TÉCNICO LISBOA

Inverse Kinematics of mobile semi-humanoid robot for movement planning

João Maria da Silveira Machado Chumbinho

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisors: Prof. Plinio Moreno López
Prof. José Alberto Rosado dos Santos Victor

Examination Committee

Chairperson: Prof. João Manuel de Freitas Xavier
Supervisor: Prof. Plinio Moreno López
Member of the Committee: Prof. José António Da Cruz Pinto Gaspar

June 2022

Declaration

I declare that this document is an original work of my own authorship and that it fulfils all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa

Acknowledgments

First of all, I would like to thank my parents for giving me the support and resources to have the maximum focus on this challenge.

Second, I would like to thank my sisters, Luisinha, Inês e Matilde, and grandma, VANDA, for helping me during breaks to recharge my energies.

At the same time, but also extremely important, It would be impossible not to mention the person that has always been by my side, my motivation, thanks for everything, Ju.

Besides all those people, I have to acknowledge my dissertation supervisor Prof. Plinio Moreno López, for his insight, support and help during those hard times of creating and developing the thesis.

Last but not least, to all my family, friends and colleagues from IST, TFC, GBR, Alfragide, DPV, and so many others that helped me through the difficult moments but especially in the good ones, which made this journey so special. Thank you.

To each and every one of you – Thank you.

Abstract

The main goal of this Master Thesis is to find the most effective way that the Robot Vizzy can reach any target pose of one of the end-effectors, selecting the arm to reach the position and moving the base accordingly. For that, the study, evaluation, and development of inverse kinematics algorithms are crucial to planning Vizzy's movements, always considering the two main components: (i) the segway platform and (ii) both Vizzy's arms. The most significant advantage of this approach is that those two components can be processed separately.

In the first part of the project, the analysis and developments focus on computing the forward and inverse kinematics for Vizzy's arms by considering an analytical approach and constantly comparing it with a control-based approach. This solution is not unique but will allow the understanding of which can be the redundancies. In this way, it is possible to recognize the space available for all of Vizzy's angles and movements.

After that, it is necessary to find the final position for the segway platform, which implies the study of all Vizzy's restrictions and limitations. This final pose for the segway platform is going to be acquired through inverse kinematics based on a hybrid approach using both analytical and Data-driven methods. Thus, gathering all this information, the final solution will provide the angles and positions that allow the placement of the robot and the respective achievement of the end-effector.

Keywords

Target, Redundancies, Segway Platform, Inverse Kinematics, End-Effector, Vizzy.

Resumo

O principal objetivo desta Tese de Mestrado é permitir que o Vizzy alcance a pose alvo a que foi proposta, através da seleção do braço para o qual é possivel alcançar a posição e a orientação pretendida. Assim sendo, o estudo, a avaliação e o desenvolvimento de cinemáticas inversas são essenciais para o planeamento do movimento do Robot, sempre tendo em consideração os seus dois componentes principais: (i) a base da plataforma e (ii) ambos os braços do Vizzy. A maior vantagem desta abordagem é a possível resolução do problema, abordando ambos os componentes de forma separada.

Assim sendo, o objetivo incidia no desenvolvimento da cinemática direta e inversa para os braços do Vizzy, aplicando uma abordagem analítica e sempre avaliando os resultados comparando-a com uma abordagem de controlo. Esta solução nunca é única, mas permite a perceção de quais são as redundâncias que, desta forma, fazem possível o reconhecimento do espaço disponivel para todos os ângulos e movimentos do Vizzy.

Posto isto, é necessário encontrar a posição final para a base do robot, o que implica o estudo de todas as restrições e limitações do Vizzy. A posição final para a base do robot é adquirida através da cinemática inversa baseada numa abordagem hibrida, usando ambas as abordagens analiticas e orientada por dados. Deste modo, através de toda esta informação, a solução final iria providenciar os ângulos e as posições que permitiriam a fixação da base e respetivo alcance do objetivo final.

Palavras Chave

Alvo, Redundâncias, Plataforma da Base, Cinemática Inversa, Cinemática Direta, Vizzy.

Contents

List of Figures

List of Tables

List of Algorithms

Listings

Acronyms

IK	Inverse Kinematics
SQP	Sequential Quadratic Programming
DH	Denavit-Hartenberg
DOF	Degrees of Freedom

1

Introduction

Contents

1.1 Motivation

The past 200 years have been considered the most revolutionary in technology until now. Before that, it would be unthinkable to believe that someone in Africa could see a person that is actually in Lisbon. The world has changed, and knowledge is easier to acquire for those who look for it. Not only talking about cellphones, television, computers but also when it comes to robots and artificial intelligence, i.e., any device that executes tasks autonomously.

