**	α_i **(rad)**	θ_i**(rad)**
1	0	$d_{bs} = 360$	$-\pi/2$	θ_1
2	0	0	$\pi/2$	θ_2
3	0	$d_{se} = 420$	$\pi/2$	θ_3
4	0	0	$-\pi/2$	θ_4
5	0	$d_{ew} = 400$	$-\pi/2$	θ_5
6	0	0	$\pi/2$	θ_6
7	0	$d_{wt} = 350$	0	θ_7

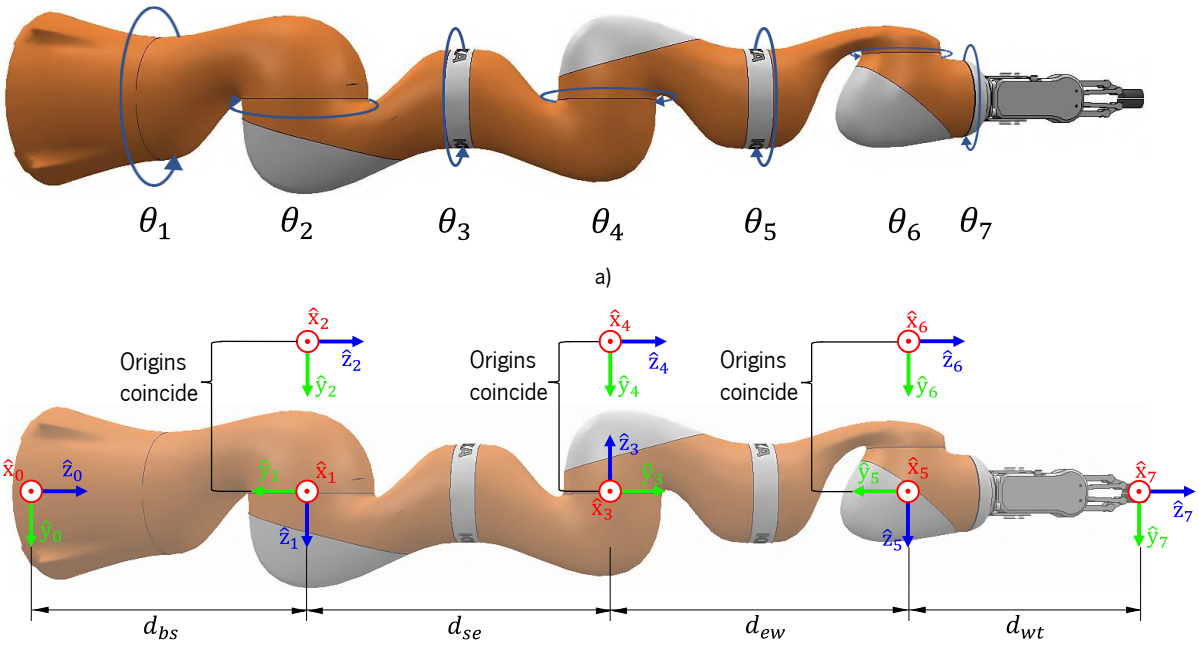


Figure 3.5 – Kuka LBR iiwa 14 R820 kinematic model: a) joints localizations and positive rotation directions; b) Coordinate frames assignment according to the Denavit-Hartenberg convention (d_{bs}: base–shoulder distance; d_{se}: shoulder–elbow distance; d_{ew}: elbow–wrist distance; d_{wt}: wrist–tip distance).

Using the matrix presented in 3.16 and the parameters of the Table 3.2, is possible to obtain the individual homogenous transformation matrices that relate all the frames that were assigned, two by two,

$$ {}^{0}T_{1} = \begin{bmatrix} c(\theta_1) & 0 & -s(\theta_1) & 0 \\ s(\theta_1) & 0 & c(\theta_1) & 0 \\ 0 & -1 & 0 & d_{bs} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.18} $$

$$ {}^{1}T_{2} = \begin{bmatrix} c(\theta_2) & 0 & s(\theta_2) & 0 \\ s(\theta_2) & 0 & -c(\theta_2) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.19} $$

$$ {}^{2}T_{3} = \begin{bmatrix} c(\theta_3) & 0 & s(\theta_3) & 0 \\ s(\theta_3) & 0 & -c(\theta_3) & 0 \\ 0 & 1 & 0 & d_{se} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.20} $$

$$ {}^{3}T_{4} = \begin{bmatrix} c(\theta_4) & 0 & -s(\theta_4) & 0 \\ s(\theta_4) & 0 & c(\theta_4) & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.21} $$

$$ {}^{4}T_{5} = \begin{bmatrix} c(\theta_5) & 0 & -s(\theta_5) & 0 \\ s(\theta_5) & 0 & c(\theta_5) & 0 \\ 0 & -1 & 0 & d_{ew} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.22} $$

$$ {}^{5}T_{6} = \begin{bmatrix} c(\theta_6) & 0 & s(\theta_6) & 0 \\ s(\theta_6) & 0 & -c(\theta_6) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.23} $$

$$ {}^{6}T_{7} = \begin{bmatrix} c(\theta_7) & -s(\theta_7) & 0 & 0 \\ s(\theta_7) & c(\theta_7) & 0 & 0 \\ 0 & 0 & 1 & d_{wt} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.24} $$

Using the equation 3.17 is possible to obtain the matrix ${}^{0}T_{7}$,

$$ {}^{0}T_{7} = {}^{0}T_{1}\,{}^{1}T_{2}\,{}^{2}T_{3}\,{}^{3}T_{4}\,{}^{4}T_{5}\,{}^{5}T_{6}\,{}^{6}T_{7}, \tag{3.25} $$

which gives the position (coordinates of the tip are in the first three rows of the last column) and orientation (the first three columns, in the first three rows, contain the coordinates of the unit vectors $\hat{x}_7$, $\hat{y}_7$, $\hat{z}_7$) of the robot's tip in relation to the base frame. With the same procedure, it is possible to obtain the position of other relevant points in the manipulator, such as the shoulder (using ${}^{0}T_{1}$ or ${}^{0}T_{2}$), elbow (using ${}^{0}T_{3}$ or ${}^{0}T_{4}$), and wrist (using ${}^{0}T_{5}$ or ${}^{0}T_{6}$).

3.1.3 Arm angle determination

As presented before, more than providing the tip's pose, it is also helpful that the FK gives the elbow position into the redundancy circle. In this dissertation, the parameter used to describe this position is the arm angle proposed by Shimizu et al. (2008). This parameter was also used by Faria et al. (2018), but some precautions were taken to consider the different configurations which the manipulator can assume to achieve a specific pose, as will be seen in this section. Recalling what was presented in the subchapter 2.2.2, the previously referred arm angle is defined by a reference and an arm plane (Figure 2.5). The latter is the SEW plane of the real manipulator (has the joint values that are being studied in the FK problem). The reference plane is the SEW plane of a virtual manipulator (SE_vW), which is a non-redundant image of the real arm (joint 3 is always in 0°).

To define the SEW and SE_vW planes, the shoulder, elbow, and wrist's coordinates of the virtual and real manipulator have to be determined. For the real manipulator, the strategy presented at the end of the last subchapter can be applied (using the information of the homogenous transformation matrices, which relate the frames assigned to the shoulder, elbow, and wrist with the base frame). As the shoulder and wrist's coordinates of the virtual manipulator are equal to those of the real arm (Figure 2.5), there is only a need to obtain the coordinates of the elbow for the virtual manipulator. For that, one has to follow two main steps: determine the angles of the joints 1, 2, 3, and 4, which allow for the virtual manipulator to have the same tip's pose as the real manipulator; use the calculated joint values to obtain the

homogeneous transformation matrix ${}^0T_3^v$ or ${}^0T_4^v$ and take from one of them the desired coordinates. In these two steps, two aspects should be taken into consideration: the joint 3 angular value is easily determined because for the virtual manipulator it is established that it is zero; the value of the joint 4 for the virtual manipulator is equal to the value of the real one because, as it will be seen, this joint will not change if the pose of the tip and the signal of the joint are maintained.

Considering the previously presented explanation, there is a need to obtain the values of the joints 1 and 2 for the virtual manipulator. To achieve this, the wrist's position relative to a frame placed on the shoulder is relevant (frame S in Figure 3.6, which results from a translation in the z-axis of the base frame, from a distance d_{bs}). As stated above, because the wrist's position of the real and virtual arm is the same, one could use an approach mainly based on FK to determine it. Despite this, a geometric approach is used because it will be helpful in the IK problem. This position (wrist relative to shoulder) can be obtained from the vector sum,

$$ {}^0p_{07} = {}^0p_{0S} + {}^0p_{S6} + {}^0p_{67}, \tag{3.26} $$

because the previous equation can be rewritten as,

$$ {}^0p_{S6} = {}^0p_{07} - {}^0p_{0S} - {}^0p_{67}, \tag{3.27} $$

and because the base and shoulder frames keep the same orientation in space (${}^0p_{S6} = {}^Sp_{S6}$),

$$ {}^Sp_{S6} = {}^0p_{07} - {}^0p_{0S} - {}^0p_{67}. \tag{3.28} $$

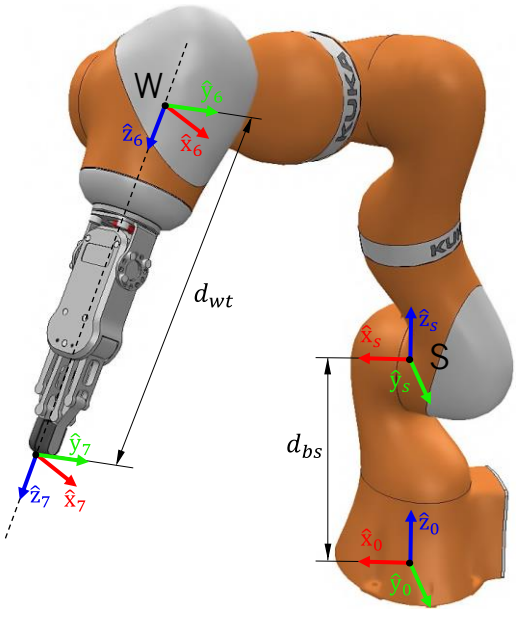


Figure 3.6 – Geometrical approach to determine the wrist's position.

It should be taken into account the following: ${}^0p_{07}$ is known because with FK one can determine ${}^0T_7$; ${}^0p_{0S}$ is given by the link distance d_{bs}: ${}^0p_{0S} = [0\ 0\ d_{bs}]$ (Figure 3.6); ${}^0p_{67}$ can be determined because ${}^7p_{67} = [0\ 0\ d_{wt}]$ (Figure 3.6), and as ${}^0R_7$ is known from ${}^0T_7$ one can write,

$$^{0}p_{67} = {}^{0}R_{7}\,{}^{7}p_{67}. \tag{3.29}$$

With the position of the wrist relative to the shoulder determined, the joint 1 value for the virtual manipulator can be obtained. Although, before, it is convenient to define de four-quadrant inverse tangent function (atan2) (The MathWorks, Inc., n.d.):

$$atan2(y,x) = \begin{cases} arctan\left(\frac{y}{x}\right) \;\; if\; x > 0, \\ arctan\left(\frac{y}{x}\right) + \pi \;\; if\; x < 0 \; and \; y \geq 0, \\ arctan\left(\frac{y}{x}\right) - \pi \;\; if\; x < 0 \; and \; y < 0, \\ \frac{\pi}{2} \; if\; x = 0 \; and \; y > 0, \\ -\frac{\pi}{2} \; if\; x = 0 \; and \; y < 0, \\ undefined \; if\; x = 0 \; and \; y = 0. \end{cases} \tag{3.30}$$

This function will be useful because it allows determining angles in the domain $[-\pi, \pi]$, i.e., with information about the quadrant where the angle is localized. Still, precautions must be taken when it is undefined. Thus, considering Figure 3.7, it can be written,

$$\theta_1^v = \begin{cases} atan2\left({}^{S}p_{S6,y}, {}^{S}p_{S6,x}\right), if\; \|{}^{S}p_{S6} \times [0\;0\;1]\| > 0 \\ 0, if\; \|{}^{S}p_{S6} \times [0\;0\;1]\| = 0 \end{cases}. \tag{3.31}$$

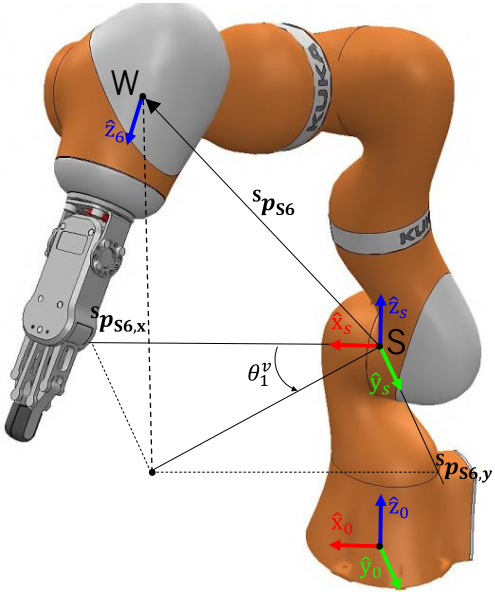


Figure 3.7 – Determination of the joint 1 value for the virtual manipulator.

Regarding the determination of θ_1^v four more aspects should be taken into consideration:

1. The condition $\|{}^{S}p_{S6} \times [0\;0\;1]\|$ is capable of describing the situation in which the atan2 is undefined because when ${}^{S}p_{S6,y}$ and ${}^{S}p_{S6,x}$ are zero, the vectors ${}^{S}p_{S6}$ and $[0\;0\;1]$ (relative to S frame) are collinear (Figure 3.8). As such, their cross product gives rise to a vector in which all the coordinates are zero.

2. In practice, $\|{}^{S}p_{S6} \times [0\ 0\ 1]\|$ will never be exactly 0. Thus, this condition will be compared with a small value instead (for example, 10^{-6}).
3. When $\|{}^{S}p_{S6} \times [0\ 0\ 1]\|$ is approximately 0, θ_1^v becomes 0, but other values could be conventioned.
4. Physically on the manipulator, the atan2 undefinition reflects the infinite possibilities that can be used in θ_1^v to put the wrist's root in the direction of the joint 1 axis (Figure 3.8). This situation is sometimes referred to as shoulder singularity[15] (Tian et al., 2021).

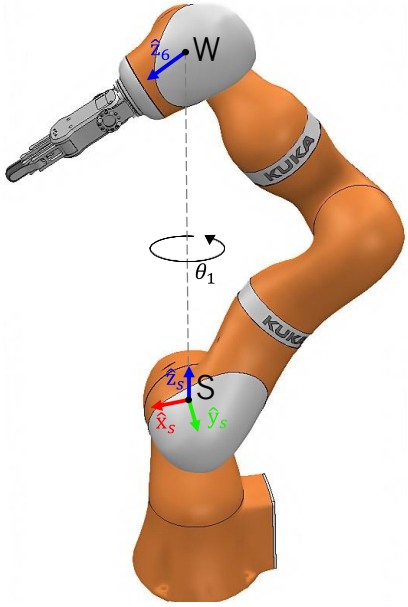


Figure 3.8 – Shoulder singularity: in this case the angular movement of joint 1 does not influence the wrist's position.

With the value of the joint 1 for the virtual manipulator determined, one needs to obtain the value of the joint 2. To achieve this, it is helpful to determine first the angle ϕ present in Figure 3.9. With the law of cosines, it can be written,

$$d_{ew} = \sqrt{{d_{se}}^2 + \|{}^{S}p_{S6}\|^2 - 2{d_{se}}^{S}p_{S6}\cos(\phi)}, \tag{3.32}$$

which can be rewritten in order to ϕ giving rise to the following expression[16],

$$\phi = arccos\left(\frac{{d_{se}}^2 + \|{}^{S}p_{S6}\|^2 - {d_{ew}}^2}{2{d_{se}}^{S}p_{S6}}\right). \tag{3.33}$$

The virtual manipulator used as a reference will not be the same for the configurations (of the real one) with the elbow joint (joint 4) positive and negative. This is useful because it will be in accordance with the

[15] This is an algorithmic singularity because it raises from the method used to calculate the joint values. These singularities are different from the kinematic ones, which are related to the robot's structure (Section 3.3 will present more details about this type).

[16] Being this a FK problem, it is sound to assume that a solution can be obtained because it is impossible to generate a combination of joint values that cause the argument of the arccos function to exceed its domain.

strategy used in the IK resolution (Section 3.2). Although the value of θ_2^v depends on the signal of the joint 4, the determination of ϕ for both cases is similar and uses the same equation.

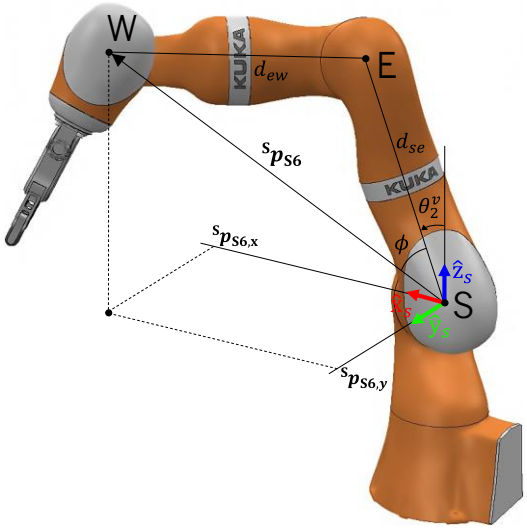


Figure 3.9 – Angle formed by the shoulder–wrist vector and the shoulder–elbow vector (ϕ), needed to obtain $\theta_2{}^v$.

With the value of ϕ and considering the representations presented in Figure 3.10, it is possible to obtain the value of the joint 2 for the virtual manipulator by making use of the following expression,

$$\theta_2^v = atan2\left(\sqrt{\left({}^{S}p_{S6,x}\right)^2 + \left({}^{S}p_{S6,y}\right)^2}, {}^{S}p_{S6,z}\right) + GC_4\phi. \tag{3.34}$$

The configuration flag GC_4 is -1 for the configurations that use the negative range of the joint 4 (Figure 3.10–a)), and 1 for the opposite case (Figure 3.10–b)).

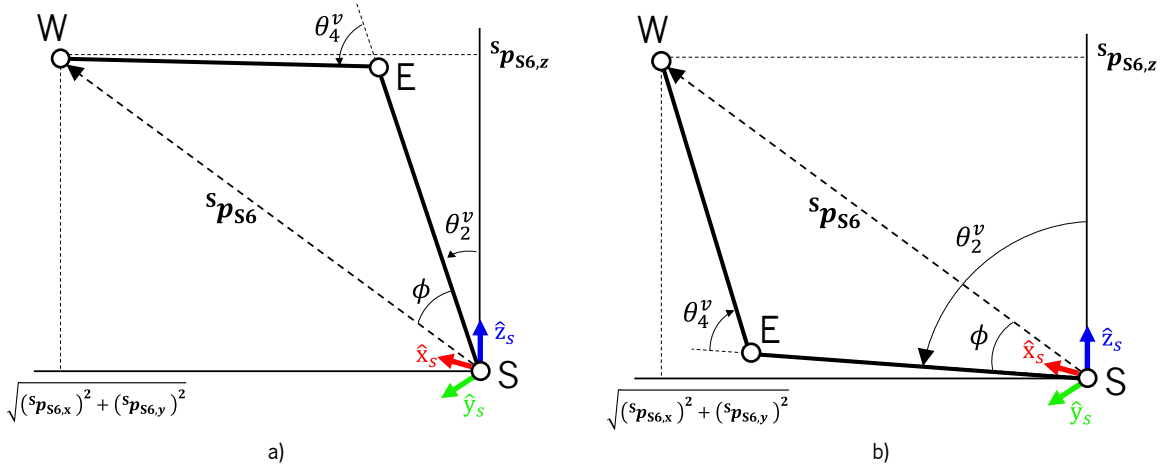


Figure 3.10 – Determination of the joint 2 value for the virtual manipulator: configuration with the joint 4 negative; configuration with the joint 4 positive.

With θ_1^v, θ_2^v, θ_3^v, and θ_4^v is possible to obtain ${}^{0}T_4^v$ (with FK), which gives the virtual elbow position relative to the base frame (${}^{0}p_{04}^v$). As such, all the information necessary to obtain the angle between the planes SEW and SE$_v$W (ψ) is known (the shoulder, elbow, and wrist's coordinates of the virtual and real manipulator). To recall, ${}^{0}p_{02} = {}^{0}p_{02}^v$, ${}^{0}p_{04}$, and ${}^{0}p_{06} = {}^{0}p_{06}^v$ can result directly from the FK matrices

${}^0T_1$ or ${}^0T_2$, ${}^0T_3$ or ${}^0T_4$, and ${}^0T_5$ or ${}^0T_6$, respectively[17]. The strategy used to determine ψ will be to define a normal vector to each plane and use the dot product to measure the angle between the vectors, which will be equal to the angle between SEW and SE$_V$W. In Figure 3.11, $\hat{v}_{SEW}^{v}$ is the unit vector normal to the reference plane SE$_V$W and $\hat{v}_{SEW}$ to the arm plane SEW.

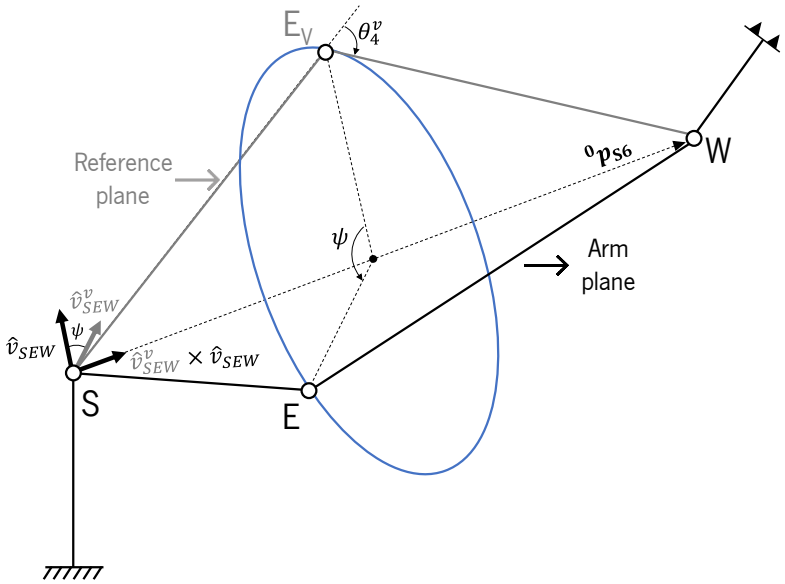


Figure 3.11 – Arm angle determination using the normal vectors to the planes SEW and SE$_V$W (for $\theta_4 < 0$ cases).

The vector v_{SEW}^{v} can be determined with the cross product of the vectors that connect the shoulder-elbow and shoulder-wrist,

$$v_{SEW}^{v} = ({}^0p_{04}^{v} - {}^0p_{02}) \times ({}^0p_{06} - {}^0p_{02}). \tag{3.35}$$

The normal vector to the plane SEW can be determined following the same approach but using the position of the real elbow instead,

$$v_{SEW} = ({}^0p_{04} - {}^0p_{02}) \times ({}^0p_{06} - {}^0p_{02}). \tag{3.36}$$

Making the cross product between these two vectors (v_{SEW}^{v} and v_{SEW}, $v_{SEW}^{v} \times v_{SEW}$) a vector collinear to ${}^0p_{S6}$ will be obtained (Figure 3.11). It is important to note that if the direction of $v_{SEW}^{v} \times v_{SEW}$ is the same of ${}^0p_{S6}$, the the arm angle ($\psi \in [-\pi, \pi]$) is positive, and in the opposite situation, the arm angle is negative[18]. Thus, the signal of ψ can be obtained by,

$$sg_{\psi} = \frac{(v_{SEW}^{v} \times v_{SEW}) \cdot {}^0p_{S6}}{\|v_{SEW}^{v} \times v_{SEW}\| \|{}^0p_{S6}\|}, \tag{3.37}$$

[17] For code implementation reasons obtaining ${}^0p_{06}$ with the vector subtraction ${}^0p_{07} - {}^0p_{67}$ can be useful. ${}^0p_{07}$ comes from ${}^0T_7$ which is known in a FK problem, and ${}^0p_{67}$ comes from equation 3.29.