Nowadays, robots/machines already have an essential role in our society. In the past, there were needed traffic police in the streets, who today are just mere traffic lights that everyone knows what they represent and the consequences of not respecting it. The same happened with the ATM. In the 50's, it would be laughable the idea of giving money to a robot. In 2022, even our cellphones show us where we can find the closer ATM to deposit and withdraw money. The world will keep developing, and the ones who adapt better will be the ones who will have easier lives.

Continuing the line of reasoning, if humans use robots and their technology to their benefit, the world will be fairer and fairer since people will have the same opportunities, despite their deficiencies, diseases, or even every peculiarity. Robots/machines have the potential to make a handicapped person walk, to make her have the opportunity to grab things or even hear. Moreover, there is nothing in the world that can pay that.

At the same time, this technology evolution will have a massive impact on society, industries, and even the education system. Robots and machines may potentially replace all jobs that represent complicated/dangerous situations for humans and let the human job be focused on knowledge and intelligence, things that, for now, are not still possible for the robots to replace.

With the evolution of technology, there is always the possibility of creating a world where robots can think and act independently, not directly depending on humans. There is where robotics and artificial intelligence connect, providing the conditions for the robot to learn by itself and improve its mechanisms of thinking and moving to allow a better living for human beings.

1.2 Problem Statement

Regarding the robot's movements, a very particular problem appears since the robot must learn how to control all the actuators for planning and executing motions in a safe manner in order to efficiently complete the mission that is proposed to do. A conventional model that serves to find the configuration of the actuators relies on rigid transformations that map the actuator values to poses in the world. When the number of actuators of the robot is large, the mapping between poses and actuators may have multiple solutions and discontinuities. In this thesis, we address the mapping between poses and actuators, focusing on robotic arms with a large number of actuators. In this particular case, the general

algorithm will be computed for a mobile robot, always considering the possibility that it may be able to have more than one arm.

In order to level up in this chapter, two different milestones need to be climbed. The first one, in the present, is to address the inverse kinematics problem for a particular robot, and the second one is that, in the future, if possible, the solution works for every single robot, where they can move and attain the objective purpose. Therefore, the only way, for now, to approach this problem is to start from the particular case and then expand to the general.

1.3 Objectives

The purpose of this master thesis is to plan the most efficient way to make a robot achieve its final goal, the target. This planning is done through Inverse Kinematics. Therefore, given a target pose (3D position + 3D orientation) in the world, the Inverse Kinematics problem finds the values of the actuators of the robot that reach the target pose. In this case, the actuators have a one-to-one mapping with the robot's degrees of freedom. In the case of mobile robotic arms, the actuators (i.e., degrees of freedom) include the arm joints and the mobile base motors. The main objective of this thesis is to solve the inverse kinematics problem of a humanoid-like mobile robot, *Vizzy*. We follow the divide and conquer approach, considering two components: (i) *Vizzy*'s both arms and (ii) its segway platform base. One of the most significant advantages of this process is the possibility of processing both components separately without interfering with each other.

On the one hand, the approach used on *Vizzy*'s both arms consisted of starting at the final goal (target) and, through Inverse Kinematics, understanding the possible positions and orientations for all the joints that constitute the arm while considering its limitations. On the other, after understanding the possible positions for the arm, this approach consists of discretizing the two-dimensional space of the motion of the base, always considering not only the feasible positions where the robot could be placed but also the positions and orientations found for arms.

1.4 Contributions

The strategies used for this problem, considering *Vizzy*'s characteristics, have the potential to work in every robot with 7 degrees of freedom in its arm, as happens with *Vizzy*. Previous approaches for inverse kinematics usually ignore the orientation to be able to find a solution, working in 3D. Our main contribution is a more effective algorithm for 6D inverse kinematics. The advantage of this approach is that it allows to reach not only a determined target but also a pre-determined orientation.

This strategy also has a considerable advantage since it not only works for the arm with a static

base platform but also develops a heuristic that, if the inverse kinematics does not work, can still find a solution using pre-calculated angles that allow achieving the closest position as possible.

Regarding obstacles, once the robot uses tabulated values to look for the ideal kinematics, it is possible to hide part of that values so that the robot does not consider them.