[18] The rotation direction is given by the right hand rule with the vector that links the shoulder and the wrist, being in accordance with the strategy used in the IK resolution (Section 3.2).

because the dot product of collinear unit vectors is 1 when they have the same direction and -1 in the opposite case. With the signal of ψ determined, the value can be obtained with the dot product of $\hat{v}_{SEW}^{v}$ and $\hat{v}_{SEW}$,

$$\psi = sg_{\psi}\, arccos\left(\frac{v_{SEW}^{v} \cdot v_{SEW}}{\|v_{SEW}^{v}\|\|v_{SEW}\|}\right). \tag{3.38}$$

Using this approach, one must take into consideration that the vector resulting from $v_{SEW}^{v} \times v_{SEW}$ will be null when v_{SEW}^{v} and v_{SEW} are collinear (when ψ is 0 (v_{SEW}^{v} has the same direction of v_{SEW}) or π (v_{SEW}^{v} and v_{SEW} have opposite directions)), and, as such, the equation 3.37, and consequently the 3.38, become invalid. In this situation, the arm angle can be defined by,

$$\psi = \begin{cases} 0 \; if \; \|v_{SEW}^{v} - v_{SEW}\| < 10^{-7}, \\ \pi \; if \; \|v_{SEW}^{v} - v_{SEW}\| \geq 10^{-7}. \end{cases} \tag{3.39}$$

The calculations of the arm angle were presented for a case in which $\theta_4 < 0$, but they are also proper when $\theta_4 > 0$ if the respective virtual elbow position is used (Figure 3.12).

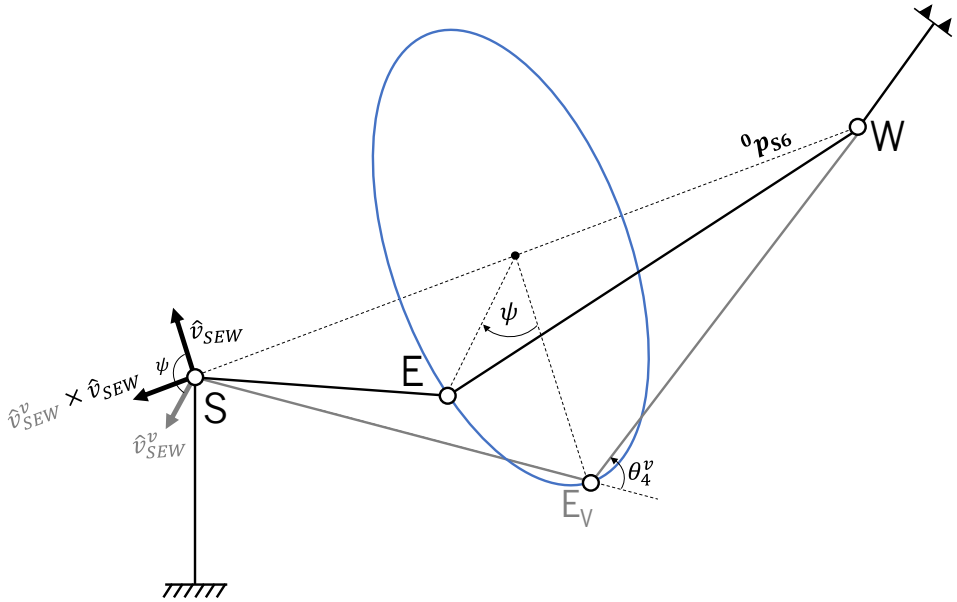


Figure 3.12 – Arm angle determination using the normal vectors to the planes SEW and SEᵥW (for the cases in which $\theta_4 > 0$).

3.2 Inverse kinematics

This subchapter presents a method to solve the IK problem of a 7-DoF serial manipulator. Recalling the definition presented in subchapter 2.2.1, the goal of IK is to determine the values of the joints that put the manipulator's tip in a specific position and orientation ($^0T_7$ is known). This problem may have multiple or no solutions. In the resolution of Faria et al. (2018), first, the values of the joint 4 and the vector $^0p_{S6}$ are determined. Then, the virtual elbow's position and orientation ($^0T_3^v$ or $^0T_4^v$) are obtained, as

presented in Section 3.1.3. Rotating the unit vectors associated to ${}^0T_3^v$ or ${}^0T_4^v$, by ψ around the shoulder-wrist vector (${}^0p_{S6}$), the ${}^0R_3$ or ${}^0R_4$ matrices can be found. With them and ${}^0R_7$ an algebraic approach can be used to determine the different possible values for each joint.

3.2.1 Shoulder-wrist vector and joint 4 determination

The determination of the shoulder-wrist vector (${}^0p_{S6}$) can be based on the approach presented in Section 3.1.3. Although, it should be noted that to calculate ${}^0p_{S6}$ one needs the shoulder's position and in the previously mentioned section the matrix ${}^0T_2$ was used obtain it. In an IK problem, not all the information in the matrix ${}^0T_2$ is known (because the joint 1 and 2 values are unknown), but the position data can still be readily determined. This is because it can be given by the first link length along the z-axis of the base coordinate frame (${}^0p_{02} = [0\ 0\ d_{bs}]$, Figure 3.6). The determined vector ${}^0p_{S6}$ allows to verify if the desired pose does not require a length in the links that go from the shoulder to the elbow and from the elbow to the wrist superior to the real one,

$$\|{}^0p_{S6}\| < d_{se} + d_{ew}. \tag{3.40}$$

The joint 4 value can be geometrically determined considering Figure 3.13 and following the presented equation,

$$\theta_4 = \pi - \gamma, \tag{3.41}$$

where γ can be given by the law of cosines,

$$\|{}^0p_{S6}\| = \sqrt{{d_{se}}^2 + {d_{ew}}^2 - 2d_{se}d_{ew}cos\,(\gamma)}, \tag{3.42}$$

because the expression presented above can be written as follows,

$$\gamma = arccos\left(\frac{{d_{se}}^2 + {d_{ew}}^2 - \|{}^0p_{S6}\|^2}{2d_{se}d_{ew}}\right). \tag{3.43}$$

As such, substituting the equation 3.43 in 3.41,

$$\theta_4 = \pi - arccos\left(\frac{{d_{ew}}^2 + {d_{se}}^2 - \|{}^0p_{S6}\|^2}{2d_{se}d_{ew}}\right), \tag{3.44}$$

which can be rewritten as,

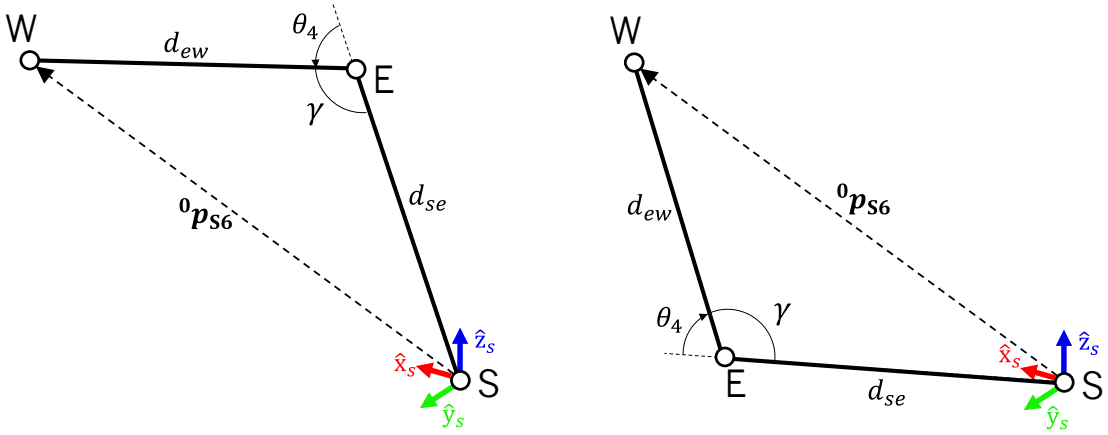
$$cos(\pi - \theta_4) = \frac{{d_{ew}}^2 + {d_{se}}^2 - \|{}^0p_{S6}\|^2}{2d_{se}d_{ew}}. \tag{3.45}$$

As $cos(\pi - \theta_4) = -cos(\theta_4)$ one can simplify the equation that gives θ_4 to,

$$\theta_4 = arccos\left(\frac{\|{}^0p_{S6}\|^2 - {d_{se}}^2 - {d_{ew}}^2}{2d_{se}d_{ew}}\right). \tag{3.46}$$

The obtained equation is suitable for the configurations in which the joint 4 is positive or negative (Figure 3.13), but its signal has to be defined by adopting the configuration flag GC_4 (that will be 1 when one wants to obtain a configuration with $\theta_4 > 0$ and -1 in the opposite situation),

$$\theta_4 = GC_4 arccos\left(\frac{\|{}^0p_{S6}\|^2 - {d_{se}}^2 - {d_{ew}}^2}{2d_{se}d_{ew}}\right). \tag{3.47}$$



a) b)

Figure 3.13 – Determination of the joint 4 value: configuration with the joint 4 negative; configuration with the joint 4 positive.

Before calculating the joint 4 value it can be a good practice to test if it will give rise to the "elbow singularity", which refers to the case in which joint 4 is 0 (the arm is stretched)(Tian et al., 2021). In this case, the input value of the arccos function used in equation 3.47 is approximately 1 (numerator and denominator are equal), and thus, it can be detected by the following condition,

$$\left|\|{}^0p_{S6}\|^2 - {d_{se}}^2 - {d_{ew}}^2 - 2d_{se}d_{ew}\right| < 10^{-6}. \tag{3.48}$$

3.2.2 Real elbow's rotation matrices determination

With the procedure presented in the Sections 3.1.3 and 3.2.1 it is possible to determine the values of joints 1, 2, 3, and 4 that allow a non-redundant image of the manipulator to reach a specific pose with its tip. Substituting these values in the DH matrices presented in the Section 3.1.2, one can obtain ${}^0R_3^v$ and ${}^0R_4^v$, which give the orientation of the DH coordinate frames placed on the elbow of the virtual manipulator (each column depicts a unit vector, and each of these, in turn, represents an axis of the coordinate system). An example of a coordinate frame respecting a ${}^0R_4^v$ matrix can be observed in Figure 3.14 ($\hat{x}_4^v$, $\hat{y}_4^v$, and $\hat{z}_4^v$). This figure also helps understand that if a coordinate frame placed on the elbow of the virtual manipulator is rotated around the shoulder-wrist vector (${}^0p_{S6}$), by an angle ψ, the corresponding coordinate frame placed on the elbow of the real manipulator can be obtained ($\hat{x}_4$, $\hat{y}_4$, and $\hat{z}_4$). Mathematically, this can be achieved with the Rodrigues' rotation formula because it allows

rotating the three unit vectors of a rotation matrix, given an angle of rotation and an axis. In its matrix notation, the Rodrigues' rotation formula gives rise to the following matrix,

$$ {}^{0}R_{\psi} = I_3 + sin(\psi)\left[\widehat{{}^{0}p_{S6}}\times\right] + (1 - cos(\psi))\left[\widehat{{}^{0}p_{S6}}\times\right]^2, \tag{3.49} $$

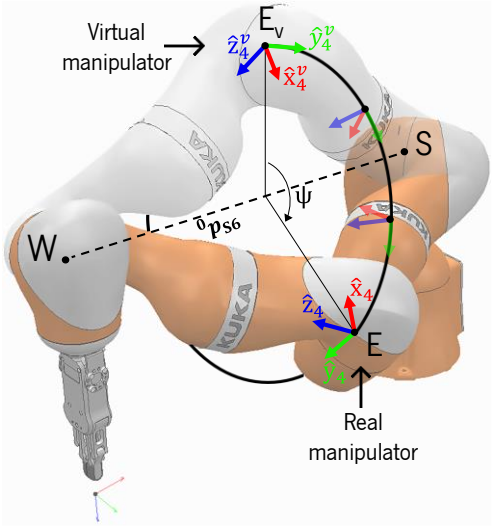


Figure 3.14 – Real elbow's rotation matrices determination, using information about the virtual manipulator's elbow.

where I_3 is a 3×3 identity matrix and $\widehat{{}^{0}p_{S6}}\times$ is the cross-product matrix for the unit vector $\widehat{{}^{0}p_{S6}}$,

$$ \widehat{{}^{0}p_{S6}}\times = \begin{bmatrix} 0 & -\widehat{{}^{0}p_{S6,z}} & \widehat{{}^{0}p_{S6,y}} \\ \widehat{{}^{0}p_{S6,z}} & 0 & -\widehat{{}^{0}p_{S6,x}} \\ -\widehat{{}^{0}p_{S6,y}} & \widehat{{}^{0}p_{S6,x}} & 0 \end{bmatrix}. \tag{3.50} $$

With ${}^{0}R_{\psi}$, ${}^{0}R_3^v$, and ${}^{0}R_4^v$ is possible to obtain ${}^{0}R_4$ and ${}^{0}R_3$ with the presented equations,

$$ {}^{0}R_4 = {}^{0}R_{\psi}\,{}^{0}R_4^v, \tag{3.51} $$

$$ {}^{0}R_3 = {}^{0}R_{\psi}\,{}^{0}R_3^v. \tag{3.52} $$

Substituting the equation 3.49 in 3.52 and assuming the notation of Faria et al. (2018) and Shimizu et al. (2008), ${}^{0}R_3$ can be written using three auxiliary matrices (A_s, B_s, and C_s),

$$ {}^{0}R_3 = A_s\, sin(\psi) + B_s\, cos(\psi) + C_s, \tag{3.53} $$

where,

$$ A_s = \left[\widehat{{}^{0}p_{S6}}\times\right]{}^{0}R_3^v, \tag{3.54} $$

$$ B_s = -\left[\widehat{{}^{0}p_{S6}}\times\right]^2{}^{0}R_3^v, \tag{3.55} $$

$$ C_s = \left[\widehat{{}^{0}p_{S6}}\;\widehat{{}^{0}p_{S6}}^{T}\right]{}^{0}R_3^v. \tag{3.56} $$

3.2.3 Joint 1, 2, and 3 values determination

The matrix ${}^0R_3$ gives the numerical values that describe the orientation of a frame placed on the manipulator's elbow (in accordance with the rules of the Denavit-Hartenberg convention[19]). An equivalent matrix, but constituted by algebraic expressions with the variables θ_1, θ_2, and θ_3, can be determined with the multiplication of the matrices ${}^0T_1$, ${}^1T_2$, and ${}^2T_3$, presented in Section 3.1.2, giving rise to,

$$ {}^0R_3(\theta_{1,2,3}) = $$

$$\begin{bmatrix} c(\theta_1)c(\theta_2)c(\theta_3) - s(\theta_1)s(\theta_3) & c(\theta_1)s(\theta_2) & c(\theta_3)s(\theta_1) + c(\theta_1)c(\theta_2)s(\theta_3) \\ c(\theta_1)s(\theta_3) + c(\theta_2)c(\theta_3)s(\theta_1) & s(\theta_1)s(\theta_2) & c(\theta_2)s(\theta_1)s(\theta_3) - c(\theta_1)c(\theta_3) \\ -c(\theta_3)s(\theta_2) & c(\theta_2) & -s(\theta_2)s(\theta_3) \end{bmatrix}. \quad (3.57)$$

The algebraic expressions present in the matrix exposed above allow one to determine equations that make use of the numerical values resulting from the matrix obtained with equation 3.53 to define the θ_1, θ_2, and θ_3 values, that put the manipulator with the desired elbow pose. For example, from matrix 3.57 is possible to conclude that if the value of the third row and second column of the numeric matrix resulting from equation 3.53 is used as an input value for the arccos function, the joint 2 value can be determined. As such, it can be written[20],

$$\theta_2 = arccos(a_{s32}\, sin(\psi) + b_{s32}\, cos(\psi) + c_{s32}). \quad (3.58)$$

Although, as $cos(\theta_2) = cos\ (-\theta_2)$, a second (and negative) possibility for θ_2 exists, which means that it is possible to obtain a solution with $\theta_2 > 0$ and with $\theta_2 < 0$. To choose between the two, just as for joint four, this dissertation will use a configuration flag, GC_2, that will be 1 for the solution that has $\theta_2 > 0$ and -1 in the opposite case,

$$\theta_2 = GC_2\, arccos(a_{s32}\, sin(\psi) + b_{s32}\, cos(\psi) + c_{s32}). \quad (3.59)$$

The same procedure can be used to determine the value of the joint 1 and joint 3, resulting in,

$$\theta_1 = atan2(GC_2[a_{s22}\, sin(\psi) + b_{s22}\, cos(\psi) + c_{s22}],\ GC_2[a_{s12}\, sin(\psi) + b_{s12}\, cos(\psi) + c_{s12}])' \quad (3.60)$$

$$\theta_3 = atan2(GC_2[-a_{s33}\, sin(\psi) - b_{s33}\, cos(\psi) - c_{s33}],\ GC_2[-a_{s31}\, sin(\psi) - b_{s31}\, cos(\psi) - c_{s31}])' \quad (3.61)$$

Regarding the latter two equations, there are some aspects that should be emphasized:

1. To determine the values of θ_1 and θ_3, elements of the matrix ${}^0R_3$ that also depend on θ_2 are used. For instance, the equation related to θ_1 (3.60) uses elements that correspond to the

[19] Figure 3.14 helps to understand that a Denavit-Hartenberg convention's frame positioned on the elbow, when rotated around the ${}^0p_{s6}$ vector, keeps respecting the DH rules. In Figure 3.14 the represented elbow's frame is the fourth and one knows that following the defined DH parameters (Table 3.2) this frame must always have its y and z-axis in the direction of the joint 4 and 5 axis, respectively, and this conditions keep present when the frame is rotated.

[20] A simpler notation, as $\theta_2 = arccos\ ({}^0R_3[3,2])$, could be used, but the presented one will be more convinent in the following section.

algebraic expressions $s(\theta_1)s(\theta_2)$ (${}^0R_3(\theta_{1,2,3})[2,2]$) and $c(\theta_1)s(\theta_2)$ (${}^0R_3(\theta_{1,2,3})[1,2]$), however the presence of an $s(\theta_2)$ does not invalidate the determination of θ_1. To better understand why, as the atan2 can result in different cases (function 3.30), it is convenient to analyze the three that can generally happen. When the second input of the atan2 function is different from zero, both the inputs are divided. Then they serve as an input for the atan function. Being both elements affected by $s(\theta_2)$, the division will cancel its effect and the determination of θ_1 remains possible. If only the second input is zero, that means that $c(\theta_1)$ is zero, which leads to the conclusion that θ_1 is equal to ±90°. Also in this case θ_1 can be determined. These two cases were presented for the θ_1 case, but for θ_3 the explanation is equivalent. When both the inputs of the atan2 function are zero, which implies that θ_2 is 0°, because $s(\theta_2)$ is zero, the determination of θ_1 and θ_3 remains possible but it will be related. In these cases ${}^0R_3(\theta_{1,2,3})$ becomes,

$$ {}^0R_3(\theta_{1,2=0,3}) = \begin{bmatrix} c(\theta_1)c(\theta_3) - s(\theta_1)s(\theta_3) & 0 & c(\theta_3)s(\theta_1) + c(\theta_1)s(\theta_3) \\ c(\theta_1)s(\theta_3) + c(\theta_3)s(\theta_1) & 0 & s(\theta_1)s(\theta_3) - c(\theta_1)c(\theta_3) \\ 0 & 1 & 0 \end{bmatrix}. \quad (3.62) $$

In this case, the mathematical combinations between the values for θ_1 and θ_3 that give rise to a numerical matrix equal to the one that results from equation 3.53, are infinite. As such, a way to solve this indeterminacy can be based on the selection of a value to θ_1 or θ_3, and on the determination of the joint value that was not specified accordingly to the one that was. For example, one can impose that the value of θ_3 is zero, and in that way the ${}^0R_3(\theta_{1,2=0,3})$ gets to be,

$$ {}^0R_3(\theta_{1,2=0,3=0}) = \begin{bmatrix} c(\theta_1) & 0 & s(\theta_1) \\ s(\theta_1) & 0 & -c(\theta_1) \\ 0 & 1 & 0 \end{bmatrix}. \quad (3.63) $$

As such, θ_1 can be given by,

$$ \theta_1 = atan2([a_{s21}\, sin(\psi) + b_{s21}\, cos(\psi) + c_{s21}], [a_{s11}\, sin(\psi) + b_{s11}\, cos(\psi) + c_{s11}]). \quad (3.64) $$

2. Another important aspect related to the equations 3.60 and 3.61 is that it is essential to take into consideration the signal of θ_2, using the configuration flag GC_2, and the signals of the matrix 3.57. Otherwise, the absolute values of the cosine and sine functions, obtained with the values of θ_1 and θ_3, may be correct but the signals not, and therefore, the numerical matrix obtained with the equation 3.53 is not be respected.

3.2.4 Joint 5, 6, and 7 values determination

At this stage, the numeric matrices ${}^0R_3$, ${}^3R_4$, and ${}^0R_7$ are known. To recall: ${}^0R_3$ was obtained with equation 3.53; ${}^3R_4$ can be obtained with the joint 4 value resulting from the Section 3.2.1, and with the ${}^3T_4$ algebraic matrix presented in Section 3.1.2; ${}^0R_7$ is known in the IK problem because the desired pose is given. Thus the numeric matrix ${}^4R_7$ can be obtained (${}^4R_7 = ({}^3R_4)^T({}^0R_3)^T{}^0R_7$). Keeping the notation of Faria et al. (2018) and Shimizu et al. (2008), ${}^4R_7$ can be written using six auxiliary matrices (A_s, B_s, C_s, A_w, B_w, and C_w),

$$ {}^4R_7 = A_w\, sin(\psi) + B_w\, cos(\psi) + C_w, \tag{3.65} $$

where,

$$ A_w = {}^3R_4^T\, A_s^T\, {}^0R_7, \tag{3.66} $$

$$ B_w = {}^3R_4^T\, B_s^T\, {}^0R_7, \tag{3.67} $$

$$ C_w = {}^3R_4^T\, C_s^T\, {}^0R_7. \tag{3.68} $$

Following the same approach of the previous section, a matrix constituted by algebraic expressions equivalent to the numeric matrix ${}^4R_7$ (obtained with equation 3.65[21]) can be obtained. This matrix has the variables θ_5, θ_6, and θ_7 and can be determined with the multiplication of the matrices ${}^4T_5$, ${}^5T_6$, and ${}^6T_7$, presented in Section 3.1.2. This multiplication gives rise to,

$$ {}^4R_7(\theta_{5,6,7}) = $$

$$ \begin{bmatrix} c(\theta_5)c(\theta_6)c(\theta_7) - s(\theta_5)s(\theta_7) & -c(\theta_7)s(\theta_5) - c(\theta_5)c(\theta_6)s(\theta_7) & c(\theta_5)s(\theta_6) \\ c(\theta_5)s(\theta_7) + c(\theta_6)c(\theta_7)s(\theta_5) & c(\theta_5)c(\theta_7) - c(\theta_6)s(\theta_5)s(\theta_7) & s(\theta_5)s(\theta_6) \\ -c(\theta_7)s(\theta_6) & s(\theta_7)s(\theta_6) & c(\theta_6) \end{bmatrix}. \tag{3.69} $$

With the matrix ${}^4R_7$ both in the numerical and algebraic form obtained, it is possible to verify that a similar procedure to the one used to determine the joints 1, 2, and 3 values can be applied in the present section. This conclusion is sound because both groups of joints constitute a spherical joint. With this in mind, similarly to θ_2,

$$ \theta_6 = GC_6 arccos(a_{w33}\, sin(\psi) + b_{w33}\, cos(\psi) + c_{w33}), \tag{3.70} $$

and similarly to θ_1 and θ_3,

$$ \theta_5 = atan2(GC_6[a_{w23}\, sin(\psi) + b_{w23}\, cos(\psi) + c_{w23}], \; GC_6[a_{w13}\, sin(\psi) + b_{w13}\, cos(\psi) + c_{w13}])' \tag{3.71} $$

$$ \theta_7 = atan2(GC_6[a_{w32}\, sin(\psi) + b_{w32}\, cos(\psi) + c_{w32}], \; GC_6[-a_{w31}\, sin(\psi) - b_{w31}\, cos(\psi) - c_{w31}])'. \tag{3.72} $$

[21] The numeric matrix ${}^4R_7$, obtained with equation 3.65, follows the Denavit-Hartenberg convention because it is derived from matrices that are in accordance with this convention.