1.5 Document Outline

The present document is structured as follows:

- In Chapter 2, different possible approaches to inverse kinematics' problems are contained.
- In Chapter 3, the theory and methods required for achieving the goals of this thesis are presented. Some aspects of the implementation of the methods are also mentioned, such as using software and information regarding the robot simulator.
- In Chapter 4, the results are presented and analysed.
- In Chapter 5, the conclusions and final remarks are presented on suggestions for future work.

2

Theoretical Background and Related Work

Contents

Concerning finding the best solution for the problem that a specific robot needs to achieve a concrete goal, inverse kinematics is the answer. Inverse Kinematics is a method that, given a specific goal position, allows to compute the set of joint angles of a robot so that it may achieve the end effector purposed to [1,2]. This problem may be highly complex since it requires the study of the robot's anatomy as its degrees of freedom, which will have an impact on the final solution [3]. There is much-related work focused on achieving the final goal considering both base and arms together, such as [4, 5]. However, the main limitation is that the solutions are not able to deal with position and orientation at the same time, or are limited to a low number of Degrees of Freedom (DOF).

2.1 Inverse Kinematics for the Robot's Arms

To solve this problem, many strategies of inverse kinematics were developed, all based on two strategies: analytical methods or control-based methods [6].

Analytic methods use rigid body transformations to build parameterized equations, where the variables usually correspond to the robot's degrees of freedom. For this to work properly, for a robot of N degrees of freedom, N unique equations must be found. However, due to 3D constraints (which yield a maximum of 6 constraints) and redundancies (i.e. multiple options for the same target), analytic solutions are limited to 6 DOF.

For control-based methods, it is essential to understand what is the main goal of the inverse kinematics. Given a target pose $x = (T_P, T_O)$, defined by its position and orientation, respectively, the goal is to find the joint angles, which represent the degrees of freedom of the robot, $\theta_1, ..., \theta_n$, that allow to define the end-effector as a funcion of $\theta_1, ..., \theta_n$. This function is defined as the forward kinematics and this relation between the target pose and the parameters can be represented by the following equation:

$$x = f(\theta_1, ..., \theta_n) \tag{2.1}$$

Therefore, the solution that is supposed to be found is the solution of the inverse kinematics that can be defined as follows:

$$(\theta_1, ..., \theta_n) = f^{-1}(x) \tag{2.2}$$

Unfortunately, there is not always an optimal solution and, even worst, may not even exist a feasible solution. Thus, this is where the control-based methods take action. They use iterative methods with the support of the Jacobian Matrices of the robot, which are the matrices of all first-order partial derivatives, defined as

$$\left(\frac{\partial x_j}{\partial \theta_i}\right)_{i,j} = J(\theta) \iff \frac{\partial f(\theta)}{\partial \theta} = J(\theta) \tag{2.3}$$

Which can also be represented as follows:

$$\dot{x} = J\dot{\theta} \tag{2.4}$$

Equation (2.3) and eq. (2.4) are crucial because they allow the determination of the solution of an inverse kinematics through Jacobian Matrices, by giving as input the position and orientation of the end-effector pretended. It is essential to mention that this solution also depends on the robot's specifications and, consequently, on the analytical solution since it also has numerical problems related to the physiognomy and redundancies of the robot in question.

In the past, multiple attempts were made to deal with redundancies on different robots [7, 8]. Those tests were even made on anthropomorphic robot arms (manipulators that have a similar structure to human arms), as [9–11], which identify the difficulties of handling redundancies, finding solutions for the robot/program in the study.

The same strategy was approached on [12] by P. Dahm and F. Joublin, who tried to compute an IK considering the shoulder and the wrist's positions static. These initial conditions were achieved since this approach relied on a robot whose body was composed only of its arm, where its shoulder defined the first joint of the robot. The wrist's immobile position came from the robot's anatomy that by analysing the analytical inverse kinematics, it would be possible to achieve its position. These premises allowed that, afterwards, the position of the elbow was found out through the restriction of the angles and the space available in between. This approach had great importance in this master thesis despite its limitations once the end-effector needs to be within the range of the robot's arms (explained in the next paragraph) so that the robot can reach the target. The same problem concerning the angles not being matchable was found when this approach was tested and verified later on [13]; however, in this case, an approach to acquire the transformation that needed to be calculated from the hand until the wrist (for instance, z_8 in fig. 3.5) was found out.

Regarding the range of the robot's arm mentioned before, several researches have been carried out in order to realise the importance of the anatomy of the robot in the possible solutions of the inverse kinematics. M. Bugday and M. Karali [14] and R. Raja, A. Dutta and B. Dasgupta [15] are examples of how it is possible to stablish the feasible regions for the joints of the arm obtained from the IK considering first a static approach relying on the sizes of the robot and, afterwards, an approach for a mobile robot considering its movement.