In case $\theta_6 = 0°$ is not possible to use equations 3.71 and 3.72 to determine θ_5 and θ_7, respectively, because ${}^4R_7(\theta_{5,6,7})$ becomes,

$$ {}^4R_7(\theta_{5,6=0,7}) = \begin{bmatrix} c(\theta_5)c(\theta_7) - s(\theta_5)s(\theta_7) & -c(\theta_7)s(\theta_5) - c(\theta_5)s(\theta_7) & 0 \\ c(\theta_5)s(\theta_7) + c(\theta_7)s(\theta_5) & c(\theta_5)c(\theta_7) - s(\theta_5)s(\theta_7) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (3.73) $$

Similar to what happened when θ_2 was zero, when θ_6 is zero, there are infinity possibilities for θ_5 and θ_7. Thus, one of the values can be conventioned and the other determined accordingly. Making θ_5 zero, ${}^4R_7(\theta_{5,6=0,7})$ gets to be,

$$ {}^4R_7(\theta_{5=0,6=0,7}) = \begin{bmatrix} c(\theta_7) & -s(\theta_7) & 0 \\ s(\theta_7) & c(\theta_7) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (3.74) $$

making θ_7 be,

$$ \theta_7 = atan2([a_{s21}\, sin(\psi) + b_{s21}\, cos(\psi) + c_{s21}], [a_{s11}\, sin(\psi) + b_{s11}\, cos(\psi) + c_{s11}]). \quad (3.75) $$

3.2.5 Final considerations

This subchapter presented a method to solve the IK, which considers the arm's redundancy and the different solutions that may exist for each arm angle. These solutions are given by the combination of the signals in the joints 2, 4, and 6 (GC_2, GC_4, GC_6) and are enumerated from 0 to 7 in Table 3.3.

Table 3.3 – Numeration used in this dissertation for the IK solutions.

IK solution	GC_2	GC_4	GC_6
0	1	1	1
1	-1	1	1
2	1	-1	1
3	-1	-1	1
4	1	1	-1
5	-1	1	-1
6	1	-1	-1
7	-1	-1	-1

Furthermore, the presented method, as will be seen in the following section, facilitates the mapping of joint limits and singularities into the redundancy circle (which is highly important to this dissertation because it will allow placing the arm angle farther from the mapped limits, which will tend to increase manipulability).

Other conventions could be used to solve the indeterminacies that can arise when θ_2 and θ_6 become zero. The ones proposed in the above sections are just examples. Although these procedures are relevant to understand the IK, they will not be crucial to this dissertation because the strategy used will be to avoid the situations in which the θ_2 and θ_6 need to be zero. The previous sections studied these cases mainly from a mathematical standpoint, but the following subchapter will give the physical meaning and safety risks associated with them ($\theta_2 = 0°$ or $\theta_6 = 0°$ will specify kinematic singularities).

3.3 Joint limits and singularities mapping into the redundancy circle

This subchapter first discusses the kinematic singularities of a 7-DoF serial manipulator. It then summarizes the method presented by Shimizu et al. (2008) and Faria et al. (2018), and extended by Wang et al. (2021), to map into the RC these singularities and also joint limits. This mapping is crucial for the trajectory tracking method proposed in this dissertation (Section 4).

3.3.1 7-DoF serial manipulator kinematic singularities

Craig (2014) presents an excellent example of kinematic singularities in mechanisms and of the problems that can arise when one is crossed. This example is related to the rear–gunner's mechanism of a World War I plane–Figure 3.15 (2 DoF: elevation and azimuth). To understand it, one should imagine an enemy plane moving in a way that makes the gunner increase the elevation continuously, and eventually passing nearly overhead (near the azimuth rotation axis–Figure 3.15). In the latter situation, an abrupt change in the value of the azimuth would be required for the gunner to be able to keep shooting at the enemy.

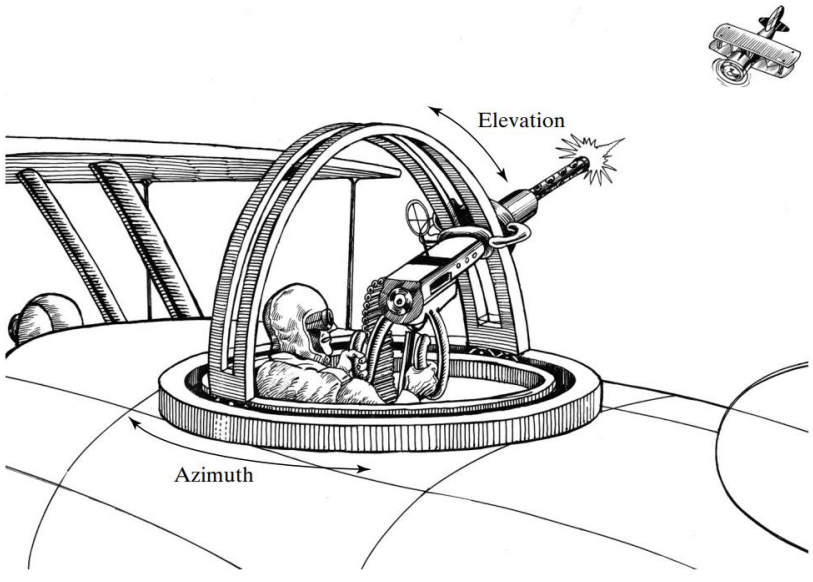


Figure 3.15 – Rear-gunner's mechanism of a World War I biplane (reproduced from Craig (2014)).

The described situation is a singularity, and all mechanisms are prone to these complications, including robots. The presented example allows to understand in a simple way fundamental aspects and problems of singularities:

- The rear–gunner could shoot static objects placed anywhere in the upper hemisphere without having problems with the presented singularity. Although, when shooting a moving airplane, problems can arise. Analogously, singularities do not prevent manipulators from achieving poses within their workspace, but they can cause problems in movements that cross them.
- If one tried to solve the IK problem for the present mechanism when the target is in the direction of the azimuth axis, an infinite number of possibilities could be chosen for the latter DoF, i.e., one would have to deal with a mathematical indeterminacy.
- When the rear–gunner orientates the stream of bullets in the direction of the azimuth's axis, the changes in the latter DoF will not affect their path. It can be said that a joint has momentarily lost its usefulness and the mechanism behaves as if it had only one degree of freedom, which will restrict the movements in the Cartesian space (i.e., the manipulability).
- Even if the enemy airplane crossed the rear–gunner's overhead at a low velocity, the value of the azimuth would have to be changed at a very high rate, i.e., near singularities, small velocities/displacements in the Cartesian space may cause major velocities/displacements in the joint space.

Singularities can be classified into:

- **Boundary singularities:** this kind of singularity happens when the arm is fully stretched or retracted, i.e., when the manipulator is driven to the boundaries of its workspace. They are detected by analyzing the joint 4 values because when the arm is stretched, this joint will tend to be zero (condition given by equation 3.48 and called "arm singularity" in Section 3.2.1), when the arm is folded, the joint 4 (angular values given by equation 3.47) will tend to exceed the limit ($\pm 120^{\circ}$).
- **Internal singularities:** are generally caused by the alignment of two or more joint axes and are not related to the workspace boundaries. In this manipulator, there are two: $\theta_2 = 0^{\circ}$ (leads to the alignment of the joints 1 and 3 axes: Figure 3.16–a)) and $\theta_6 = 0^{\circ}$ (leads to the alignment of the joints 5 and 7 axes: Figure 3.16–b)). These singularities can not be identified in such a straightforward way as the previously mentioned ones because the values θ_2 and θ_6 depend on the arm angle. Thus, these singularities will be mapped into the RC in the next subchapter. Figure

3.16 helps to understand why in the Sections 3.2.3 (IK–Determination of the joint 1, 2, and 3 values) and 3.2.4 (IK–Determination of the joint 5, 6, and 7 values), one concluded that when the joint 2 or 6 was 0, there would be a mathematical indetermination. When these joints are in the 0 position, they align the previous and next joint axes, making them have the same practical effect. For example, when the joint 6 is 0, having the joint 5 and 7 in the positions 15° and 30°, respectively, is the same that having the joint 5 and 7 in the positions 0° and 45°, respectively.

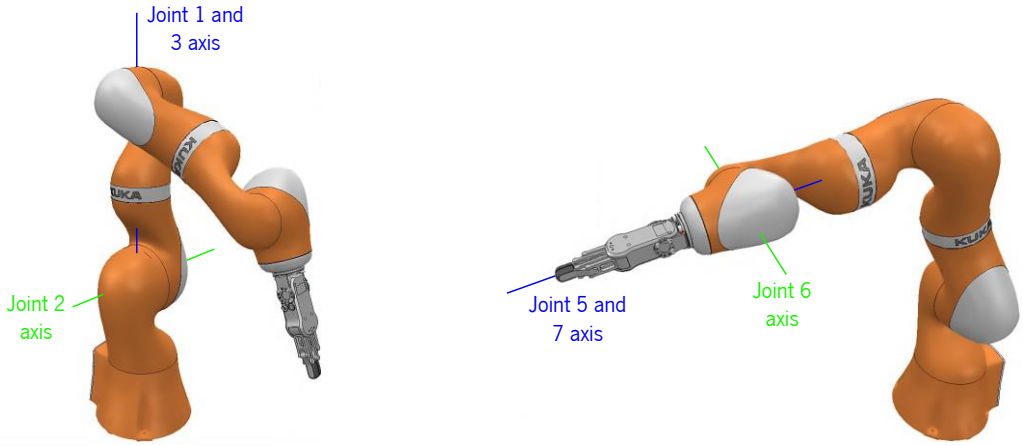


Figure 3.16 – Internal singularities: a) $\theta_2 = 0°$; b) $\theta_6 = 0°$.

With the gunner's example, it was possible to introduce four aspects of singularities. Still, the fourth can be hard to imagine in a complex 7-DoF serial manipulator (small displacements in Cartesian space giving rise to major joint movements). As such, an example is presented in Figure 3.17: with the joint 6 in 0°, a displacement of only 50 mm in this joint axis direction originates joint displacements of roughly 90°.

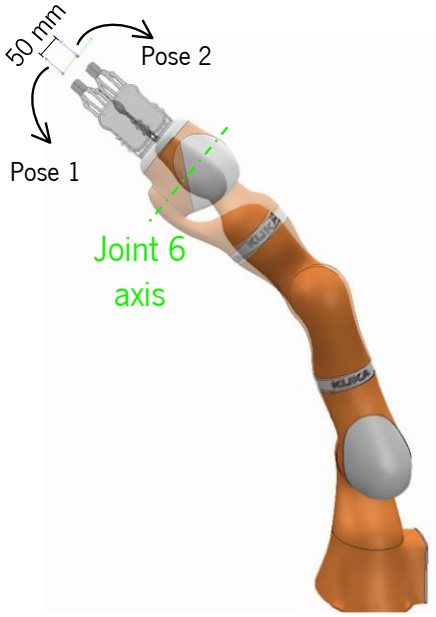


Figure 3.17 – Small displacement in the Cartesian space causing a major one in the joint space, due to the joint 6 singularity.

3.3.2 Arm angle feasible intervals

To map the joint limits and singularities into the redundancy circle, one has to study the behavior of the joints with the variation of the arm angle from $-\pi$ to π. This behavior is given by the equations 3.59–3.61 and 3.70–3.72. The latter equations can be written in two generic ways. For the joints 1, 3, 5, and 7,

$$\theta_i(\psi) = atan2(GC_k[a_n\, sin(\psi) + b_n\, cos(\psi) + \ c_n]\,, \\ GC_k[a_d\, sin(\psi) + b_d\, cos(\psi) + \ c_d])' \tag{3.76}$$

and for the joints 2 and 6,

$$\theta_i(\psi) = GC_k arccos(a\, sin(\psi) + b\, cos(\psi) + \ c). \tag{3.77}$$

The differentiation of these functions can give the necessary information for one to understand their overall behavior. With that information, it becomes easier to outline a strategy for mapping joint limits and singularities into the redundancy circle. Starting with the generic equation 3.76, considering the following notation simplification:

$$\theta_i(\psi) = atan2(u, v), \tag{3.78}$$

where,

$$u = GC_k[a_n\, sin(\psi) + b_n\, cos(\psi) + \ c_n], \\ v = GC_k[a_d\, sin(\psi) + b_d\, cos(\psi) + \ c_d], \tag{3.79}$$

and knowing that the derivative of $atan2(u, v)$ is given by,

$$-\frac{uv'}{v^2 + u^2} + \frac{vu'}{v^2 + u^2}, \tag{3.80}$$

one can obtain,

$$\frac{d\theta_i}{d\psi} = \frac{a_t\, sin(\psi) + b_t\, cos(\psi) + c_t}{v^2 + u^2}, \tag{3.81}$$

where,

$$a_t = GC_k(c_n b_d - b_n c_d), \\ b_t = GC_k(a_n c_d - c_n a_d), \\ c_t = GC_k(a_n b_d - b_n a_d). \tag{3.82}$$

The stationary points of the function 3.76 (ψ_0) can be obtained making $\frac{d\theta_i}{d\psi} = 0$ and using a change of variable known as Weierstrass-substitution method,

$$\psi_0 = 2\, arctan\left(\frac{a_t \pm \sqrt{a_t^2 + b_t^2 - c_t^2}}{b_t - c_t}\right). \tag{3.83}$$

According to the value of $a_t^2 + b_t^2 - c_t^2$, the function 3.76 can assume three distinct behaviors:

1. When $a_t^2 + b_t^2 - c_t^2 > 0$ there are two stationary points. The two generic profiles that can result in this case can be seen in Figure 3.18–a) and b). The difference between the two cases is that in the second situation $\theta(\psi)$ crosses the boundaries of the atan2 function domain ($[-\pi, \pi]$), which leads to discontinuities.

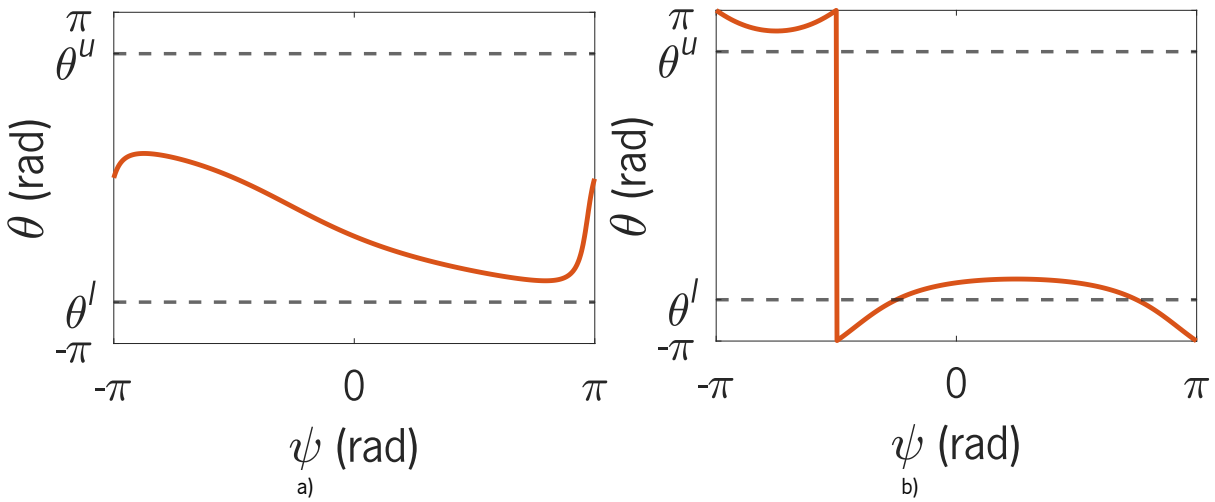


Figure 3.18 – $\theta(\psi)$ generic profile for joints 1, 3, 5, and 7 when stationary points exist: a) $\theta(\psi)$ does not cross the atan2 domain boundaries; b) $\theta(\psi)$ crosses the atan2 domain boundaries.

2. If $a_t^2 + b_t^2 - c_t^2 < 0$ no stationary points exist, and the profile of the function 3.76 is monotonic. The generic case can be observed in Figure 3.19 (where the apparently abrupt variation is related to the atan2 domain).

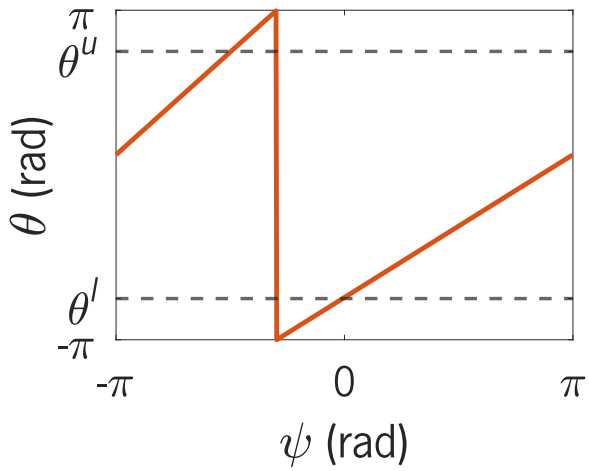


Figure 3.19 – $\theta(\psi)$ generic profile for joints 1, 3, 5, and 7 when no stationary points exist.

3. If $a_t^2 + b_t^2 - c_t^2 = 0$ a singularity exists. $\theta(\psi)$ becomes indeterminate because both u and v of equation 3.78 turn out to be 0 (Sections 3.2.3 and 3.2.4). In this case, the generic profiles of the function 3.76 have a discontinuity that should not be mistaken with the one caused by the boundaries of the atan2 function domain (Figure 3.20–a) to b)).

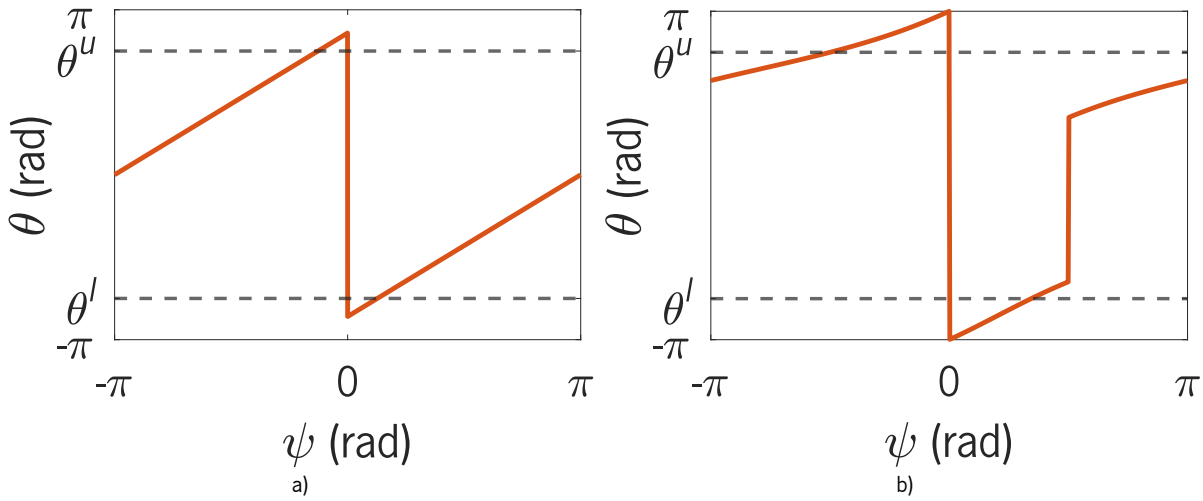


Figure 3.20 – $\theta(\psi)$ generic profile for joints 1, 3, 5, and 7 when a singularity exists: a) $\theta(\psi)$ does not cross the atan2 domain boundaries; b) $\theta(\psi)$ crosses the atan2 domain boundaries.

Knowing the generic profiles of the function 3.76, it is now helpful to determine the values of the arm angle where the joints cross the limits (if they exist), i.e., the values of ψ where the generic profiles $\theta(\psi)$ cross the limits θ^l or θ^u. For that, one can use the equation that results from applying the Weierstrass-substitution method to the equation 3.76,

$$\psi = 2arctan\left(\frac{-b_p \pm \sqrt{b_p^2 - 4a_p c_p}}{2a_p}\right), \tag{3.84}$$

where,

$$\begin{aligned} a_p &= GC_k[(c_d - b_d)\,tan(\theta) + (b_n - c_n)], \\ b_p &= 2GC_k[a_d\,tan(\theta) - a_n], \\ c_p &= GC_k[(b_d + c_d)\,tan(\theta) - (b_n + c_n)]. \end{aligned} \tag{3.85}$$

Finally, the following procedure can be followed to obtain the joints 1, 3, 5, and 7 feasible intervals:

1. If $\theta(\psi)$ crosses its limits (θ^l and θ^u of the above figures), the corresponding ψ values are usually given by equation 3.84. The generic profiles previously studied allow to understand that two or four ψ values are obtained[22], which should be arranged in ascending order (like $[\psi_1\ \psi_2\ \psi_3\ \psi_4]$). Subsequently, the signal of $\theta'(\psi_1)$ is calculated (using equation 3.81), and compared with the one of $\theta(\psi_1)$ (which will be θ^l or θ^u). If the signal is the same then the arm angle feasible intervals are: $[-\pi \rightarrow \psi_1;\ \psi_2 \rightarrow \psi_3;\ \psi_4 \rightarrow \pi]$. Otherwise: $[\psi_1 \rightarrow \psi_2;\ \psi_3 \rightarrow \psi_4]$.

[22] When the function $\theta(\psi)$ crosses the limits θ^l or θ^u, it does not always crosses two times each of them. Although, the equation 3.84 will still have two solutions. As such, one of them has to be discarded. To know which one should be, the obtained ψ values can be used to calculate the correspondent θ (equations 3.60, 3.61, 3.71, and 3.72). The one which is different from θ^l or θ^u must be rejected.

2. If it is not possible to obtain any intersection between $\theta(\psi)$ and its limits (θ^l and θ^u) the calculation of θ using any ψ in the equations 3.60, 3.61, 3.71, or 3.72 will reveal if the arm angle feasible interval is $[-\pi \rightarrow \pi]$ or null.

Following the same procedure presented for the generic equation 3.76 (related to joints 1, 3, 5, and 7) with 3.77 (concerns the joints 2 and 6), first, its derivative is calculated,

$$\frac{d\theta_i}{d\psi} = -GC_k\left(\frac{a\,cos(\psi) - b\,sin(\psi)}{sin(\theta_i)}\right). \tag{3.86}$$

Subsequently, the extremes of the equation 3.77 (ψ_0) can also be obtained making $\frac{d\theta_i}{d\psi} = 0$,

$$\begin{aligned} \psi_{0,1} &= atan2(a, b), \\ \psi_{0,2} &= atan2(-a, -b). \end{aligned} \tag{3.87}$$

In the present case, two generic profiles can arise. The first is represented in Figure 3.21–a) and happens when no arm angle corresponds to a singular configuration. In the opposite situation (when a singular configuration exists for a specific arm angle), the generic profile is similar to the one in Figure 3.21–b). In both situations, $\theta(\psi)$ is a continuous function defined in $[-\pi,\ 0]$ or $[0,\ \pi]$ rad depending on GC_k.

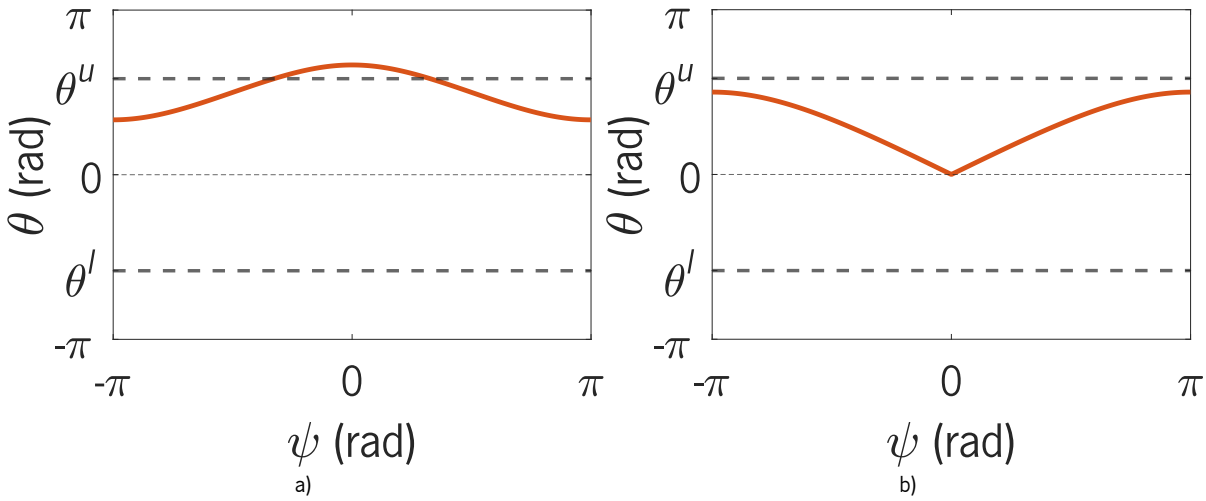


Figure 3.21 – $\theta(\psi)$ generic profile for joints 2 and 6: a) without singular configuration; b) with singular configuration.

To test if the $\theta(\psi)$ profile has a singular configuration, one should recall from Section 3.3.1 that singularities happen when the joints 2 or 6 are in the zero position. As the equation 3.77 is always between 0 and π rad, or $-\pi$ and 0 rad (depending on the GC_k) one knows that if a singular configuration exists it corresponds to an extreme of the $\theta(\psi)$ profile. As such, one can test the ψ values that result from making the derivative of the profile equal to zero ($\psi_{0,1}$ and $\psi_{0,2}$ obtained with the equations from 3.87) using the following condition (because when θ is zero $a\,\sin(\psi) + b\,\cos(\psi) + c$ becomes equal to 1),

$$\psi_{sing} = \begin{cases} \psi_{0,i} \; if \; \left| \left| a \, sin(\psi_{0,i}) + b \, cos(\psi_{0,i}) + \; c \right| - 1 \right| \leq 5 \times 10^{-3}, i \in \{1,2\}, \\ nonexistance \; otherwise. \end{cases} \tag{3.88}$$

A simple strategy to avoid singularities consists in excluding the intervals containing the singular arm angles from the feasible intervals, using a safety margin δ (e.g. 7°): $\left[\psi_{sing} - \delta, \psi_{sing} + \delta\right]$. The arm angle feasible intervals for the joints 2 and 6 can be determined with a procedure similar to the one presented for joints 1, 3, 5, and 7. However, for the joints 2 and 6, the arm angle values where the $\theta(\psi)$ profile crosses the joint limits (θ^l or θ^u), if they exist, are given by,

$$\psi = 2arctan\left(\frac{a \pm \sqrt{a^2 + b^2 - (c - cos\,(\theta))^2)}}{cos(\theta) + b - c}\right). \tag{3.89}$$

The feasible intervals of the different joints (excluding joint 4 because it does not depend on the arm angle) must be intersected, giving rise to a final group of feasible intervals where one knows that the studied joints are inside their limits and singularities are avoided (Figure 3.22).

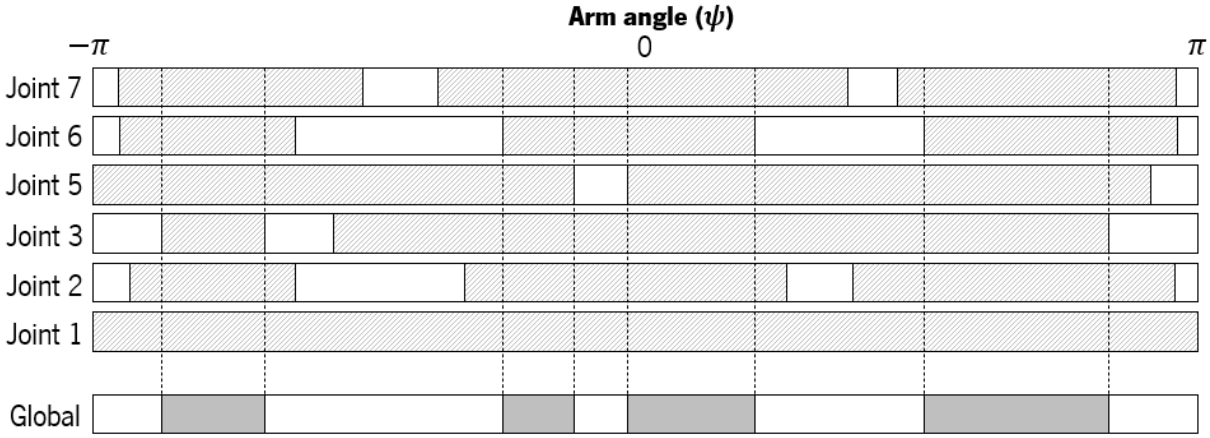


Figure 3.22 – Intersection of the feasible intervals of each joint originating a group of global feasible intervals.

The following section extensively uses the resolutions of the kinematic problems presented in this chapter to solve a trajectory tracking problem considering manipulability. The feasible intervals are used not only to avoid joint limits and singularities but also to increase the distance to them, i.e., to increase manipulability. Also, the IK resolution is widely used in the mapping from the Cartesian to the joint space. FK is used in the validation of the proposed method.

4. TRAJECTORY TRACKING CONSIDERING MANIPULABILITY

This section introduces a novel trajectory tracking approach that increases manipulability. Then, the results obtained with it are studied considering the distance to joint limits and singularities. Finally, a validation of the approach in simulation and with the real robot is presented.

4.1 Method description

This section explains the proposed method. The theoretical aspects presented in the previous chapter constitute the foundations of the present one. An overview of this method, which has three main steps (1 - Scanning, 2 - Arm angle series, 3 - Arm angle series choice), can be seen in Figure 4.1.

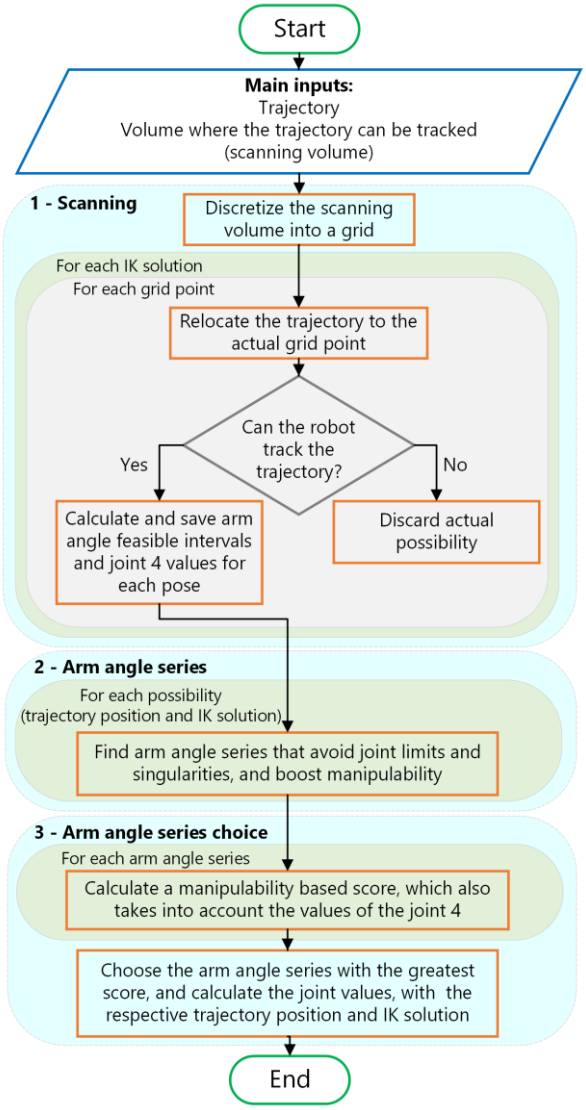


Figure 4.1 – Flowchart with the main steps of the proposed method.

4.1.1 Scanning

The first step of the presented approach is to scan the volume where one considers that the robot can track the trajectory. This work selects the scanning volume considering the intersection between the active and passive manipulator's workspace (the active manipulator is responsible for performing the inspection movements, and the passive has a holding function). Still, in other applications, the scanning volume can be defined by taking into account the requirements of the specific task. This scanning volume is discretized, giving rise to a grid, where the points are equally spaced in all dimensions (50 mm in the case of this dissertation, but this value can be refined if there is a need) (Figure 4.2).

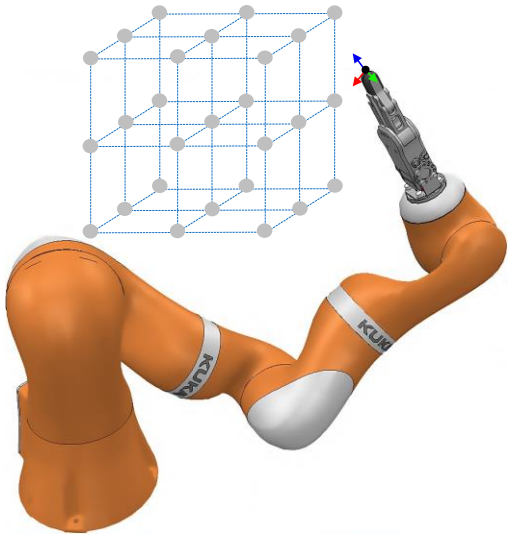


Figure 4.2 – Manipulator's workspace zone discretization, giving rise to a grid (image for exemplification purposes, it does not consider accurate dimensions and the existence of a passive manipulator).

Then, the trajectory is sequentially relocated so that the position of the first pose coincides with each grid point, as shown three times from Figure 4.3–a) to c). In this figure, the frames represent the poses to be achieved sequentially and define the trajectory.

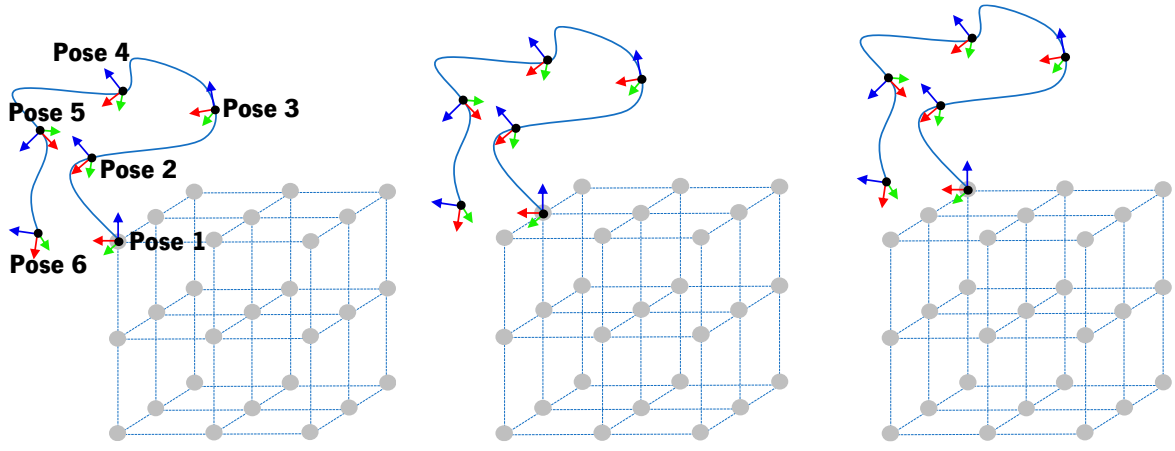


Figure 4.3 – Exemplification of three relocalization steps (image for exemplification purposes, it does not consider accurate dimensions, and the trajectory is not real).

For every trajectory location and all the IK solutions, the feasible intervals (which avoid joint limits and singularities) are calculated for each pose (using the procedure presented in the subchapter 3.3) and sequentially represented. For example, relative to the trajectory presented in Figure 4.3–a), for a specific IK solution and place of the trajectory, one could determine and then represent the feasible intervals for each of the six poses as depicted in Figure 4.4 (in **gray**). From here, this type of representation will be called a "map" because it allows finding arm angle series ("paths" as the **orange** one in Figure 4.4), which make the trajectory execution possible (with increased manipulability). This section is more focused on maps and the following on paths.

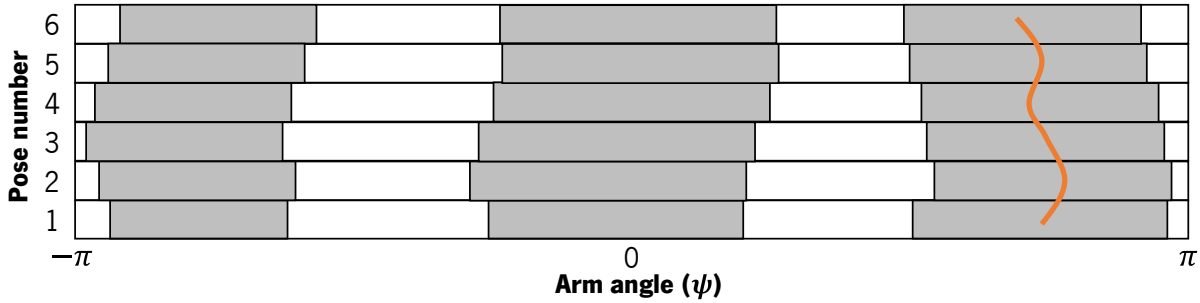


Figure 4.4 – Representation of the arm angle feasible intervals (in gray) for the consecutive poses of a trajectory (image for exemplification purposes, the intervals are not real).

This scanning procedure is crucial for two reasons:

1. It allows searching for continuous maps with wide intervals to place the arm angle farther from its limits in each pose (because it is known that in the boundaries of the intervals, there is a joint limit or singularity, which should be kept far because they decrease manipulability).
2. If the trajectory is kept in the same place, joint 4 can not be used to influence manipulability because its absolute value only depends on the poses (3.47). Thus, independently of the arm angle and IK solution, the absolute distance to its singularity (0°) and joint limit (±120°) will not change. Changing the place where the trajectory is reproduced allows to change the absolute values of the joint 4, and choose the most convenient ones.

Mathematically, the trajectory to be tracked can be represented in a matrix: each line corresponds to a pose; the first three columns match the position – x, y, and z coordinates; the last three columns concern the orientation – Euler angles[23] α, β, and γ, which give rise to the desired orientation matrix, with,

$$^{0}R_{7} = Rot(x,\alpha)Rot(y,\beta)Rot(z,\gamma). \quad (4.1)$$

[23] From Euler's rotation theorem, it can be drawn that there is no need to use more than three rotations about the coordinate axes to obtain the orientation of any frame relative to another. It should be noted that successive rotations can not concern the same axis. Thus, there are three possibilities for the first rotation, and the second and third have both two possibilities. This gives rise to a total of twelve groups of Euler angles. This thesis uses the XYZ group because it is the one that was chosen by the creators of CoppeliaSim, which is the used robotics simulator (Corke, 2013).

Euler angles are used because they only require three values, and the absolute orientation matrix requires nine.

At the beginning of the proposed method, the received trajectory (matrix $trajectory$) is placed so that the position of the first pose becomes $(0, 0, 0)$ relative to the robot's base frame. To achieve this (matrix $trajectoryIn000$), one can use the following approach, considering that x_1, y_1, and z_1 give the first pose's position of the received trajectory (: is a notation simplification which means "in all rows" or "in all columns"),

$$trajectoryIn000[:, 1] = trajectory[:, 1] - x_1, \tag{4.2}$$

$$trajectoryIn000[:, 2] = trajectory[:, 2] - y_1, \tag{4.3}$$

$$trajectoryIn000[:, 3] = trajectory[:, 3] - z_1. \tag{4.4}$$

Then, to place the trajectory so that its first pose's position coincides with the desired grid point (with coordinates x, y, and z), a similar procedure can be applied,

$$trajectory[:, 1] = trajectoryIn000[:, 1] + x, \tag{4.5}$$

$$trajectory[:, 2] = trajectoryIn000[:, 2] + y, \tag{4.6}$$

$$trajectory[:, 3] = trajectoryIn000[:, 3] + z. \tag{4.7}$$

For every trajectory location and all the IK solutions, maps like the one shown in Figure 4.4 are built from the bottom up, pose by pose. Although, if one of the following conditions is detected, the map generation is canceled, being the map discarded:

1. Pose outside the reachable workspace (condition given by 3.40).
2. Elbow singularity (joint 4 near zero position - condition given by 3.48)[24].
3. Joint 4 out of limits (absolute angular value higher than 120° - angular values given by equation 3.47).
4. Impossibility of obtaining a set of continuous values for the arm angle, from the beginning to the end of the map, i.e., when any interval of the current pose does not intersect an interval of the previous pose that has had an intersection with the intervals of the pose prior to the latter. For example, in Figure 4.5–a) when the actual pose is i, the respective feasible intervals are intersected with the interval of pose $i - 1$ which has the **blue** stripes (because it is the only one that intersects the feasible interval of pose $i - 2$). As the interval with the **green** stripes intersects the latter (the one with the **blue** stripes), it is saved to be intersected with the intervals of the next pose ($i + 1$) and test if a continuous arm angle variation keeps being possible. Figure

[24] Both the condition 1 and 2 refer to cases in which the arm is stretched.

4.5–b) shows the opposite situation because it is not possible to intersect any of the feasible intervals of the pose i with the interval with **blue** stripes of the pose $i-1$. In these cases the generation of the map is canceled, and it is discarded. When one is dealing with a map's first pose this test is not done, and all the intervals are saved to be used in the testing of the pose 2.

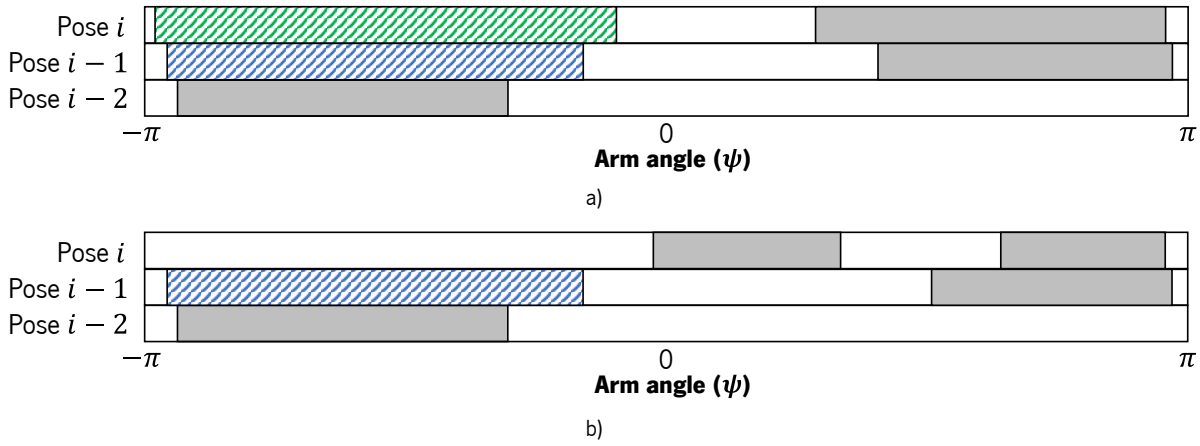


Figure 4.5 – Maps continuity test: a) example in which the intervals of the actual pose (i) do not invalidate a continuous variation of the arm angle; b) opposite situation to a).

5. Wrist's root (intersection of the last three joints' axes of the manipulator) is the direction of the joint 1 axis, or close to it. In this case, the map generation is canceled mainly because an abrupt joint variation can arise. In the initial stage of the IK method, the θ_1^v value is calculated using the x and y projections of the wrist's root and the atan2 function (equation 3.31 and Figure 3.7). This means that if, for instance, a trajectory forces the wrist's root to move from the first to the third quadrant (of the xy plane), the θ_1^v value will at a certain point have an abrupt variation of about 180°. The latter change will be directly related to the virtual manipulator. Still, as it is the base for determining the joint angles in the real manipulator, the latter is inevitably affected.

At the end of this step, there will be a group of maps like the one represented in Figure 4.6, which respect the conditions presented above. Each of these will be related to a known joint 4 variation. The following section will be dedicated to the paths that allow crossing the obtained maps with increased manipulability.

4.1.2 Arm angle series

To establish the paths (arm angle series) that cross the obtained maps, one must consider the goal of increasing manipulability. As presented in Section 2.2.2, the manipulability is negatively affected by the proximity to joint limits and singularities, and one knows that these latter are present in the feasible intervals' boundaries. As such, it makes sense that some works tend the arm angle values toward the

midpoints of these intervals (to increase manipulability) (Dou et al., 2022; Wiedmeyer et al., 2021). This will also be the approach used in this section.

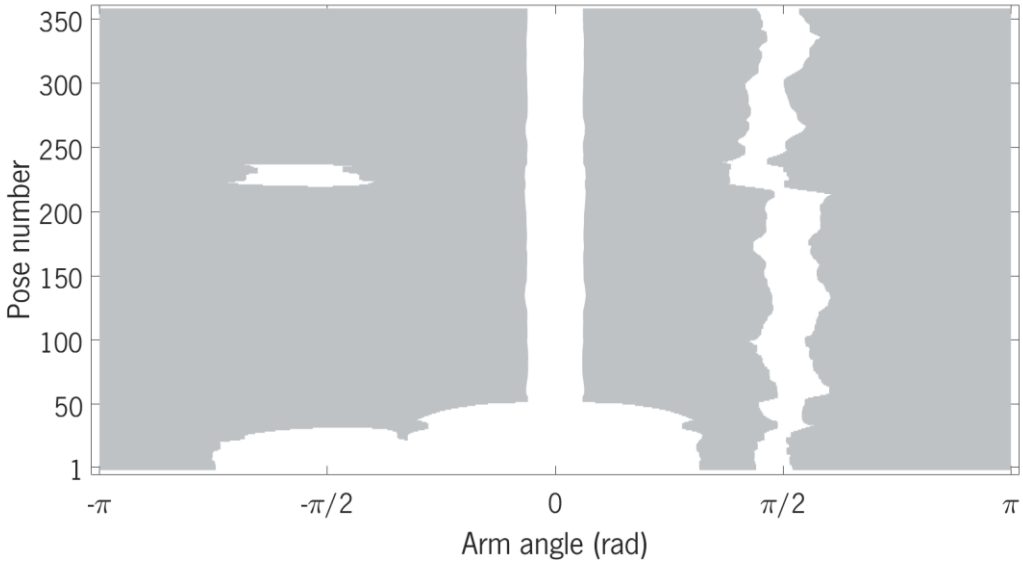


Figure 4.6 – Example of a map that results from the scanning step (for a specific solution of the IK and workspace position).

The definition of the paths is divided into two major steps: first, the possible paths' starting points for each map are determined, considering the centers of the feasible intervals for the first pose (black circle and triangle of Figure 4.7); second, from the starting points, paths are established, guided by the midpoints of the intervals (orange and blue paths of Figure 4.7).

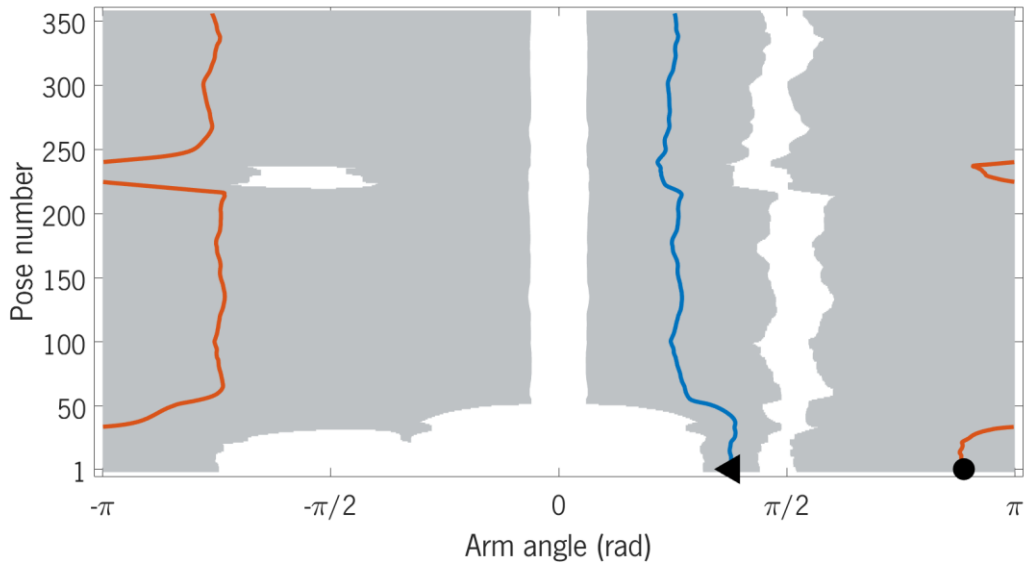


Figure 4.7 – Two major steps used in the present section: 1 – Definition of the paths' starting points (black triangle and circle); 2 – Paths established from the previously defined starting points (in orange and blue).