All the thesis referred before did not find a suitable solution for the possibility of dealing with more than 6 DOF and how to approach all those redundancies. However, a different approach was considered

on [16] which defined that it would be possible to assume that the two reference frames of the two joints present in the shoulder could always be considered in the same plane, in the plane X-Y. This would simplify the choice of the position for the shoulder and, consequently, the elbow's position and orientation.

At the same time, it would be possible to combine the analytical method by Asfour and Dillman [16], with the one provided by Xinyang Tian, Qinhuan Xu, Qiang Zhan [17] since it defends the usage of a static arm combined with a wrist's position known, to determine the possible positions for the elbow. This approach allows to build an intuitively Inverse-Kinematics model for humanoid arms with the usage of the fewer parameters as possible to get all of the robot's joints.

In sum, these Analytical Methods created for the inverse-kinematics model needed to be tested against other approaches on the market and, this is the reason why, this master thesis is going to compare the Analytical method with the ones created through Control based methods. This comparison was made through ROS (Robot Operating System) [18], which provides a set of software libraries and tools that help to build robot applications, such as trac_ik [19] and IKFast [20] Kinematics Solvers.

The first one, trac_ik [21] is an alternative Inverse Kinematics solver that receives as input the position and orientation of the end effector. It considers the initial position as the origin and concurrently runs two IK implementations: Inverse Jacobian Algorithm and a Sequential Quadratic Programming (SQP) nonlinear optimization approach. These 2 IK implementations run independently and return whenever one of those converges to an answer.

The first Inverse Kinematics computed by trac_ik is an improved KDL Inverse Jacobian Algorithm based on Newton's convergence algorithm that detects and mitigates local minima due to joint limits by random jumps. Newton's convergence is essential here because it is the one responsible for finding successively better approximations to the zeroes of a real-valued function. This strategy is typically used when the number of equations is different from the number of variables or if the Jacobian cannot be assumed nonsingular.

Regarding the Sequential Quadratic Programming nonlinear optimization, it is also a method used to find the numerical solution to constrained nonlinear optimization problems by using a variety of quadratic error metrics that better handles joint limits. Thus, the SQP is an iterative procedure that generates iterates converging to a solution of the problem defined in eq. (2.1) and eq. (2.2) by solving quadratic programs that are approximations to the nonlinear programming problems, defined as:

$$\begin{aligned} &\min_{\theta} p(\theta), \\ st: &\, h(\theta) = 0, \\ &\, g(\theta) \leq 0, \end{aligned} \tag{2.5}$$

Where $p : \mathbb{R}^n \rightarrow \mathbb{R}; h : \mathbb{R}^n \rightarrow \mathbb{R}^m, g : \mathbb{R}^n \rightarrow \mathbb{R}^p$. In this specific context, p represents the distance between the target pose $x = (T_O, T_P)$ and the pose computed by $f(\theta_1, ..., \theta_n)$. Therefore, the SQP is the nonlinear optimization responsible for minimizing the distance between the end-effector and the solution created by the forward kinematics function.

In sum, Trac_ik not only presents a high-efficiency rate computing the inverse kinematics on various humanoid platforms [22] (compared with the other pseudoinverse Jacobian IK solvers, such as Orocos Kinematics and Dynamics Library (KDL) [23], that still exhibit a high rate of false-negative failures), but also a high-speed test finding the final goal. This method will be compared with an analytical method to develop the best algorithm possible.

On the other side, ROS also allows the computation of an analytical based method, IKFast, which is the standard one, once it automatically computes an analytical solution, based on the analyses of any complex kinematic chain for common patterns, with the most stable solutions in shortest time [24]. However, it also has a limitation since IKFast theoretically works with 6 degrees of freedom. For a 7DOF robot arm manipulator, it requires to pick the joint with the minor importance and discretize the range of that joint during planning. This discretization tests all values until a solution for the joint limits, the collision, and the planning constraints is found. In theory, it would not be a problem, but since the purpose is to create the inverse kinematics for all body and have the possibility to, in the future, choose the best solution between both arms, this approach will not be taken in account.

All models considered and analyzed until now were inverse kinematics models limited only to the robot's arms, which led to the fact that the implementation of inverse kinematics, including the base, still was necessary.

2.2 IK Including the Base

In any robot that is composed of two parts that can be dealt with separately, namely the base and the trunk (leading obviously to the arms), two different approaches are used in the industry nowadays [25]. The first one consists of choosing the base's position before the computation of the inverse kinematics for the arms, and the second one is based on solving the inverse kinematics for the robot's arms and only at that point determine the position of the base.