To better understand the first step, one should consider Figure 4.8 and recall that the method presented in Section 3.3 gives the arm angle feasible intervals in a one-dimensional array, with values between $-\pi$ and π, where each value is a boundary of a feasible interval. For the feasible intervals present in Figure

4.8 the returned array would be $[-\pi, -\frac{\pi}{2}, -\frac{\pi}{4}, \frac{\pi}{4}, \frac{3\pi}{4}, \pi]$. It should be noted that although the method presented in Section 3.3 returns the intervals in the domain from $-\pi$ to π, at the $-\pi/\pi$ boundary there is no discontinuity, i.e., the arm angle can cross it without abrupt joint variations. This finding is proven in the same section, in the study of the joints variation with the arm angle. Thus, in the example given in Figure 4.8 the intervals $\left[-\pi, -\frac{\pi}{2}\right]$ and $\left[\frac{3\pi}{4}, \pi\right]$ are actually the same.

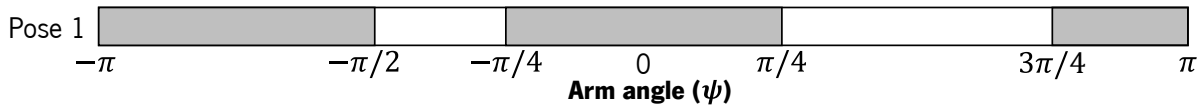


Figure 4.8 – Different cases that can arise in the definition of the arm angle series' initial values.

Considering the previously presented explanation, determining the centers of the intervals that do not cross the $-\pi/\pi$ boundary is straightforward. They can be obtained simply by calculating the mean between the boundary values of the intervals. For instance, the center of the interval $\left[-\frac{\pi}{4}, \frac{\pi}{4}\right]$ present in Figure 4.8 could be determined by,

$$\frac{\left(-\frac{\pi}{4} + \frac{\pi}{4}\right)}{2} = 0.$$

On the other hand, when the feasible interval crosses the $-\pi/\pi$ boundary, it is impossible to use such a direct approach. Thus, first, the interval is converted in a way that where it was considered the $-\pi/\pi$ boundary now will be placed the origin of the arm angle domain. For the interval $\left[-\pi, -\frac{\pi}{2}\right]$ and $\left[\frac{3\pi}{4}, \pi\right]$ of Figure 4.8, this origin change can be observed in Figure 4.9. The bottom values in this figure are the original and the upper ones result from the origin change.

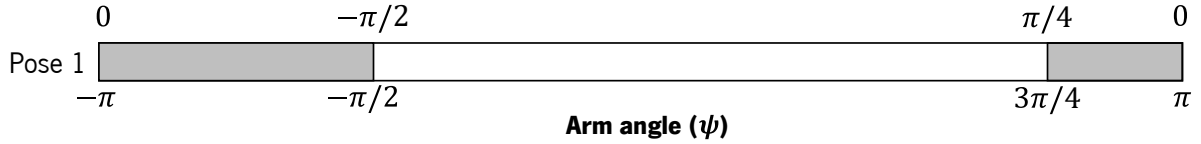


Figure 4.9 – Origin change to consider the lateral continuity in the $-\pi/\pi$ boundary.

The upper values are calculated using the differences between the ones of the bottom. In the example of Figure 4.9,

$$-\pi - \left(-\frac{\pi}{2}\right) = -\frac{\pi}{2},$$

$$\pi - \frac{3\pi}{4} = \frac{\pi}{4}.$$

With this scale, it is already possible to obtain the middle of the interval using the mean (Figure 4.10),

$$\frac{\left(-\frac{\pi}{2}+\frac{\pi}{4}\right)}{2}=-\frac{\pi}{8},$$

which can be reconverted to the original scale by making (Figure 4.10),

$$-\pi-\left(-\frac{\pi}{8}\right)=-\frac{7\pi}{8}.$$

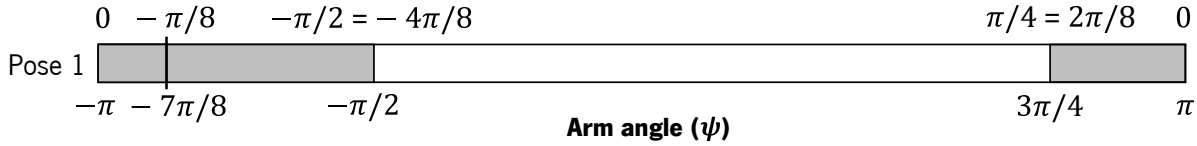


Figure 4.10 – Calculation of the center of an interval considering the lateral continuity in the $-\pi/\pi$ boundary.

The procedure exposed above allowed to calculate the center points of the intervals presented in Figure 4.8 (the possible paths' starting points, represented as $\psi_{(t-1,1)}$ and $\psi_{(t-1,2)}$ in Figure 4.11), but it can be generalized for different cases as shown next in pseudocode.

Arm angle values calculation for the first pose

```
for each map
    allow ← boundaries of the feasible intervals in the first pose
    if allow(1) is −π and allow(end) is π
        neg ← −π − allow(2)
        pos ← π − allow(end−1)
        meanValue ← mean(neg, pos)
        if meanValue ≥ 0
            armAngle(1) ← π − meanValue
        else
            armAngle(1) ← −π − meanValue
        end
        if length(allow) > 4
            for(j = 3; j ≤ length(allow) – 2; j = j + 2)
                armAngle(end + 1) ← mean(allow(j), allow(j + 1))
            end
        end
    else
        for(j = 1; j ≤ length(allow); j = j + 2)
            armAngle(end + 1) ← mean(allow(j), allow(j + 1))
        end
    end
end
```

For every map, each of the determined arm angle values is used as a starting point for a path, which one establishes from the bottom up, pose by pose. Going back to the example of Figure 4.8, where one has

two initial arm angle values (as represented in Figure 4.11: $\psi_{(t-1,1)}$ and $\psi_{(t-1,2)}$), there is now the need to obtain the arm angle values for pose 2. Starting with the more straightforward case, $\psi_{(t-1,2)}$, the first step is to find the feasible interval of pose 2 that contains $\psi_{(t-1,2)}$. This interval is represented in green in Figure 4.11.

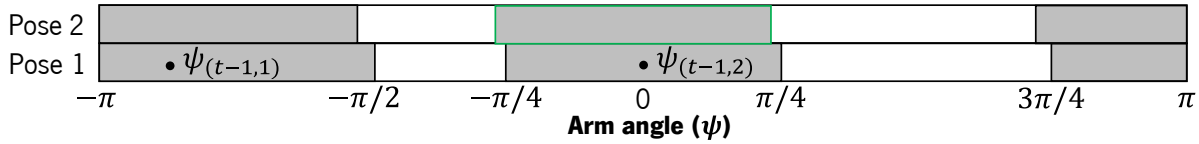


Figure 4.11 – Growth of the paths from the arm angle values found for pose 1.

With this interval and with the previous arm angle (in this case $\psi_{(t-1,2)}$), is possible to use the following equation (proposed by Faria et al. (2018)) to determine the arm angle for pose 2,

$$\psi_{(t)} = \psi_{(t-1)} + K\left(\frac{\psi^u - \psi^l}{2}\right)\left[\mathrm{e}^{-\alpha\left(\frac{\psi_{(t-1)}-\psi^l}{\psi^u-\psi^l}\right)} - \mathrm{e}^{-\alpha\left(\frac{\psi^u-\psi_{(t-1)}}{\psi^u-\psi^l}\right)}\right], \tag{4.8}$$

where,

- $\psi_{(t)}$ is the arm angle to be determined.
- $\psi_{(t-1)}$ is the arm angle from the previous pose (in this case $\psi_{(t-1,2)}$).
- ψ^l and ψ^u are the lower and upper limits of the current pose feasible interval that contains $\psi_{(t-1)}$ (for the present example is the green one from Figure 4.11), respectively.
- $K \in [0, 1]$ is a constant that controls the strength of repulsion from the interval limits.
- $\alpha \in \mathbb{R}^+$ is a constant that controls the distance to the borders from where the repulsion begins.

When using this equation, the parameters K and α are tuned so that the value of the arm angle powerfully tends to the center of the feasible intervals. Thus, a high K and a low α are used ($K = 1$, $\alpha = 7$). A lower value for α is not used because that would make the arm angle control overly reactive, which could negatively affect the smoothness of the movements. This tuning is an advantage of using this equation over directly using the average value of the intervals. Despite this, when the feasible intervals expand or shrink rapidly, this combination of parameters can cause an abrupt variation in the arm angle. As such, the variation is limited to 0.1 radians. This procedure allows finding an arm angle value for the pose 2 of Figure 4.11 ($\psi_{(t,2)}$ in Figure 4.12), which, together with the arm angle $\psi_{(t-1,2)}$, would constitute a segment of a path that not only avoids joint limits and singularities but also tries to increase the distance to them. Furthermore, the variation between the arm angles is smooth. If this method is used iteratively, from pose to pose, a path that crosses a whole map can be found.

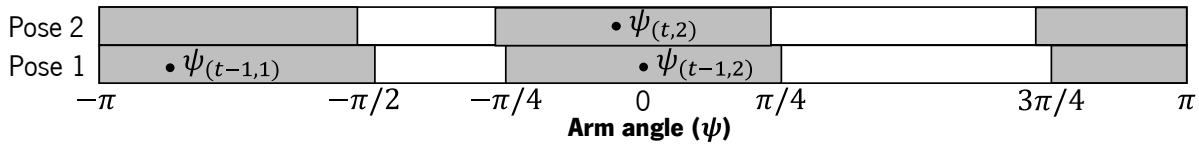


Figure 4.12 – Determination of the arm angle value of a pose (pose 2) based on the arm angle of the previous pose (pose 1), considering manipulability and smoothness.

It should be noted that when the feasible interval crosses the $-\pi/\pi$ boundary (first interval of pose 2 in Figure 4.12), the values of $\psi_{(t-1)}$, ψ^l, and ψ^u cannot be directly inserted in equation 4.8. This is because one does not have ψ^l and ψ^u in the format of a single interval, and thus, an origin change is needed (similar to the one presented in the calculation of the intervals' centers for the first pose). Consequently, $\psi_{(t-1)}$ needs to be converted to take into consideration the new origin as well. After this, one can use equation 4.8, but the result must be reconverted because all the other procedures are based on the domain from $-\pi$ to π. The following pseudocode summarizes these three steps (origin change; calculation of $\psi_{(t)}$ using equation 4.8; origin reconversion).

Path generation when the feasible intervals cross the $-\pi/\pi$ **boundary**

1: allow ← boundaries of the feasible intervals for the actual pose
2: $\psi_{(t-1)}$ ← previous arm angle, which is contained by the intervals of the actual pose
3: neg ← $-\pi$ − allow(2)
4: pos ← π − allow(end−1)
5: **if** $\psi_{(t-1)} \geq -\pi$ and $\psi_{(t-1)} \leq$ allow(2)
6: $\quad \psi_{(t-1),conv} \leftarrow -\pi - \psi_{(t-1)}$
7: **else**
8: $\quad \psi_{(t-1),conv} \leftarrow \pi - \psi_{(t-1)}$
9: **end**
10: **Calculate** $\psi_{(t),conv}$ using equation 4.8 ($\psi_{(t-1)}$ is $\psi_{(t-1),conv}$; ψ^l is neg; ψ^u is pos)
11: **if** $\psi_{(t),conv} \leq 0$
12: $\quad \psi_{(t)} \leftarrow -\pi - \psi_{(t),conv}$
13: **else**
14: $\quad \psi_{(t)} \leftarrow \pi - \psi_{(t),conv}$
15: **end**

The procedures presented in this section give rise to a group of paths (arm angle series), like those represented in blue and orange in Figure 4.7, that allow solving the maps determined in Section 4.1.1 (Scanning stage). These paths are guided by the centers of the intervals because this approach tends to boost manipulability. Still, it can fail to solve some maps. Despite this, those usually have interval limits appearing abruptly near the current arm angle (Figure 4.13). This case is dangerous because joint limits or singularities will be close, but they will not be reflected in the arm angle feasible intervals. In these

cases, it is even convenient that the approach fails. When it does not, the chosen arm angle will be close to limits, which decreases, at least momentarily, the manipulability.

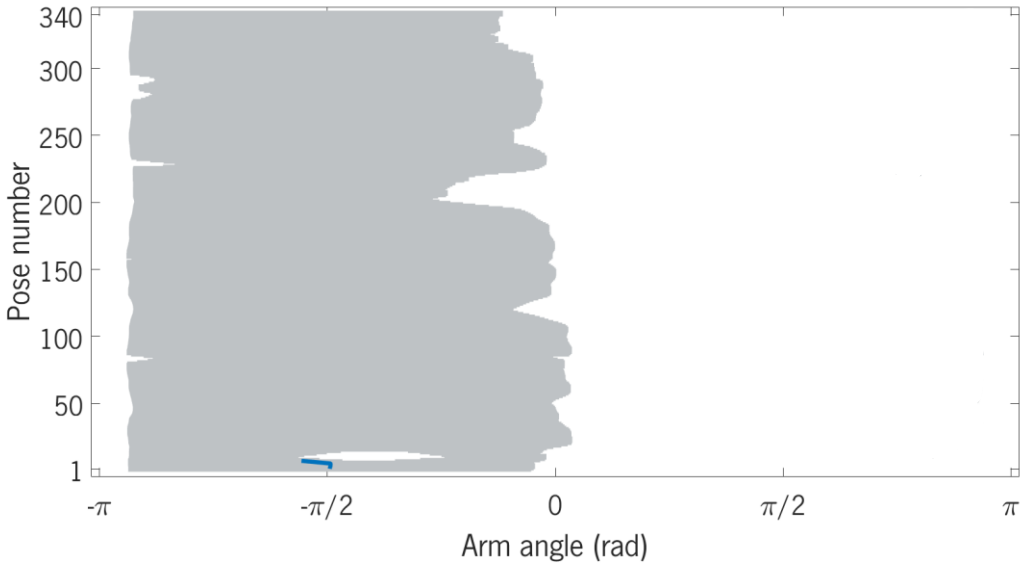


Figure 4.13 – Example of a situation in which joint limits appear abruptly near the path that was being generated (represented in blue).

4.1.3 Arm angle series choice

With the different possibilities of arm angle series determined, it is now necessary to choose one that boosts manipulability, also considering the joint 4 values associated with these series (this joint is independent of the arm angle, thus has to be considered separately). Starting specifically with them (arm angle series/paths), the first step is to calculate the minimum distance, in each pose, of the calculated arm angles to the boundaries of the feasible interval in which they are contained. As the centers of the intervals guide the paths, the distance to the two boundaries is usually similar (black arrows in Figure 4.14). Although, it will only be the same in the first pose because, in the others, the centers are not strictly followed to avoid abrupt variations and an overly reactive control (as explained in Section 4.1.2). Thus, the shortest distances are saved for safety in each pose, and then the minimum (ψ_{min}) and mean (ψ_{mean}) values of all the minimal distances are calculated (this procedure is done for all paths).

A similar procedure must be applied to the joint 4 values associated with each path. For all poses, the distance to its limit (120°/-120°) and singularity (0°) is calculated, and the minimum value of these two is saved. This procedure gives rise to a set, and its minimum ($\theta_{4\,min}$) and mean ($\theta_{4\,mean}$) values are calculated. With the data of ψ_{min}, ψ_{mean}, $\theta_{4\,min}$, and $\theta_{4\,mean}$ for each path, one can attribute a score (ms) to all using the following metric,

$$ms = \frac{\psi_{min}}{max(\psi_{min})} + \frac{\psi_{mean}}{max(\psi_{mean})} + \frac{\theta_{4\,min}}{max(\theta_{4\,min})} + \frac{\theta_{4\,mean}}{max(\theta_{4\,mean})}. \quad (4.9)$$

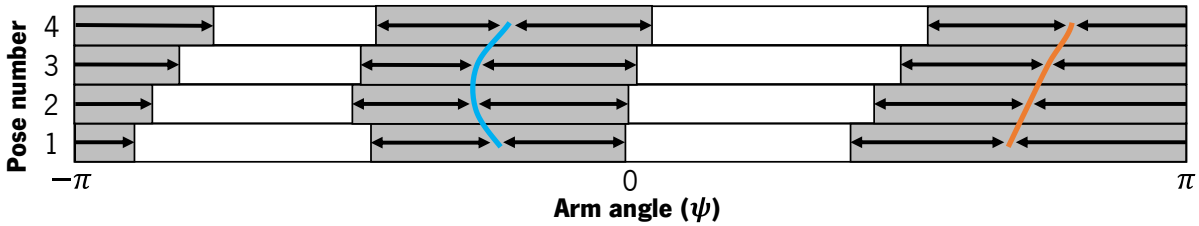


Figure 4.14 – Distance of the paths to the boundaries of the feasible intervals.

Concerning the proposed metric, some aspects can be presented in more detail:

- Both the minimum and mean values are used in this equation because, if one used only the mean, extensive distances could hide narrow ones. This would compromise the goal of increasing manipulability because joint values would be close to the limits or singularities in certain poses.
- The division for the maximum values intends to make equal the influence of each term. Because the values related to ψ tend to be greater than the values associated to θ_4, a less significant variation in ψ could easily match a higher one in θ_4.
- One could argue that different equation terms should have different impacts, and weights could be used for that purpose, but, as will be seen in the next section, this tunning is not necessary to obtain good results.
- For safety, when the terms $\frac{\psi_{min}}{max(\psi_{min})}$ and $\frac{\theta_{4\,min}}{max(\theta_{4\,min})}$ are less than 0.3, the path is not considered.

The proposed metric tends to give the higher scores to paths like the one in Figure 4.15, associated with joint 4 values like those depicted in Figure 4.16. This is because both the paths and the joint 4 keep consistently a high distance to joint limits and singularities (increasing manipulability).

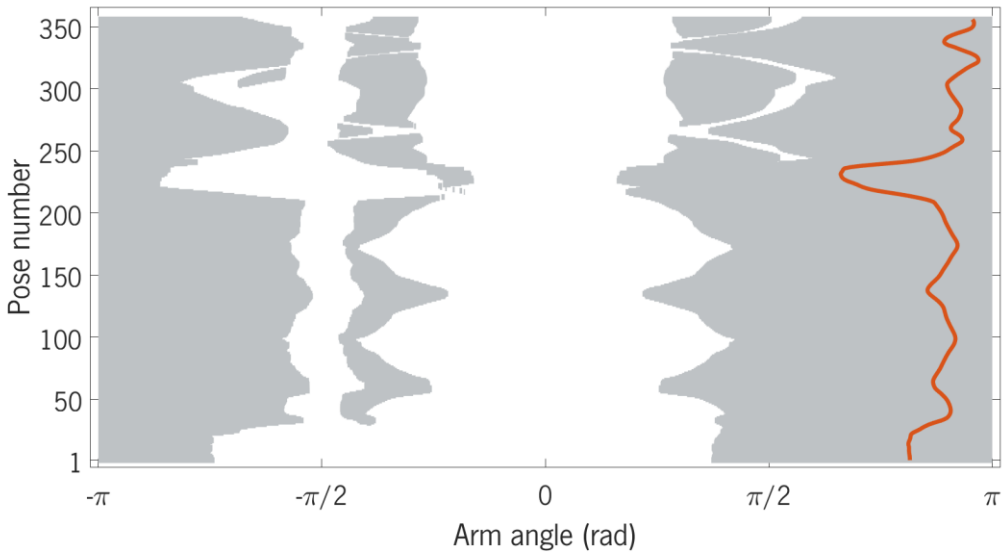


Figure 4.15 – Example of a path that the proposed metric tends to choose.

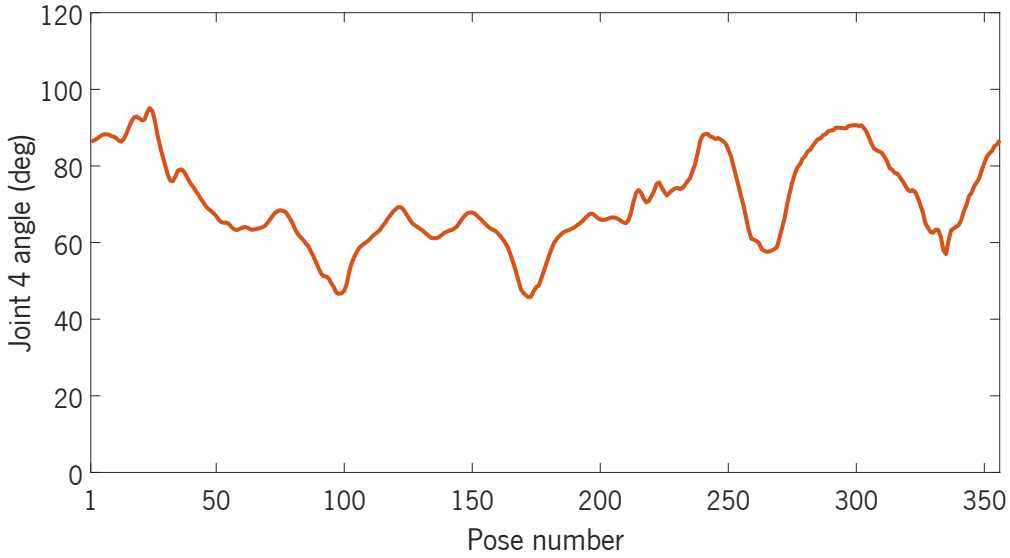


Figure 4.16 – Example of joint 4 values that the proposed metric tends to choose.

A score is calculated for each path. The one with the highest score, associated with a trajectory position and IK solution, is selected. Then, the joint values to be sent to the robot's controller are generated using the IK resolution presented in Section 3.2. With the proposed method explained, it is now convenient to present some tests to verify if the originated joint values can track the acquired trajectory and keep a high manipulability consistently. This will be the focus of the next section.

4.2 Method validation

This section is dedicated to testing the trajectory tracking method proposed in the previous subchapter. First, the generated joint values are analyzed to verify if they consistently keep a high distance to the limits and singularities. Then, the trajectory that one obtains with these joint values is compared to the desired one, both in simulation and in the real world. Also, with the real manipulator, the deformations induced in the leather pieces are compared with the ones that are needed. Although the trajectories acquisition is not in the scope of this work, before presenting all the other contents, a brief presentation of how the trajectories are obtained will be exposed for contextualization purposes.

4.2.1 Trajectories acquisition

The computer vision system described in this section was developed in the same project as this dissertation but gave rise to a different one (Ribeiro, 2022). This system uses information of the hands to generate the positions and of the leather pieces to obtain the necessary orientations. Regarding the position, first, it extracts the 3D coordinates of 21 key points (relative to the camera). Then, one of these

points is used for hand position tracking. To ensure that the selected point is always close to the leather pieces and does not tend to be occluded, some restrictions are imposed on the hand, namely by considering a fixed configuration that the demonstrator must fulfill (Figure 4.17).

Figure 4.17 – Hand posture that must be used in the demonstration.

In terms of orientation, as it was necessary to resort to the information about the leather piece configuration during the manipulation, a 3D tracking system was used (Figure 4.18) to obtain a mesh representing the object's surface (Figure 4.19). After this, the gradient was taken in the direction of the object's length and width on the points near the grasping position. With these gradients, a coordinate system was constructed (Figure 4.19), and the corresponding orientation matrix was computed. The latter was corresponded manually to the referential used in the extremity of the end-effector.

Figure 4.18 – 3D tracking system used in the orientations acquisition.

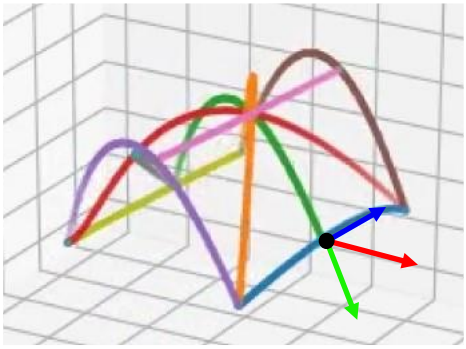


Figure 4.19 – Object surface reconstruction and obtained frame for determining the orientations.

This computer vision system made it possible to obtain two different types of inspection movements: the first bends concavely or convexly the object and moves its concavity extremity along its length; the second twists it on different axes. The obtained trajectories were exported in the format described in Section 4.1.1 (matrix in which each line corresponds to a pose; the first three columns match the position – x, y, and z coordinates; the last three columns concern the orientation – Euler angles), and were used to obtain the results described in the following sections.