Starting the problem by defining the position of the platform's base, it is required the understanding of how far the base could be from the end-effector [26], which depends on the joints of the robot and its articulations' length as well [27]. After that, the positions of the space are tested through discretization of the space as it can be shown in [28, 29]. This approach implies very fast inverse kinematics through an analytical or control-based method since the efficiency of the IK depends on a discretization of hundreds of thousands of potential positions for the base [26]. It is important to emphasize that the

discretization always depends on the specifications of the robot and the way that the inverse kinematics performs/adapts to slight variances on its angles/positions [30].

The second way executes the inverse kinematics firstly and, only afterwards, defines the position of the base. As shown in [31] for the KUKA robot, the idea consisted of projecting the end-effector orientation to the floor plane, which provided the orientation of the mobile platform. This heuristic works very well for Youbot because of the physical limitations of the robot, but in the case of a humanoid robot, it has very few physical limitations of the end-effector that could be taken advantage from.

Another way to do it, as it is referred to in [32], is by using a Data-driven approach, which alleviates the requirement for complete mechanical and mathematics descriptions, requiring only the use of pre-learned postures to match the positions of the end-effectors to a feasible pose learned from the database. In order to do that, it is necessary to execute the forward kinematics method successively to reach a uniform distribution of the points in the space, which is a disadvantage because if the mechanical description is not known, it implies that it is not known if the points are uniformly distributed in the space as well. Besides that, the training data is ambiguous, resulting in limitations since multiple joint configurations may have the same position in task space, leading to low accuracy, mainly when the threshold set is not low enough. On the other side, it requires much memory and, at the same time, the usage of algorithms to acquire the angles pretended most efficiently.

At this point, it is possible to have an overview of what are the market's strategies regarding inverse kinematics. In order to deepen the understanding and further develop these strategies for *Vizzy*, the robot's structure must be studied.

3

Implementation

Contents

In contemplation of using an Analytical method analysis, the *Vizzy*'s Anatomy and limitations must be studied.

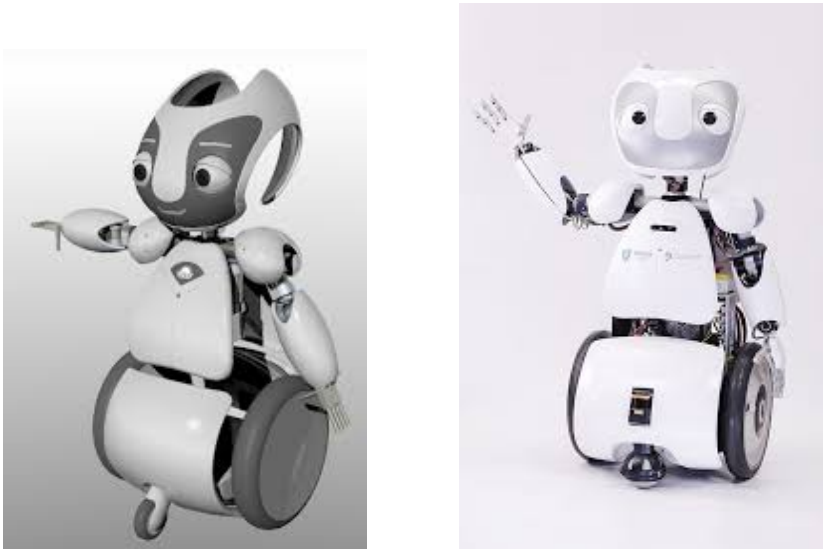


Figure 3.1: *Vizzy* Robot

3.1 *Vizzy*'s Specifications

Vizzy is a humanoid robot with a particular purpose: to serve and assist people in daily life tasks [33]. This goal requires a deep study of all possible movements and perceptions of the robot and even more profound research on algorithms and optimizations that allow the robot to have the most efficient and "cleaner" movement possible. Figure 3.1 shows different pictures of the robot in evaluation and table 3.1 compares the range of human movements against *Vizzy*'s range of movements.