4.2.2 Manipulability analysis

To obtain the results presented in this section, in the scanning phase (Section 4.1.1) of the developed method was used a volume (500×500×500 mm) in a convenient place to be covered by two manipulators placed side by side (centered in the coordinates (250; 250; 450) mm in respect to the robot's base referential). It is intended that the final system of the project in which this dissertation is inserted has two robotic arms: the first will be active and responsible for performing the inspection movements; the second will be passive and have a holding function (the latter can not be only a fixed support because it will have to help in the grasping task and relocate the tip to the place where the manipulability is increased).

For the first trajectory acquired with the system presented in the previous chapter, the path that obtained the biggest score ($ms = 3.75$) can be observed in Figure 4.20. It corresponds to the trajectory that has its initial point in the coordinates (500; 500; 650) mm (in respect to the robot's base referential) and to the second solution of the IK (joints 2 and 6 will be positive, and joint 4 negative). This path leads to the variations in the joint angles exposed in Figure 4.21. A summary of the distances to the joint limits (from joint 1 to joint 7 are, respectively, $\pm 170°$, $\pm 120°$, $\pm 170°$, $\pm 120°$, $\pm 170°$, $\pm 120°$, and $\pm 175°$) and singularities (happen when the joints 2, 4, or 6 are 0°) can be observed in Table 4.1. Using the latter, it can be argued that the arm maintains a high manipulability in every pose. This statement is sound

because, in the worst case, one still has a safe distance of 18.9° (happens in joint 7), and all joints are on average more than 40° away from joint limits and singularities.

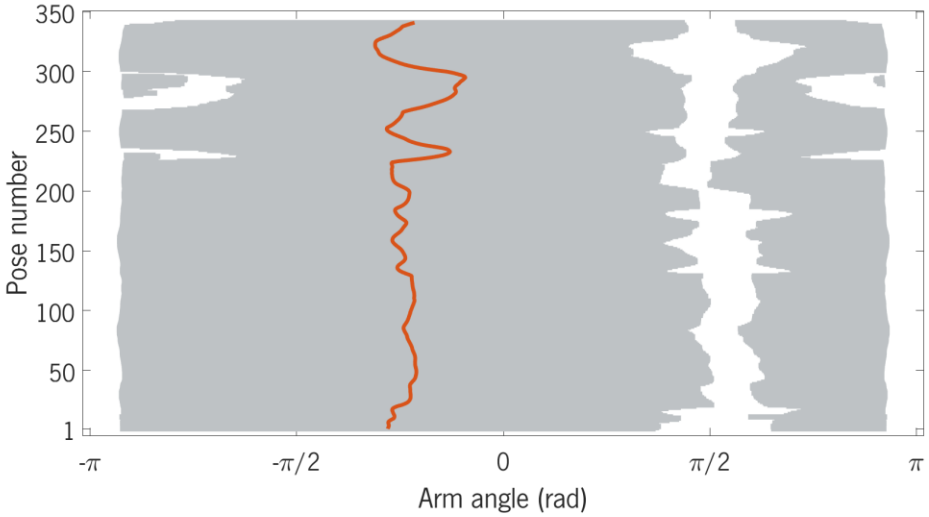


Figure 4.20 – Obtained results for the first acquired trajectory: selected arm angle series.

Table 4.1 – Minimum and average distances to joint limits and singularities obtained with the proposed method for the first trajectory tested.

Joint	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Minimum distance to joint limits/singularities (deg)	44.9	24.2	86.3	28.0	84.4	33.0	18.9
Average distance to joint limits/singularities (deg)	60.3	42.6	109.3	47.1	115.3	52.8	88.3
Standard deviation of the average distance (deg)	7.8	6.5	9.6	7.2	11.2	5.7	36.7

To give another point of view on how much the proposed method can help increase manipulability, data in a similar format to Table 4.1 is presented in Table 4.2, but for a map with narrow passages, and for an arm angle series that is not guided by the centers of the intervals (the arm angle values tend to change only when it is extremely necessary). Still, the first pose of the trajectory remains inside the previously presented scanning volume (in the coordinates (150; 300; 650) mm in respect to the robot's base referential).

Table 4.2 – Minimum and average distances to joint limits and singularities that could arise if the proposed method was not used (for the first trajectory tested).

Joint	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Minimum distance to joint limits/singularities (deg)	2.5	15.9	25.4	3.5	142.6	31.1	4.0
Average distance to joint limits/singularities (deg)	13.9	32.4	32.6	25.3	155.1	49.9	95.8
Standard deviation of the average distance (deg)	6.1	8.0	4.0	10.4	8.7	6.7	56.3

This is the type of result that one could obtain if the proposed method was not used. The problem with it is: the trajectory execution is possible, although the values of the joint 1 are highly close to the limits, negatively impacting the manipulability. Also, except for the fifth, all other joints show worse behavior than the one obtained with the proposed method.

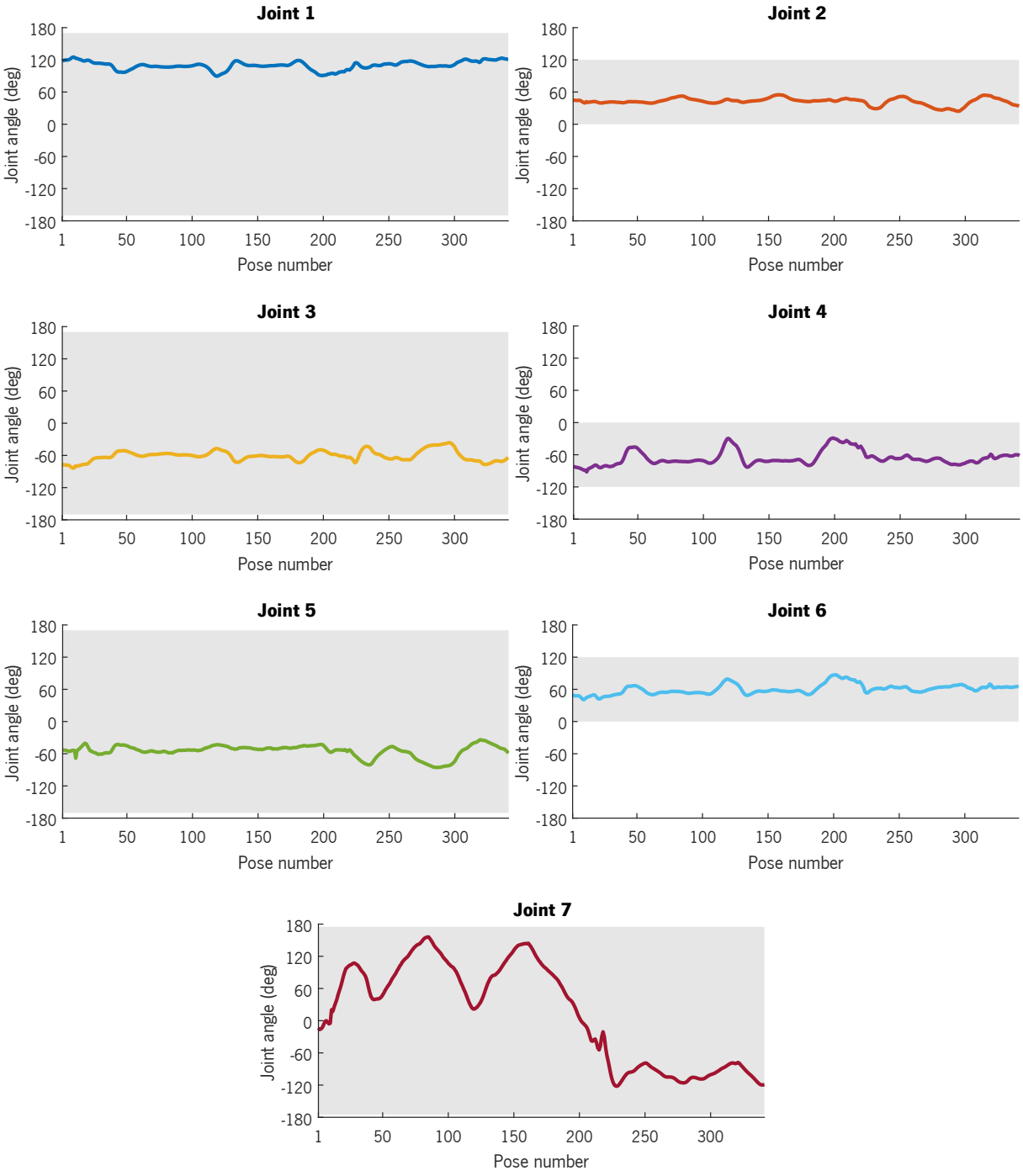


Figure 4.21 – Obtained results for the first acquired trajectory: joint values. The grey zones are the allowed ranges for the joints considering their limits (from joint 1 to joint 7 are, respectively, ±170°, ±120°, ±170°, ±120°, ±170°, ±120°, and ±175°), the manipulator's singularities (happen when the joints 2, 4, or 6 are 0°), and the IK solution (if the joints 2, 4, and 6 are positive or negative).

Relative to the second acquired trajectory, the path that obtained the best score ($ms = 3.64$) can be observed in Figure 4.22. It belongs to the trajectory that has its initial point in the coordinates (500; 500; 700) mm (in respect to the robot's base referential) and to the fourth solution of the IK (joints 2 and 4 will be positive, and joint 6 negative). This path leads to the variations in the joint angles exposed in Figure 4.23. A summary of the distances to the joint limits and singularities can be observed in Table 4.3. As in the first trajectory, it is sound to state that the joint values generated give rise to a high manipulability in every pose. In the worst case, one will still have a safe distance of 20.4° (joint 2), and all joints are on average more than 40° away from joint limits and singularities (Table 4.3).

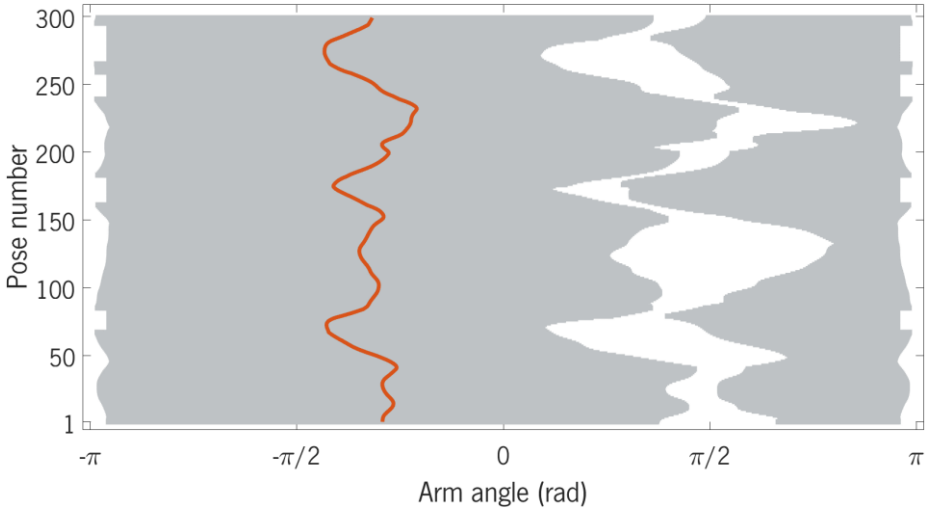


Figure 4.22 – Obtained results for the second acquired trajectory: selected arm angle series.

Table 4.3 – Minimum and average distances to joint limits and singularities obtained with the proposed method for the second trajectory tested.

Joint	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Minimum distance to joint limits/singularities (deg)	140.1	20.4	112.4	20.9	89.6	38.4	29.1
Average distance to joint limits/singularities (deg)	157.6	42.6	126.1	43.2	115.9	52.4	110.5
Standard deviation of the average distance (deg)	8.2	10.6	6.1	8.6	16.9	5.0	47.6

The results show that following the proposed approach, one can obtain a set of joint values which keep a high distance to the limits and singularities, increasing manipulability. A simulator was employed to test the robot's behavior with these joint values before using the real manipulator. This is the main topic of the following subchapter.

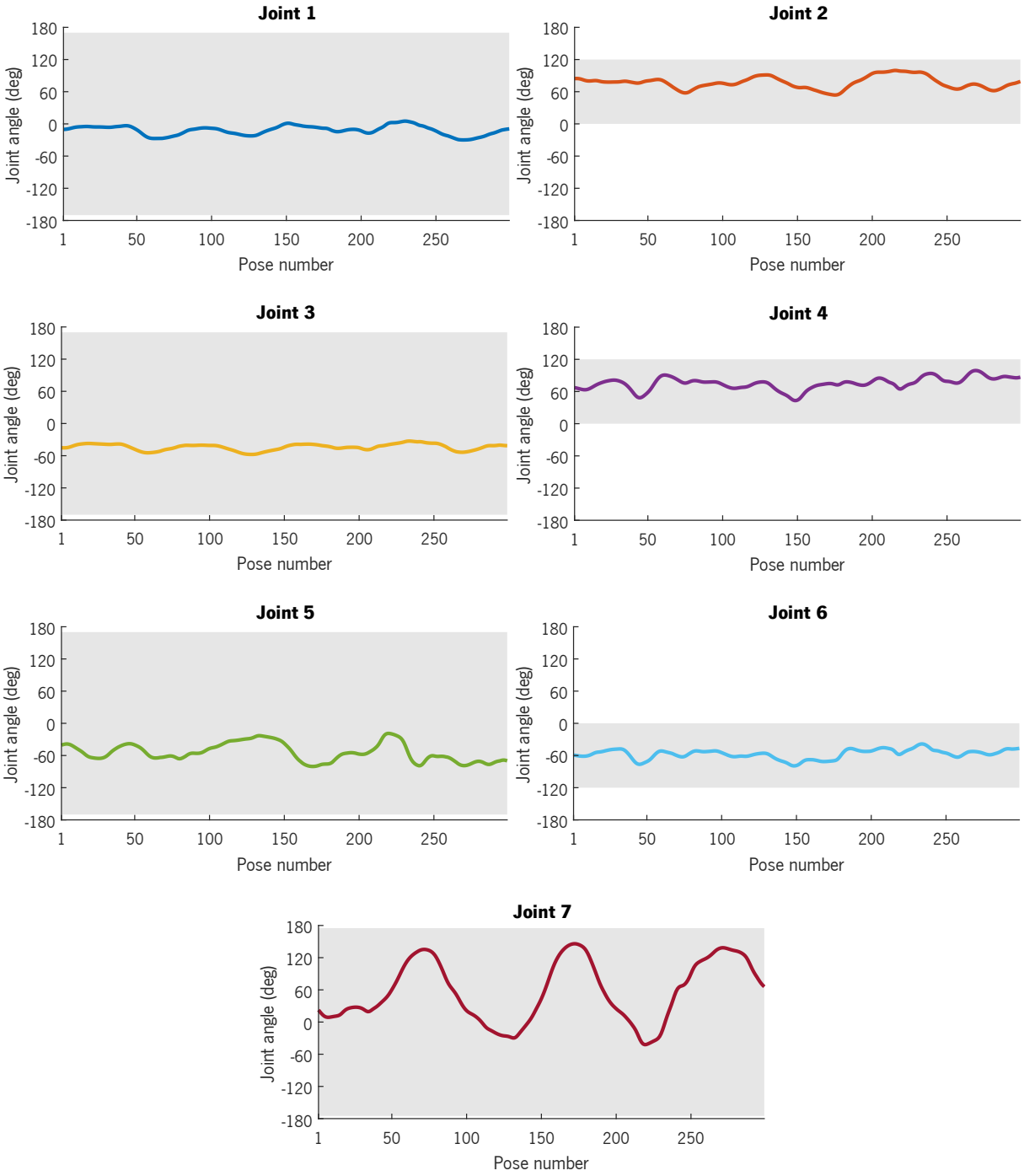


Figure 4.23 – Obtained results for the second acquired trajectory: joint values. The grey zones are the allowed ranges for the joints considering their limits (from joint 1 to joint 7 are, respectively, ±170°, ±120°, ±170°, ±120°, ±170°, ±120°, and ±175°), the manipulator's singularities (happen when the joints 2, 4, or 6 are 0°), and the IK solution (if the joints 2, 4, and 6 are positive or negative).

4.2.3 Simulation validation of the trajectory tracking problem

A robotics simulator was used to test the trajectory tracking before resorting to the real robot for two reasons:

- **Safety:** during the development of the proposed method, occasionally, joint values with abrupt variations were generated, collisions with the base on which the robot is mounted happened, or singularities were crossed (sometimes intentionally for testing purposes). All these situations could damage the real manipulator, and thus before ensuring that they would not happen, it was safer to use a simulator.
- **Convenience:** as will be seen in the following section, commanding the real manipulator requires a whole setup consisting of several pieces of equipment. Whenever a test needs to be performed, each element needs to be turned on and initialized, which can be considered cumbersome compared to using a simulator. Furthermore, it is easier to verify in simulation if the arm follows the desired trajectory, because the latter can be represented three-dimensionally in the simulation environment. To achieve the same ease with the real manipulator it would be necessary to use augmented reality.

Different robotics simulators could be used, e.g., Webots, Gazebo, and RoboDK, but the chosen one was CoppeliaSim for the following reasons:

1. CoppeliaSim is the most commonly used simulator in the laboratory where this dissertation was developed (Mobile and Anthropomorphic Robotics Laboratory (MARLab)–University of Minho). As such, a large amount of support can be obtained, and a lot of work has already been done in order to easily simulate arm movements.
2. This simulator has, by default, the models of the arm and gripper used (Kuka LBR iiwa 14 R820 and OnRobot RG2, respectively. The selection and more details about the gripper will be presented in the next section (Section 4.2.4)).
3. Trajectories can be easily imported/exported (in Comma Separated Values (CSV) files), and manipulated.
4. The robots in CoppeliaSim can be commanded using a script in Matlab, the software already used to program the proposed trajectory tracking method, which makes the writing of the needed programs fast and the debug process simple.
5. CoppeliaSim works on different operating systems, which is helpful because both Windows and Ubuntu are used in this work.

To adapt already existing CoppeliaSim simulation scenarios and Matlab files to the needs of this dissertation, the following steps were performed:

1. The files remApi.m, remoteApiProto.m, remoteApi.dll, arm_robot.m, and program_arm.m were placed in a Matlab search path to be accessible. The first three are files provided by the simulator developers and are mainly related to the communication between Matlab and CoppeliaSim. The other two were written by the research team in the MARLab. The file arm_robot.m is a class where all the needed functions are implemented (e.g., to initialize the communication between the two softwares, send commands, and read data). The program_arm.m is the script where one should write the desired command program (using the functions of the class arm_robot.m). A summary of the one used in this dissertation is represented below in pseudocode.

Strategy used to command the arm in CoppeliaSim from Matlab

```
1:  initialize arm_robot class //the communication is established here
2:  joints ← matrix with the joint values (each row respects a pose and each column a joint)
3:  dt ← 0.05 //Time increment. The default value from CoppeliaSim is 0.05 s.
4:  for (j = 1; j < number of poses)
5:    k ← 0
6:    pose_reached ← 0
7:    if j is equal to 1
8:      T ← 3 //T is the approximate time between poses in seconds.
9:    else
10:     T ← 0.5
11:   end
12:   while pose_reached is equal to 0
13:     k ← k + dt
14:     if k is not greater than T
15:       for (i = 1; i ≤ 7)
16:         armJoints(i) ← joints(j, i) + [joints(j + 1, i) − joints(j, i)]×(k/T)
17:       end
18:     end
19:     send armJoints
20:     trigger simulation
21:     joints_position ← read joints
22:     if ‖joints_position − joints(j + 1, 1:7)‖ ≤ 2.5×10^-4
23:       pose_reached ← 1
24:     end
25:   end
26: end
```

Three notes that help understand the presented pseudocode are: the first line of the matrix "joints" (line 2) gives respect to the home position of the arm and, thus, is constituted only by

zeros; the conditional statement of the line 13 intends to compensate for delays that can happen in the simulator when one is trying to achieve a pose; in line 15, linear interpolations between the joint values are done, to increase the number of values that are sent to the robot and obtain a smoother movement.

2. The research team in the MARLab already had CoppeliaSim scenarios prepared to work with the Matlab files described above, but with different manipulators and grippers. As such, the arm originally placed was eliminated, and the Kuka LBR iiwa 14 R820 was inserted. The gripper (OnRobot RG2) was then assembled to the robot's tip and manually closed (because the leather pieces have a minimal thickness). The floor's dimensions were reduced to mimic better the base on which the robot is mounted. Finally, a graph was attached to the robot's tip for two reasons: it is represented with a coordinate frame, that can be made equal to the last one of the Denavit-Hartenberg convention, allowing to understand if the desired orientations are achieved; it can leave a track of a specific color which makes possible to see if the desired positions are reached. These elements can be observed in Figure 4.24[25].

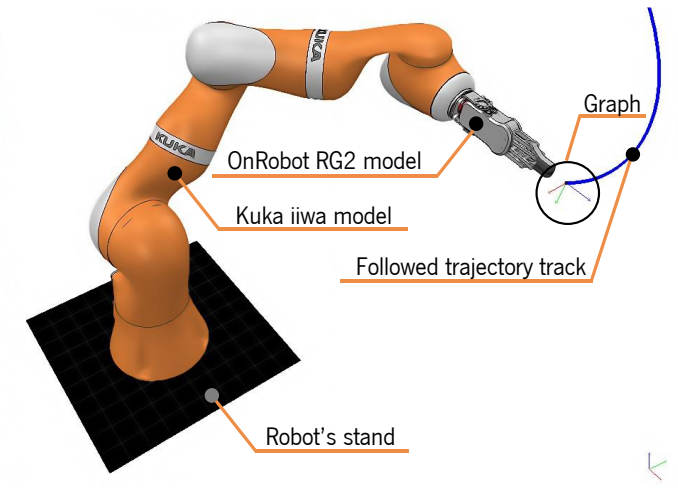


Figure 4.24 – Elements inserted in the CoppeliaSim scenario.

To simulate the tracking of a specific trajectory, first, the trajectory is imported to the scenario previously described. To achieve this, one can use the option "Import path from CSV". The selected CSV should have the trajectory's data in the format described in Section 4.1.1 (each line corresponds to a pose; the first three columns match the position – x, y, and z coordinates; the last three columns concern the orientation – Euler angles (XYZ)). An example of a trajectory imported to CoppeliaSim can be observed in

[25] In Figure 4.24 the graph is not exactly attached to the gripper's tip because the robot's flange dimension has a minor error. As such, the graph is farther from the tip to compensate for this offset and represent the actual manipulator and gripper's dimensions.

Figure 4.25–a) and b). With the desired trajectory in the scenario, one can send the joint values as explained before. During the simulation, different verifications can be done: the arm can not make abrupt movements; the line described by the graph attached to the tip of the robot must overlap the line which represents the trajectory (as in the Figure 4.25–c) the dark blue line is overlapping the light blue one); the axes of the tip's graph must overlap the frames that represent the orientations in each pose (like in the Figure 4.25–d)); the manipulator can not collide with its stand. The simulations were successful considering the aspects previously presented for the two acquired trajectories (Section 4.2.1).

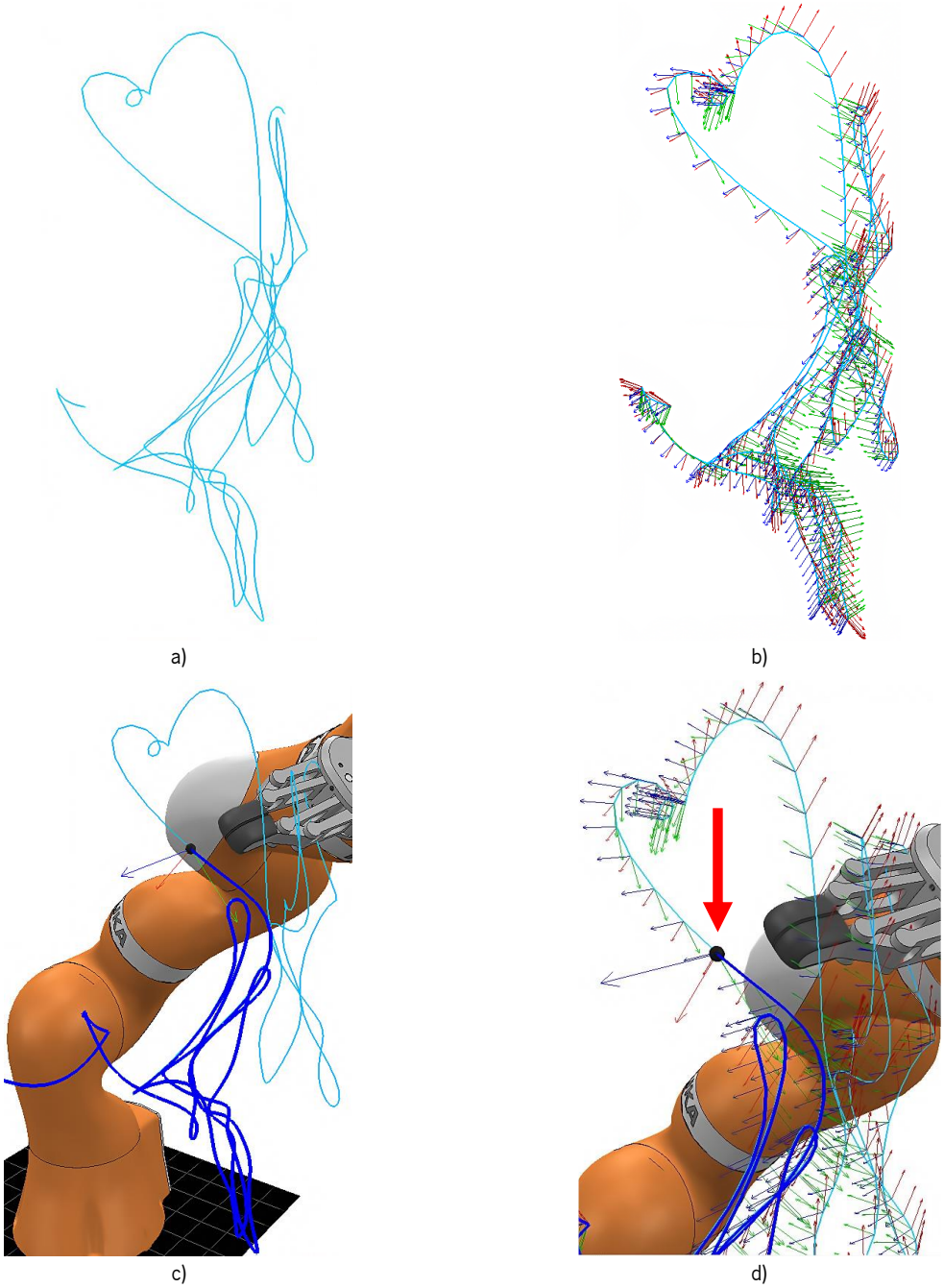


Figure 4.25 – Trajectory tracking in CoppeliaSim: a) representation of a trajectory without orientation information; b) representation of a trajectory with the orientations in each pose (represented by frames); c) position evaluation; d) orientations evaluation.
Full simulation available in: https://youtu.be/TcV9hgg4SVo.

In the trajectory tracking simulations, only a visual analysis was done because CoppeliaSim can not accurately reproduce the behavior of the arm's controller. As such, a detailed evaluation of the simulations would not be valuable. Although, they allowed to make a rough validation of the method and ensured that it was safe to test the generated joint values in the real robot, which will be the subject of the next section.

4.2.4 Practical validation of the proposed method

To validate the proposed method with the real robot was necessary to mount a testing setup. Before the beginning of the present dissertation, it was already decided that the active manipulator to be used was the Kuka LBR iiwa 14 R820. LBR stands for "leichtbauroboter", which means lightweight robot in German (weights 29.9 kg), iiwa for "intelligent industrial work assistant", 14 is the robot's payload in kilograms (Annex I), and R820 is the distance from the shoulder to the wrist in millimeters. This arm is designed for human-robot collaboration scenarios and, as such, includes features like rounded edges, torque and velocity monitoring, and collision detection. Other important characteristics of this manipulator, like joint limits, maximum velocities, and torques, can be consulted in Table 4.4.

Table 4.4 – Kuka LBR iiwa 14 R820 joint limits, maximum velocities, and torques.

Joint	Angular limit (deg)	Maximum velocity (deg/s)	Maximum torque (N m)
1	±170	85	320
2	±120	85	320
3	±170	100	176
4	±120	75	176
5	±170	130	110
6	±120	135	40
7	±175	135	40

Although the manipulator was already defined, other decisions were not made (e.g., the gripper and the type of command to be used in the arm). As such, before presenting the obtained results, these decisions are discussed. Starting with the gripper, there were two possibilities. The first was to use an already existing gripper in the laboratory where this dissertation was developed (Schunk PG 070 C). The second was to buy a new one. As the first option could save thousands of euros, some experiments were done. Unfortunately, they were not well succeeded because the gripper was several years old, and thus, not all the instructions manuals were available. Moreover, Schunk no longer supported that gripper model. As

such, a new one had to be selected. Both aspects of the laboratory and the specific project in which this dissertation is inserted were taken into consideration:

- **Be collaborative**: like the manipulator, the gripper must also be collaborative because the presence of people in their vicinity will be allowed.
- **Maximum stroke and load:** at least 60 mm and 2 kg, respectively, to allow for gripping the wide range of objects used in the laboratory (e.g., bottles and tubes).
- **Adjustable force and stroke:** like the previous requirement, these can also increase the number of objects that can be grabbed.
- **Simple communication interface:** to speed up the setup process.

Table 4.5 presents the characteristics (related to the previously presented requirements) of some of the most popular collaborative grippers currently on the market (compatible with the robot to be used).

Table 4.5 – Most significant characteristics of some collaborative grippers that could be used on the project in which this dissertation is inserted (information obtained with possible suppliers).

Manufacturer and model	Stroke control (mm)	Force control (N)	Maximum load (kg)	Communication interface
OnRobot RG2	Yes, [0–110]	Yes, [3–40]	2	Ethernet or digital IO
OnRobot 2FG7	Yes, [1–39] or [35–73]	Yes, [0–110]	7	Ethernet or digital IO
Schunk Co–act EGP–C 40–N–N	No, [16; 28]	Yes, [35; 70; 105; 135]	0.7	Digital IO
Robotiq 2F–140	Yes, [0–140]	Yes, [10–125]	2.5	Modbus RTU (RS–485)
Robotiq Hand–E	Yes, [0–50]	Yes, [20–185]	4.7	Modbus RTU (RS–485)

From Table 4.5 the grippers OnRobot RG2 and Robotiq 2F–140 stand out with adjustable wide strokes. Moreover, they comply with the force and load requirements. Although the Ethernet interface available in the OnRobot gripper could be considered a bit more simple, the decision between the two should be made mainly by the price. The OnRobot gripper is approximately 450 euros cheaper and thus, was the selected one.

The gripper OnRobot RG2 (Figure 4.26) when acquired for the robot Kuka LBR iiwa comes with a group of accessories:

- **Compute box:** the gripper's controller.
- **Adapter/quick changer:** to be mounted on the robot's flange, allowing the gripper to be attached.
- **Connection cables:** Ethernet and connection cable between the compute box and adapter.
- **Velcro tapes:** to strap the cable that goes from the compute box to the adapter/quick changer to the arm.

The installation of these components is a quick process in which only very general precautions have to be taken into account. For example: the tightening torque of the adapter's bolts must be 10 N m; the cable that connects the compute box and the adapter must be strapped to the arm with extra length, to ensure that it is not pulled when the robot moves. Also, the adapter has a cable holder that must be used to prevent any excessive strain on the cable's connector (Figure 4.26); the metallic enclosure of the arm's controller and of the compute box can not be connected (no galvanic connection between the two) (OnRobot, 2021).

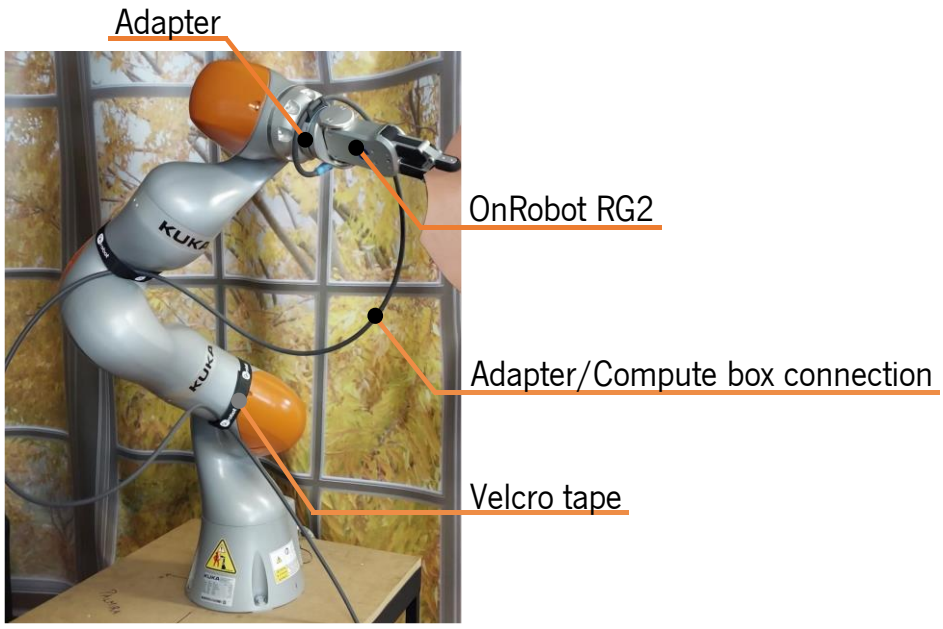


Figure 4.26 – OnRobot RG2 installation.

Two procedures must be followed when a gripper is attached to the arm for the first time. First, using the software Kuka Sunrise.Workbench (robot's development environment), one has to create an object template for the tool, adding properties like the mass, Center of Gravity (CG), and principal moments of inertia. Only the first two were inserted (1 kg and (0; 0; 114.3) mm (relative to the tip's frame–Figure

3.5), respectively[26]). OnRobot did not provide the principal moments of inertia. Also, it was impossible to obtain a 3D model of the gripper with mass properties, which would allow obtaining the desired values with a CAD software (e.g., SolidWorks). Moreover, the robot's controller has an option to determine this kind of property, however it is not reliable for grippers with such a reduced mass. This problem can be considered minor because, with the present mass and dimensions, the principal moments of inertia will be residual compared to the maximum permissible, which is 0.3 kg m^2, and the mass and CG are known. Concerning these latter, the payload diagram of the robot can be observed (Annex I) to ensure that the load capacity is respected. Additionally, a procedure associated with the relation between the mechanical robot axes' positions and the motors' angles needs to be done (position mastering). This process can be easily triggered using a preprogrammed option present in the robot's teach pendant (Kuka, 2019). A Kuka technician validated all these procedures.

The setup needed is constituted by the gripper, arm, and two more computers[27], and all communicate via Ethernet. A router is used to connect all the devices. It should be noted that the network settings were adapted to be in accordance with the robot's controller default IP: 172.31.1.147 (subnet mask: 255.255.0.0), and all the devices were defined with a static one, like 172.31.1.XXX (subnet mask: 255.255.255.0) (to ensure that they would not change every time the setup was used). These two procedures can be done using the router configuration menu (which can be accessed with its IP and an Internet browser). However, when dealing with the gripper, there is a minor detail to consider. There are four switches on its back, and following the instructions given by OnRobot, the combination between them should be the one present in Figure 4.27 (1, 2, and 3 down, and 4 up). Otherwise, it will not be possible to attribute the desired static IP to the gripper (OnRobot, 2021).

Figure 4.27 – Switches combination (in the gripper's compute box) that allows the attribution of a static IP.

[26] Only the data of the gripper and adapter was used because the leather pieces to be grabbed have a negligible mass.

[27] As it will be seen, the robot's command will be done using a computer running Ubuntu, but Kuka Sunrise.Workbench only runs on Windows. As such, two computers are needed.

Using the gripper's IP and an Internet browser is possible to access a GUI, which allows one to control different aspects, e.g., opening and closing the fingers (Figure 4.28), configure I/O signals, and observe the state of the sensors.

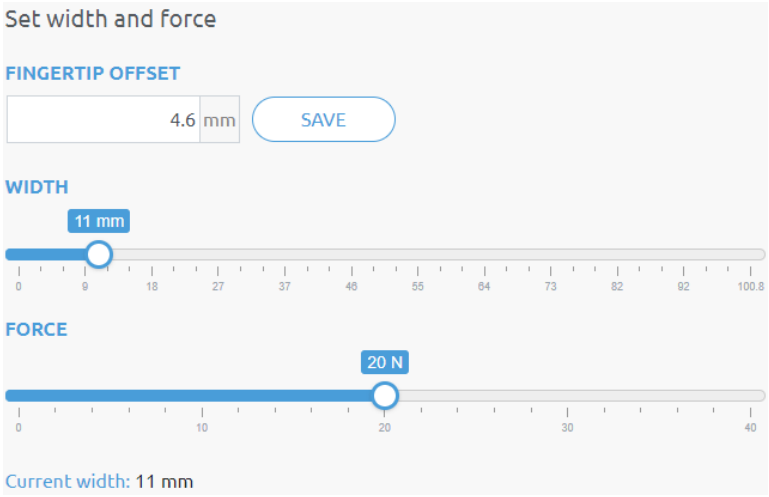


Figure 4.28 – GUI that allows to open and close the gripper (with adjustable width and force).

The robot's command could be done using different interfaces. One could use the arm's official development environment, Kuka Sunrise.Workbench, or other nonofficial interfaces such as the ones developed in the University of Sheffield (Mokaram et al., 2017; Mokaram et al., 2018), Coimbra (Safeea & Neto, 2019a; Safeea & Neto, 2019b), or Munich (Hennersperger et al., 2017; Virga et al., 2019).
To choose the adequate one for this dissertation application, one should recall some requirements:

- **Use ROS (Robot Operating System):** ROS is a framework with a set of software libraries and tools, organized in packages, that can speed up the development of robotic applications and the integration of different pieces of equipment (e.g., in future works, the integration of the robot and gripper's command will be easier). ROS can also be seen as a communication interface. To better understand its basics, some concepts should be briefly presented: *nodes* are executables that can be run simultaneously and communicate with each other. A node that pretends to share an information, *publisher*, sends it on a *message* to a *topic*. This latter will transmit it to all the nodes that subscribed to that topic, *subscribers*. Nodes can also communicate using services. Here, a *service client* sends a *request* to a *service server*, which answers with a *response*. The communication of the different elements in a ROS network is coordinated by a ROS *master* (Sepúlveda, 2018).
- **Allow online commands:** in this work, all the possible decisions must be taken considering that in future works online corrections will be introduced in the followed trajectories (when the desired deformations are not being achieved in the leather pieces). Thus, the chosen interface

should be capable of commanding the arm online. Particularly, the motion class SmartServo from the Kuka package Sunrise.Servoing should be used for two reasons: it is recommended for visual servoing applications; it will be easier to obtain smooth and safe movements because it has velocity, acceleration, and jerk limitations (Kuka, 2018).

With these restrictions, the interface IIWA STACK, from the University of Munich, was selected. The one from Sheffield uses ROS but without the package Sunrise.Servoing, and the Coimbra's one employs this latter, but not the SmartServo motion class, and it is not in ROS. With this decision, the instructions from Virga et al. (2019) were followed to install the package IIWA STACK on an Ubuntu 18 computer and make the necessary changes using Kuka Sunrise.Workbench.

As Matlab can be connected with ROS (using the ROS Toolbox), and it is the primary programming platform used in this dissertation, two procedures were executed so that it was possible to work with the IIWA STACK package from it, and to make this process less cumbersome. The first procedure consisted of using the ROS Custom Message add-on to create the package's ROS custom messages in Matlab. The second was to create a class (ros_interface.m) where some functions were implemented to make sending and receiving messages to/from topics and calling services more straightforward. Finally, a script that uses these functions, like the one presented below, was written to send the joint values to the arm online.

Strategy used to command the arm with the IIWA STACK package through the MathWorks ROS Toolbox

1: **rosinit** //connects Matlab to the ROS network
2: **initialize** ros_interface class
3: joints ← matrix with the joint values (each row respects a pose and each column a joint)
4: **set** joint velocity, acceleration, acceleration override
5: **for** (j = 1; j ≤ number of poses)
6: **send** joints(j, 1:7)
7: **wait** 0.2 // intentional delay for making the readings in the next line ready
8: **while** time to reach joint(j, 1:7) > 0.6 // time in seconds
9: **end**
10: **end**

Although, before using it, the command *roscore* must be used in the Ubuntu command line. Immediately after or before, an application on the robot's controller must also be initiated. There are two ways of doing this procedure: the application can be run with the arm's teach pendant, which is problematic because it requires the enabling switch to be continuously pressed (to authorize the robot's movement); the application can be initialized externally. The latter option only requires the specification, using the Kuka Sunrise.Workbench, of two points: the robot's application to be externally activated; the IP and port of the

computer used for external control. Moreover, one will need a program that uses UDP (User Datagram Protocol) to initialize the previously selected application (the Java one present in the instructions manual of the robot can be used) (Kuka, 2019). As this second option is more convenient, it was the chosen one. To verify if the desired deformations on the leather pieces could be reproduced during the trajectory tracking with the robot some elements had to be used additionally to the active arm and gripper–Figure 4.29.

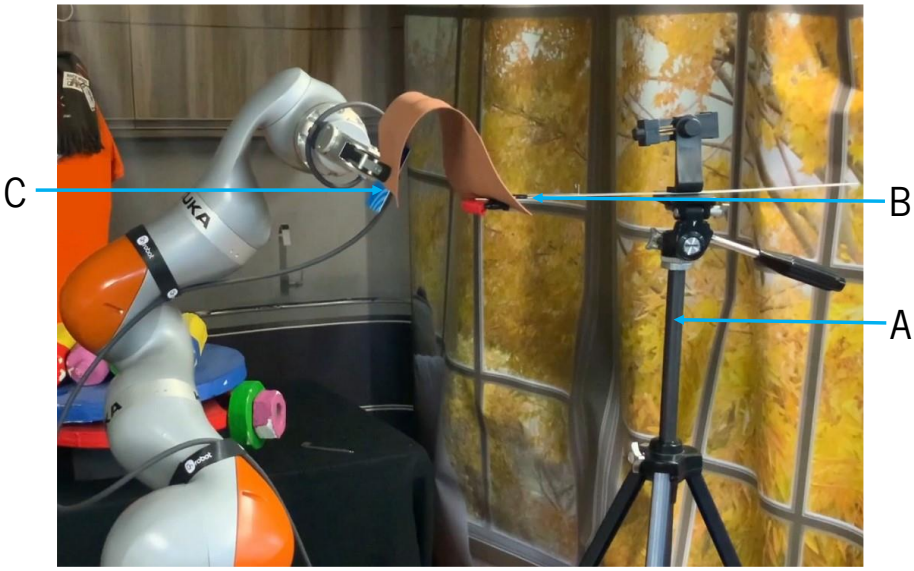


Figure 4.29 – Aditional elements to the gripper and active manipulator used to verify if the desired deformations could be reproduced.

The passive arm, for simplicity, is not yet a manipulator. A grasp support, which is compliant in two degrees of freedom[28] (rotation and translation in an axis perpendicular to the ground) (Figure 4.29–A), was used to secure the leather between two metal strips (Figure 4.29–B). It should be noted that the contact area of the gripper's fingertips with the deformable object is increased with a card (approximately 86×54 mm) to mimic the human fingers better (Figure 4.29–C). This setup allowed to verify that the desired deformations in the leather piece were reproduced when the joint values generated by the proposed trajectory tracking method were sent to the robot–Figure 4.30.

While the manipulator did the movements present in Figure 4.30, the joint values were read each time a new pose was sent. Then, with FK, the tip's pose at that moment was calculated and compared with the one of the desired trajectory, allowing to obtain the following data:

- The manipulator reached the acquired poses with a mean positioning error of 5.5 mm (standard deviation: 3.3 mm).

[28] In this reproduction, it was essential to have a compliant passive manipulator because it made the system tolerant to errors. These errors result, for example, from the difficulty that is to reproduce the initial relative position between the gripping supports, which can lead to excessive stretching.

- The mean angular deviations of the x, y, and z unit vectors that specify the orientations were, respectively: 2.8°, 2.7°, and 0.8° (standard deviations: 1.7°, 1.8°, and 0.6°, respectively).

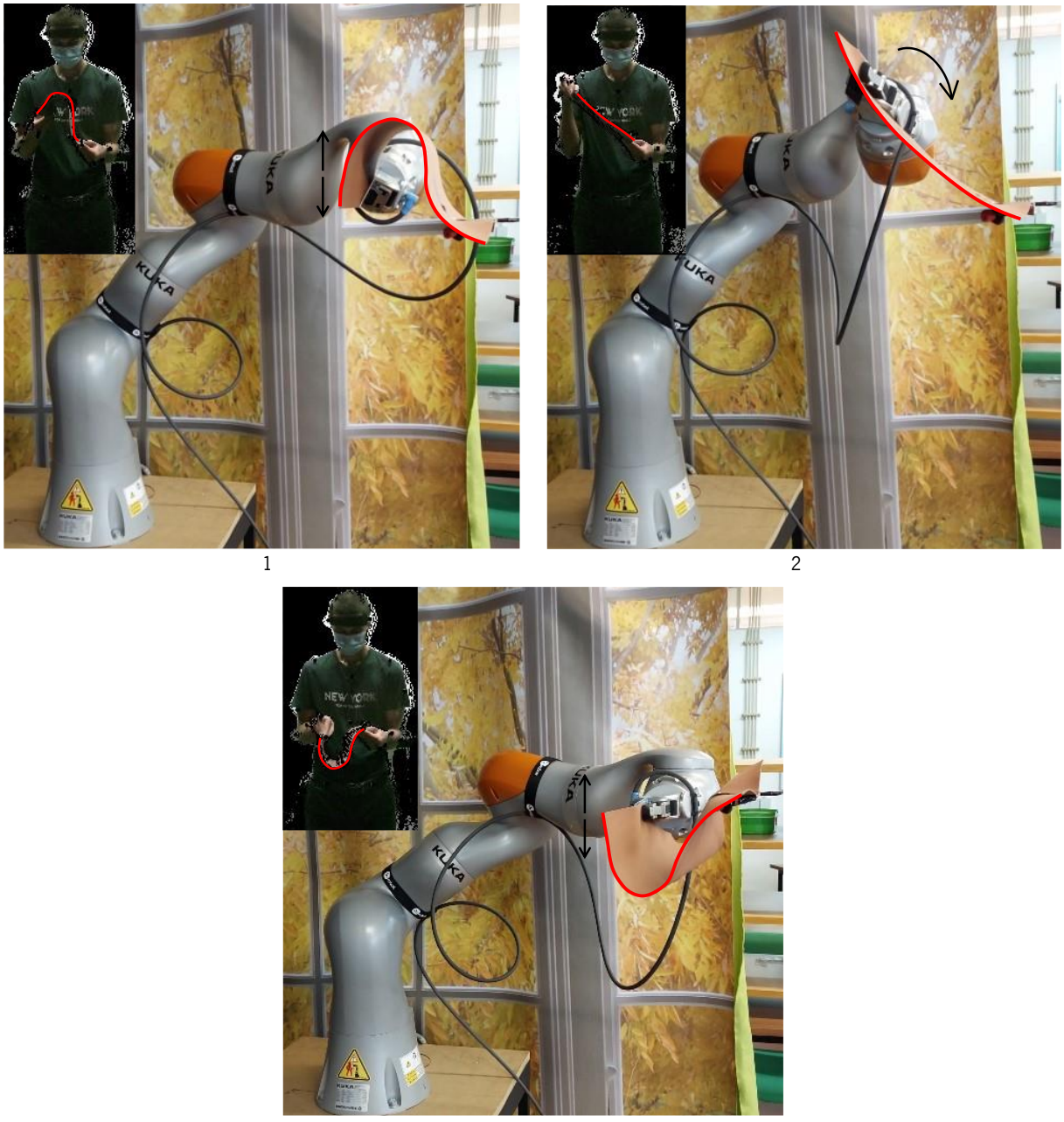


Figure 4.30 – Robotic system reproducing the demonstrated bending movements (original movement in the upper left corner): 1 – Leather piece bent convexly and movement of its concavity extremity along its length; 2 – Transition between the situation 1 and 3; 3 – Leather piece bent concavely and movement of its concavity extremity along its length.
Full video available in: https://www.youtube.com/watch?v=sbY8fS4se0U.

It should be noted that this accuracy was negatively affected to favor the safety and smoothness of the movements. Despite this, as shown in Figure 4.30, it is in accordance with the task's requirements. The torsion movement present in the other sections of this chapter could not be validated with the real setup due to a loss of synchronism between the orientations and positions in the trajectory acquisition.