Joint			**Standard human**	**Vizzy**
Head		Rotation	-70 to 70	-53 to 53
		Neck flexion	-50 to 60	-18 to 37
		Eye rotation	-40 to 40	-38 to 38
		Eyes flexion	-40 to 40	-38 to 38
Arm	Shoulder	scapula	-45 to 130	-18 to 18
		flexion	-60 to 180	-135 to 75
		abduction	-135 to 90	-75 to 0
		rotation	-90 to 90	-85 to 85
	Elbow	flexion	-150 to 0	-110 to 0
	Forearm	pronation	-90 to 90	-85 to 85
	Wrist	abduction	-20 to 50	-35 to 35
	Wrist	flexion	-70 to 90	-35 to 35

Table 3.1: Range of human movements vs. *Vizzy*'s range of movements. All the angular values are in degrees

This robot is based on human characteristics, and inevitably, the study of the robot was divided into two different parts: segway platform and all its body upwards, more specifically: both arms. The behaviour of both parts will be evaluated according to their specifications and their efficiency in developing the inverse kinematics. Regarding the arms, it had not only angle limitations (fig. 3.2), but also all its Danvit-Hartenberg parameters defined, which gave 9 DOF (7 at the arm and two on the rest of the body). Those limitations are based on *Vizzy*'s anatomy, which will define all the possible movements that the robot will be capable of.

Kin. chain	Joint	A	D	α	θ offset	Limits
Left Arm	Waist	0	0.0805	90	0	-20 to 20
	shoulder scapula	0	-0.212	90	0	-18 to 18
	shoulder flexion	0	-0.10256	90	-90	-75 to 135
	shoulder abduction	0	0	90	110	0 to 75
	shoulder rotation	0	-0.16296	90	90	-85 to 85
	Elbow flexion	0	0	90	0	0 to 110
	Forearm pronation	0	0.18635	90	0	-85 to 85
	Wrist abduction	0	0	90	-90	-35 to 35
	Wrist flexion	0.1	0	90	180	-35 to 35
Right Arm	Waist	0	0.0805	90	0	-20 to 20
	shoulder scapula	0	0.212	90	0	-18 to 18
	shoulder flexion	0	0.10256	90	-90	-75 to 135
	shoulder abduction	0	0	90	110	0 to 75
	shoulder rotation	0	0.16296	90	90	-85 to 85
	Elbow flexion	0	0	90	0	0 to 110
	Forearm pronation	0	-0.18635	90	0	-85 to 85
	Wrist abduction	0	0	90	-90	-35 to 35
	Wrist flexion	-0.1	0	90	180	-35 to 35

Figure 3.2: *Vizzy*'s Denavit-Hartenberg parameters. A and D units are meters and α and θ offset are in degrees.

Despite having 3 degrees of freedom on the shoulder, they are not all positioned on the same spot. This master thesis will consider the insider part of the shoulder as S_1 (defined as z_2 on fig. 3.5) and the position of the outside part of the shoulder (where are localized z_3 and z_4) as S_2, as it can be seen in Figure 3.3. Besides that, z_5 with z_6, z_7 with z_8 and z_9 represent the elbow, the wrist and the palm of the hand, respectively.

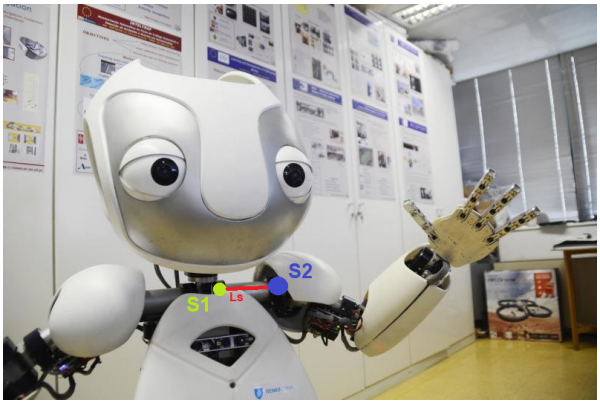


Figure 3.3: Representation of S_1 and S_2

In Figure 3.4, it is possible to verify the system of colors used to define the reference axis z,y, and x, represented by the blue, green, and red arrows, respectively. At the same time, Figure 3.5 also shows the nomenclature used for each link size, which appears in the transformation matrix between joints, where L_{Waist} represents the size of the waist, L_{Trunk} represents the dimensions of the trunk, L_S the shoulder, L_U the upper arm, L_F stands for the forearm and last but not the least, L_H represents the dimensions of *Vizzy*'s hand. The following Table 3.2 gives the exact dimensions of each one:

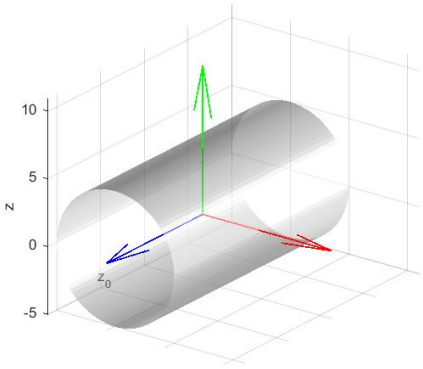
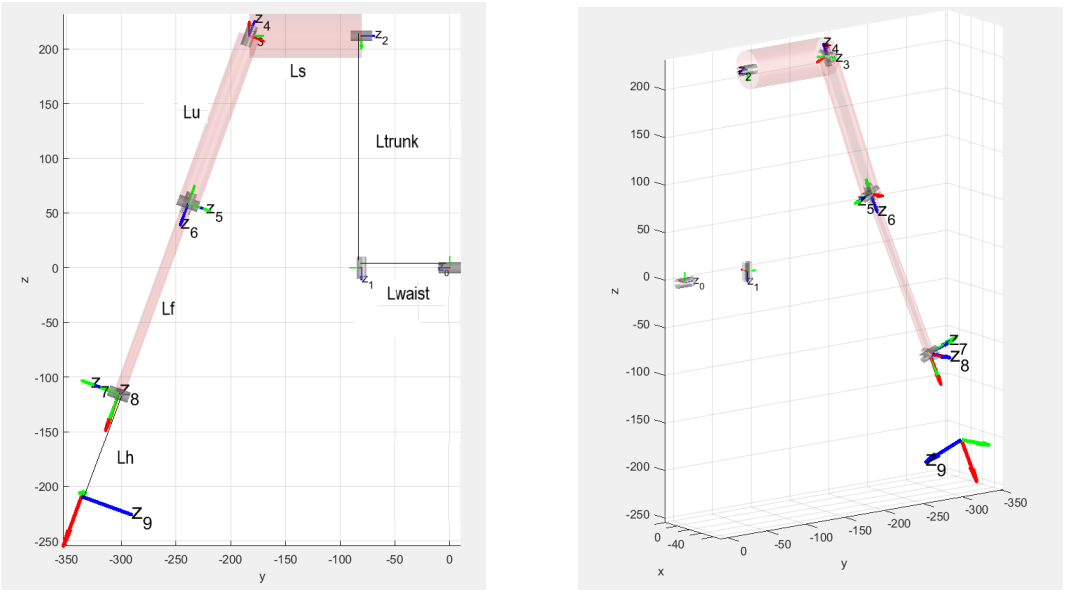


Figure 3.4: Reference axis' z, y and x



(a) Y - Z view in the world reference frame.

(b) View From Behind

Figure 3.5: Joint Angles' Orientation for Left Arm

Regarding the description of the coordinate transformation between frame (j$-$1) and frame j, the following equation defines the homogeneous transformation matrix, having in consideration the *Vizzy*'s

Dimension	Size (mm)
L_{Waist}	80.5
L_{Trunk}	212
L_S	102.56
L_U	162.96
L_F	186.35
L_H	100

Table 3.2: Different dimensions for *Vizzy*

Denavit-Hartenberg parameters:

$$G_{(j-1)j}(a,d,\alpha,\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta)\cos(\alpha) & \sin(\theta)\sin(\alpha) & a\cos(\theta) \\ \sin(\theta) & \cos(\theta)\cos(\alpha) & -\cos(\theta)\sin(\alpha) & a\sin(\theta) \\ 0 & \sin(\alpha) & \cos(\alpha) & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

There is a slight exception on equation eq. (3.1). When *Vizzy* was created, there was supposed to exist another joint between z_1 and z_2; however, it did not happen in practice. That is why there is not G_{23}, and this one was replaced by G_{34} (the same happens for the joints and transformations that come afterwards).

3.2 Analytical Method for *Vizzy*

In order to allow the robot *Vizzy* to reach the target, it is essential to subdivide the main problem into small pieces, not only to simplify, but also to make it attainable. Thus, the first step will be to calculate all the angles that constitute the arm through an analytical inverse kinematics method, considering the base as the origin and, at the same time, static. This initial calculation will be made only for reachable end-effectors, which means that will only be considered end-effectors computed through forward kinematics. This analytical method will be compared, after implemented, with another method implemented with control-based methods.

Furthermore, an original random position for the segway will be included so that the robot has to determine possible positions for its base and, at the same time, determine the inverse kinematics responsible for achieving the final goal. This means that a translation and rotation must be applied to the base so that the inverse kinematics can be calculated posteriorly.