This chapter (Chapter 4) exposed the most significant contribution of the present dissertation, i.e., a trajectory tracking method that considers manipulability. Furthermore, the results of different tests, both in simulation and with the actual arm, were presented. The most important conclusions of these experiments were: the desired trajectories were correctly tracked, and a safe distance to joint limits and singularities was achieved (i.e., a high manipulability). It was possible to induce the desired deformations in the leather pieces.

With these results obtained, it only remains to present a summary of the conclusions obtained throughout this dissertation and the future work suggestions. These will be the topics of the following and last chapter.

5. CONCLUSIONS

As presented in the introduction, this work contributed to automating a quality inspection process of leather pieces, using a learning from demonstration approach. The main goal was to solve a physical equivalence problem, i.e., to reproduce, using a 7-DoF serial manipulator, the trajectory described by the hand of an expert during the inspection process. During the resolution of this problem, the capability to change the tip's pose arbitrarily in every pose, i.e., the manipulability, was considered. This prepares the robot to be subjected to online corrections needed in future works.

Despite the importance of all the available works in the literature, it can be argued that Faria et al. (2018) is the most suitable one to be used in the present dissertation. It allows to generate maps, which are constituted by the feasible arm angle intervals (avoid joint limits and singularities) for each pose. Also, a method to find continuous paths that solve the generated maps is proposed, and the IK problem is solved. These procedures are possible for all the eight possible IK solutions. Moreover, the singularities detection strategy can have its robustness increased if the method of Wang et al. (2021) is applied.

The works of Faria et al. (2018) and Wang et al. (2021) would probably be sufficient to solve a trajectory tracking problem in a high number of situations. However, the robot's manipulability issues need greater caution. This is particularly important to allow the robot to perform online corrections. Hence, this dissertation contributes with a novel method to improve manipulability. First, the trajectory is sequentially relocated using a grid in the manipulator's workspace zone where the trajectory can be tracked. This scanning allows using joint 4 to influence manipulability. It also allows searching for the maps with the wider feasible intervals. The maps are generated and saved for each grid point and IK solution. Then, paths that solve this maps are defined guided by the interval's centers. The continuity of the maps in the boundary $-\pi/\pi$ was considered, contrary to what was seen until now in the literature, and allowing a much more accurate tracking of the intervals' middle points. In the end, a path that reliably keeps a high distance to the limits of the feasible intervals, and that is associated to a joint 4 variation which also mantains consistently a safe distance to its limit and singularity, is chosen using a manipulability based score. With the paths, the position of the trajectory and the IK solution selected, the joint values are calculated using IK.

Using two real inspection trajectories, the proposed method was able to keep a minimum distance to joint limits and singularities of at least 18.9°, and a minimum average distance of at least 42.6°. This data helps to strengthen the thesis that the proposed trajectory tracking method increases manipulability,

and prepares the robot to be subjected to online corrections. Also, the IK and redundancy resolution approaches used can cope with online requirements, as well as the setup to command the manipulator. The generated joint values were used to confirm the tracking of the trajectories both in simulation and real life. In both, the movements were safe and smooth. Particularly in the experiments with the real robot, it was possible to conclude that the desired deformations in the leather pieces can be achieved using the present LfD approach.

As the arm angle is an abstraction for six joints it can fail to represent the real proximity to the boundaries of all joints or singularities. Despite this, as the proposed method searches for maps with many consecutive poses with wide intervals, the probability of some limit or singularity being close, far from the extremes of those intervals is low, or it would eventually break them. Thus, the present method tends to be immune to this problem, as the obtained results show. If this approach fails in the future, the presented challenge can be formulated as an optimization problem using each of the individual joints, but this is a more complex procedure. For example, the number of variables increases from two (arm angle and joint 4) to seven (each of the individual joints). Thus, it should only be used if a clear need exists.

The contributions of the present dissertation gave rise to the paper "Trajectory tracking for the inspection of deformable objects considering manipulability of a 7-DoF serial manipulator", presented at the 2022 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC) (Borges et al., 2022).

5.1 Future work

This dissertation is expected to be a starting point for an easy programable robotic system that can induce the necessary deformations in leather pieces to inspect them. Although the contributions of this work can be considered significant, there are still a group of tasks that one will have to pay close attention to in future works:

- At the moment, the system is working in an open loop, i.e., it reproduces the acquired trajectories and it is expected that the desired deformations are achieved in the leather pieces. Although this approach works well when the exemplification conditions are carefully reproduced, some minor differences (e.g., gripping positions of the leather pieces) can lead to excessive stretching and different deformations. An online correction system should be implemented to increase the overall robustness of the used approach.

- Integrate the command of the gripper currently installed in the active manipulator (OnRobot RG2) into the command of the latter, using a ROS interface like Kiyokawa (2022).
- Implement a second arm in the testing setup to work as a passive support, and integrate its command with all the other components.
- The chosen gripper has the advantage of having the fingertips easily changed. New ones with a higher contact area should be designed, produced, and mounted. This will allow to better emulate a human hand and increase the proposed approach's success rate.
- Perform more tests with different types of movements.
- Implement force control to allow the execution of inspection movements that stretch the leather pieces.
- Study the possibility of using the methods presented in Section 2.3 to generalize the exemplified trajectories for leather pieces with different characteristics from those used in the exemplifications.

BIBLIOGRAPHY

Aggarwal, C. C. (2018). *Neural networks and deep learning: A textbook* (1ª ed.). Springer International Publishing. https://doi.org/10.1007/978-3-319-94463-0

Argall, B. D., Chernova, S., Veloso, M., & Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, *57*(5), 469–483. https://doi.org/10.1016/j.robot.2008.10.024

Asfour, T., & Dillmann, R. (2003). Human-like motion of a humanoid robot arm based on a closed-form solution of the inverse kinematics problem. *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, *2*, 1407–1412. https://doi.org/10.1109/IROS.2003.1248841

Billard, A. G., Calinon, S., & Dillmann, R. (2016). Learning from Humans. In B. Siciliano & O. Khatib (Eds.), *Springer Handbook of Robotics* (pp. 1995–2014). Springer International Publishing. https://doi.org/10.1007/978-3-319-32552-1_74

Borges, C., Ribeiro, J., Louro, L., Vicente, P., Faria, C., Monteiro, S., & Bicho, E. (2022). Trajectory tracking for the inspection of deformable objects considering manipulability of a 7-DoF serial manipulator. *2022 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, 198–204. https://doi.org/10.1109/ICARSC55462.2022.9784811

Chernova, S., & Thomaz, L. (2014). Robot learning from human teachers. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *8*(3), 1–121. https://doi.org/10.2200/S00568ED1V01Y201402AIM028

Chui, H., & Rangarajan, A. (2003). A new point matching algorithm for non-rigid registration. *Non-rigid Image Registration*, *89*(2), 114–141. https://doi.org/10.1016/S1077-3142(03)00009-2

Corke, P. (2013). *Robotics, vision and control: Fundamental algorithms in Matlab* (1st ed.). Springer

Publishing Company, Incorporated. https://doi.org/10.1007/978-3-319-54413-7

Craig, J. (2014). *Introduction to robotics: Mechanics and control* (3rd ed.). Pearson.

Crum, W. R., Hartkens, T., & Hill, D. L. G. (2004). Non-rigid image registration: Theory and practice. *The British Journal of Radiology*, *77*, S140–S153. https://doi.org/10.1259/bjr/25329214

Dahm, P., & Joublin, F. (1997). *Closed form solution for the inverse kinematics of a redundant robot arm*.

Dou, R., Yu, S., Li, W., Chen, P., Xia, P., Zhai, F., Yokoi, H., & Jiang, Y. (2022). Inverse kinematics for a 7-DOF humanoid robotic arm with joint limit and end pose coupling. *Mechanism and Machine Theory*, *169*, 104637. https://doi.org/10.1016/j.mechmachtheory.2021.104637

Edmonds, M., Gao, F., Xie, X., Liu, H., Qi, S., Zhu, Y., Rothrock, B., & Zhu, S. (2017). Feeling the force: Integrating force and pose for fluent discovery through imitation learning to open medicine bottles. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3530–3537. https://doi.org/10.1109/IROS.2017.8206196

Evrard, P., Gribovskaya, E., Calinon, S., Billard, A., & Kheddar, A. (2009). Teaching physical collaborative tasks: Object-lifting case study with a humanoid. *2009 9th IEEE-RAS International Conference on Humanoid Robots*, 399–404. https://doi.org/10.1109/ICHR.2009.5379513

Faria, C. (2020). *Development of a robotic system to assist neurosurgeons in minimally invasive stereotactic procedures* [Unpublished doctoral dissertation]. Universidade do Minho.

Faria, C. (2021). *Kuka_iiwa_ik* [Computer software]. GitHub. https://github.com/neuebot/kuka_iiwa_ik

Faria, C., Ferreira, F., Erlhagen, W., Monteiro, S., & Bicho, E. (2018). Position-based kinematics for 7-DoF serial manipulators with global configuration control, joint limit and singularity avoidance. *Mechanism and Machine Theory*, *121*, 317–334. https://doi.org/10.1016/j.mechmachtheory.2017.10.025

Garcia, N., Rosell, J., & Suarez, R. (2019). Motion planning by demonstration with human-likeness evaluation for dual-arm robots. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *49*(11), 2298–2307. https://doi.org/10.1109/TSMC.2017.2756856

Gong, M., Li, X., & Zhang, L. (2019). Analytical inverse kinematics and self-motion application for 7-DoF redundant manipulator. *IEEE Access*, *7*, 18662–18674. https://doi.org/10.1109/ACCESS.2019.2895741

Gutzeit, L., Fabisch, A., Otto, M., Metzen, J. H., Hansen, J., Kirchner, F., & Kirchner, E. A. (2018). The BesMan learning platform for automated robot skill learning. *Frontiers in Robotics and AI*, *5*. https://doi.org/10.3389/frobt.2018.00043

Hennersperger, C., Fuerst, B., Virga, S., Zettinig, O., Frisch, B., Neff, T., & Navab, N. (2017). Towards MRI-based autonomous robotic US acquisitions: A first feasibility study. *IEEE Transactions on Medical Imaging*, *36*(2), 538–548. https://doi.org/10.1109/TMI.2016.2620723

Huang, S. H., Pan, J., Mulcaire, G., & Abbeel, P. (2015). Leveraging appearance priors in non-rigid registration, with application to manipulation of deformable objects. *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 878–885. https://doi.org/10.1109/IROS.2015.7353475

Kim, H., & Rosen, J. (2015). Predicting redundancy of a 7 DoF upper limb exoskeleton toward improved transparency between human and robot. *Journal of Intelligent & Robotic Systems*, *80*(1), 99–119. https://doi.org/10.1007/s10846-015-0212-4

Kiyokawa, T. (2022). Onrobot ROS drivers for OnRobot Grippers. [Computer software]. GitHub. https://github.com/Osaka-University-Harada-Laboratory/onrobot

Kreutz-Delgado, K., Long, M., & Seraji, H. (1992). Kinematic analysis of 7-DoF manipulators. *The International Journal of Robotics Research*, *11*(5), 469–481. https://doi.org/10.1177/027836499201100504

Kuhlemann, I., Schweikard, A., Jauer, P., & Ernst, F. (2016). Robust inverse kinematics by configuration control for redundant manipulators with seven DoF. *2016 2nd International Conference on Control, Automation and Robotics (ICCAR)*, 49–55. https://doi.org/10.1109/ICCAR.2016.7486697

Kuka. (2018). *Kuka Sunrise.Servoing 1.16*.

Kuka. (2019). *Kuka Sunrise.OS 1.16 and Kuka Sunrise.Workbench 1.16*.

Kuka. (2020). *Option media flange*.

Lee, S., & Bejczy, A. (1991). Redundant arm kinematic control based on parameterization. *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, *1*, 458–465. https://doi.org/10.1109/ROBOT.1991.131621

McConachie, D., Dobson, A., Ruan, M., & Berenson, D. (2020). Manipulating deformable objects by interleaving prediction, planning, and control. *International Journal of Robotics Research*, *39*(8), 957–982. https://doi.org/10.1177/0278364920918299

McKerrow, P. (1991). *Introduction to robotics*. Addison-Wesley Publishing Company.

Mokaram, S., Aitken, J. M., & Law, J. (2018). ROS-integrated API for the Kuka LBR iiwa collaborative robot. [Computer software]. GitHub. https://github.com/jonaitken/KUKA-IIWA-API.git

Mokaram, S., Aitken, J. M., Martinez-Hernandez, U., Eimontaite, I., Cameron, D., Rolph, J., Gwilt, I., McAree, O., & Law, J. (2017). A ROS-integrated API for the Kuka LBR iiwa collaborative robot. 20th IFAC World Congress, 50(1), 15859–15864. https://doi.org/10.1016/j.ifacol.2017.08.2331

Moradi, H., & Lee, S. (2005). Joint limit analysis and elbow movement minimization for redundant manipulators using closed form method. In D. Huang, X. Zhang, & G. Huang (Eds.), *Advances in Intelligent Computing* (pp. 423–432). Springer Berlin Heidelberg. https://doi.org/10.1007/11538356_44

Myronenko, A., & Song, X. (2010). Point set registration: Coherent Point Drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(12), 2262–2275. https://doi.org/10.1109/TPAMI.2010.46

Nenchev, D., Tsumaki, Y., & Takahashi, M. (2004). Singularity-consistent kinematic redundancy resolution for the S-R-S manipulator. *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, *4*, 3607–3612. https://doi.org/10.1109/IROS.2004.1389975

Nielsen, M. A. (2015). *Neural Networks and Deep Learning*. Determination Press. https://books.google.pt/books?id=STDBswEACAAJ

Oh, J., Bae, H., & Oh, J. (2017). Analytic inverse kinematics considering the joint constraints and self-collision for redundant 7DoF manipulator. *2017 First IEEE International Conference on Robotic Computing (IRC)*, 123–128. https://doi.org/10.1109/IRC.2017.46

OnRobot. (2021). *User manual for Kuka robots*.

Ravichandar, H., Polydoros, A. S., Chernova, S., & Billard, A. (2020). Recent advances in robot learning from demonstration. *Annual Review of Control, Robotics, and Autonomous Systems*, *3*(1), 297–330. https://doi.org/10.1146/annurev-control-100819-063206

Ribeiro, J. (2022). *Development of an active vision system for robot inspection of complex objects* [Master's dissertation, Universidade do Minho]. RepositóriUM. https://hdl.handle.net/1822/77362

Rivas, P. (2020). *Deep Learning for Beginners*. Packt Publishing Ltd. https://www.packtpub.com/product/deep-learning-for-beginners/9781838640859

Safeea, M., & Neto, P. (2019a). Kuka Sunrise Toolbox: Interfacing collaborative robots with Matlab. *IEEE Robotics & Automation Magazine*, *26*(1), 91–96. https://doi.org/10.1109/MRA.2018.2877776

Safeea, M., & Neto, P. (2019b). IIWA control toolbox. [Computer software]. GitHub. https://github.com/Modi1987/IIWAContorlToolbox.git

Saha, M., & Isto, P. (2007). Manipulation planning for deformable linear objects. *IEEE Transactions on Robotics*, *23*(6), 1141–1150. https://doi.org/10.1109/TRO.2007.907486

Sanchez, J., Corrales, J. A., Bouzgarrou, B. C., & Mezouar, Y. (2018). Robotic manipulation and sensing of deformable objects in domestic and industrial applications: A survey. *International Journal of Robotics Research*, *37*(7), 688–716. https://doi.org/10.1177/0278364918779698

Sanchez, J., Mohy El Dine, K., Corrales, J. A., Bouzgarrou, B., & Mezouar, Y. (2020). Blind manipulation of deformable objects based on force sensing and finite element modeling. *Frontiers in Robotics and AI*, *7*. https://doi.org/10.3389/frobt.2020.00073

Sariyildiz, E., Ucak, K., Oke, G., Temeltas, H., & Ohnishi, K. (2012). Support Vector Regression based inverse kinematic modeling for a 7-DOF redundant robot arm. *2012 International Symposium on Innovations in Intelligent Systems and Applications*, 1–5. https://doi.org/10.1109/INISTA.2012.6247033

Schulman, J., Ho, J., Lee, C., & Abbeel, P. (2016). Learning from Demonstrations Through the Use of Non-rigid Registration. In M. Inaba & P. Corke (Eds.), *Robotics Research: The 16th International Symposium ISRR* (pp. 339–354). Springer International Publishing. https://doi.org/10.1007/978-3-319-28872-7_20

Sepúlveda, J. (2018). *Desenvolvimento de um assistente robótico para neurocirurgias estereotáxicas: Solução baseada no robô Sawyer* [Master's dissertation, Universidade do Minho]. RepositóriUM. https://hdl.handle.net/1822/59471

Shimizu, M., Kakuya, H., Yoon, W., Kitagaki, K., & Kosuge, K. (2008). Analytical inverse kinematic computation for 7-DoF redundant manipulators with joint limits and its application to redundancy

resolution. *IEEE Transactions on Robotics*, *24*(5), 1131–1142. https://doi.org/10.1109/TRO.2008.2003266

Siciliano, B., Sciavicco, L., Villani, L., & Oriolo, G. (2009). Kinematics. In M. Grimble & M. Johnson (Eds.), *Robotics: Modelling, Planning and Control* (pp. 39–103). Springer London. https://doi.org/10.1007/978-1-84628-642-1_2

Silva, E. (2011). *Reaching, grasping and manipulation in anthropomorphic robot systems* [Doctoral dissertation, Universidade do Minho]. RepositóriUM. http://hdl.handle.net/1822/18243

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (Second edition). The MIT Press.

Tang, T. (2020, March 15th). A framework for manipulating deformable linear objects by coherent point drift [Video]. YouTube. https://www.youtube.com/watch?v=21OJfrLfGQY

Tang, T., Liu, C., Chen, W., & Tomizuka, M. (2016). Robotic manipulation of deformable objects by tangent space mapping and non-rigid registration. *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2689–2696. https://doi.org/10.1109/IROS.2016.7759418

Tang, T., Wang, C., & Tomizuka, M. (2018). A framework for manipulating deformable linear objects by coherent point drift. *IEEE Robotics and Automation Letters*, *3*(4), 3426–3433. https://doi.org/10.1109/LRA.2018.2852770

The MathWorks, Inc. (n.d.). atan2: Four-quadrant inverse tangent. https://www.mathworks.com/help/matlab/ref/atan2.html#bucseo2_vh

Tian, X., Xu, Q., & Zhan, Q. (2021). An analytical inverse kinematics solution with joint limits avoidance of 7-DoF anthropomorphic manipulators without offset. *Journal of the Franklin Institute*, *358*(2), 1252–1272. https://doi.org/10.1016/j.jfranklin.2020.11.020

Tondu, B. (2006). A closed-form inverse kinematic modelling of a 7R anthropomorphic upper limb based on a joint parametrization. *2006 6th IEEE-RAS International Conference on Humanoid Robots*, 390–397. https://doi.org/10.1109/ICHR.2006.321302

Vahrenkamp, N., Asfour, T., Metta, G., Sandini, G., & Dillmann, R. (2012). Manipulability analysis. *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, 568–573. https://doi.org/10.1109/HUMANOIDS.2012.6651576

Virga, S., Esposito, M., Peters, A., Stoyanov, T. (2019). IIWA STACK. [Computer software]. GitHub. https://github.com/IFL-CAMP/iiwa_stack

Vogt, D., Stepputtis, S., Grehl, S., Jung, B., & Ben Amor, H. (2017). A system for learning continuous human-robot interactions from human-human demonstrations. *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2882–2889. https://doi.org/10.1109/ICRA.2017.7989334

Vogt, D., Stepputtis, S., Grehl, S., Jung, B., & Ben Amor, H. (2017). A system for learning continuous human-robot interactions from human-human demonstrations. *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2882–2889. https://doi.org/10.1109/ICRA.2017.7989334

Vu, H. (2012). *Control of an anthropomorphic manipulator involved in physical human-robot interaction* [Master's dissertation, Universidade do Minho]. RepositóriUM. http://hdl.handle.net/1822/19955

Waldron, J., & Schmiedeler, J. (2016). Kinematics. In B. Siciliano & O. Khatib (Eds.), *Springer Handbook of Robotics* (pp. 11–36). Springer International Publishing. https://doi.org/10.1007/978-3-319-32552-1_2

Wang, J., Lu, C., Zhang, Y., Sun, Z., & Shen, Y. (2021). A numerically stable algorithm for analytic inverse kinematics of 7-DoF S-R-S manipulators with joint limit avoidance. *Journal of Mechanisms and Robotics*, 1–15. https://doi.org/10.1115/1.4053375

Wiedmeyer, W. (2020). *Kinematics*. [Computer software]. GitHub. https://github.com/KITrobotics/rll_sdk/tree/master/kinematics

Wiedmeyer, W. (2020). *Kinematics*. [Computer software]. GitHub. https://github.com/KITrobotics/rll_sdk/tree/master/kinematics

Wiedmeyer, W., Altoé, P., Auberle, J., Ledermann, C., & Kröger, T. (2021). A real-time-capable closed-form multi-objective redundancy resolution scheme for seven-DoF serial manipulators. *IEEE Robotics and Automation Letters*, *6*(2), 431–438. https://doi.org/10.1109/LRA.2020.3045646

Xie, Z., Zhang, Q., Jiang, Z., & Liu, H. (2020). Robot learning from demonstration for path planning: A review. *Science China Technological Sciences*, *63*(8), 1325–1334. https://doi.org/10.1007/s11431-020-1648-4

Xu, W., Yan, L., Mu, Z., & Wang, Z. (2016). Dual arm-angle parameterisation and its applications for analytical inverse kinematics of redundant manipulators. *Robotica*, *34*(12), 2669–2688. Cambridge Core. https://doi.org/10.1017/S0263574715000284

Yoshikawa, T. (1985). Manipulability and redundancy control of robotic mechanisms. *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, *2*, 1004–1009. https://doi.org/10.1109/ROBOT.1985.1087283

Zaplana, I., & Basanez, L. (2018). A novel closed-form solution for the inverse kinematics of redundant manipulators through workspace analysis. *Mechanism and Machine Theory*, *121*, 829–843. https://doi.org/10.1016/j.mechmachtheory.2017.12.005

Zhang, H. (2019, May 16th). hand gesture control +trajectory RPD + 3d curve programming [Video]. YouTube. https://www.youtube.com/watch?v=I-mkXT_k8kA

Zhang, H., Liu, S., Lei, Q., He, Y., Yang, Y., & Bai, Y. (2020). Robot programming by demonstration: A novel system for robot trajectory programming based on robot operating system. *Advances in Manufacturing*, *8*(2), 216–229. https://doi.org/10.1007/s40436-020-00303-4

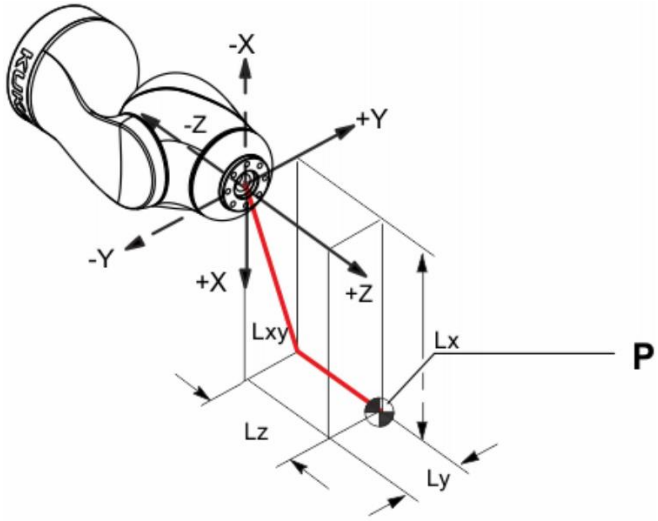
Zhang, T., McCarthy, Z., Jow, O., Lee, D., Chen, X., Goldberg, K., & Abbeel, P. (2018). Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 5628–5635. https://doi.org/10.1109/ICRA.2018.8461249

Zhu, Z., & Hu, H. (2018). Robot learning from demonstration in robotic assembly: A Survey. *Robotics*, *7*(2). https://doi.org/10.3390/robotics7020017

Annex I. PAYLOAD DIAGRAM FOR THE KUKA LBR IIWA 14 R820

This annex was adapted from Kuka (2020).

Load center of gravity



Permissible mass inertia at the design point (Lx, Ly, Lz): 0.3 kg m².

Payload diagram