From now on, to define a transformation, there will be used a notation that depends on a rotation and a translation, which can be expressed as:

$$T = \begin{bmatrix} T_{\overrightarrow{e_x}} & T_{\overrightarrow{e_y}} & T_{\overrightarrow{e_z}} & T_P \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} T_R & T_P \\ 0 & 1 \end{bmatrix} \tag{3.2}$$

where T_P represents the position and $T_R = (T_{\overrightarrow{e_x}}, T_{\overrightarrow{e_y}}, T_{\overrightarrow{e_z}})$ its orientation.

3.2.1 Analytical Method for the *Vizzy*'s arms with Base in the origin

By providing the target end-effector to a function responsible for the Inverse Kinematics Method, this function must be capable of finding out the positions and orientations for all joints that compose the *Vizzy*'s arms to achieve the final goal.

Before obtaining the Inverse Kinematics for both *Vizzy*'s arms, it is essential, by analyzing Figure 3.5, to emphasize that the base (z_0) will always have coordinates $(x, y, 0)$, since it belongs to the xOy plane. Besides that, since the base is supposed to be fixed at this initial stage, it was assumed that the base is not only static but also at the origin of the referential, which led to its correspondent transformation matrix eq. (3.3):

$$G_{00}(angle\_base) = \begin{bmatrix} \cos(angle\_base) & 0 & \sin(angle\_base) & 0 \\ \sin(angle\_base) & 0 & -\cos(angle\_base) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

Equation 3.3: G_{00} with Denavit-Hartenberg (DH) parameters: $a = 0$, $d = 0$, $\alpha = \frac{\pi}{2}$ and $\theta = angle\_base$.

As it can be seen in eq. (3.3), G_{00} always depends on the $angle\_base$, which is the angle responsible for the rotation of the robot through itself, as it can be seen in Figure 3.6. Regarding the $angle\_base$, since it is pretended to compute an analytical method only for the arms, it was considered that the $angle\_base$ is going to be an input of the system and, only in the next chapter, its value is going to be updated.

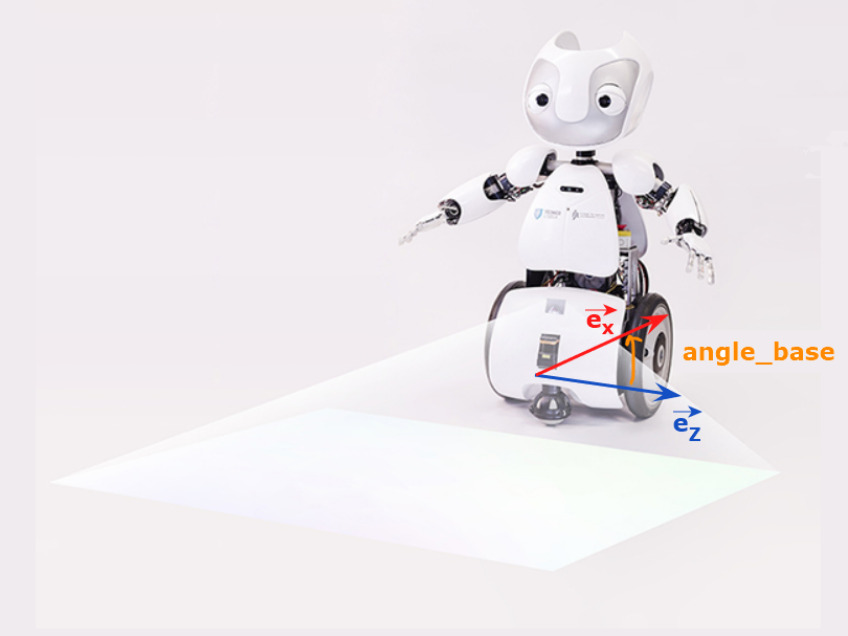


Figure 3.6: Representation of the $angle\_base$ in 3D

3.2.1.A Position and Orientation of the Shoulder S_1

In order to compute the Inverse Kinematics for the robot's arms, it requires that the position and orientation of the shoulder are established. However, from fig. 3.5, it is possible to acquire that there are still 2 joints that influence the shoulder's position, which are the waist angle (defined by z_0) and the shoulder's scapula angle (z_1). For this reason, the matrix G_{sL2} was created, representing the transformation matrix responsible for going from the base until the shoulder. Thus, G_{sL2} is composed by G_{00}, G_{01} and G_{12} as it can be seen in eq. (3.6).

$$G_{01}(\theta_0) = \begin{bmatrix} \cos(\theta_0 + \frac{\pi}{2} + \frac{\pi}{2}(-1)^s) & 0 & \sin(\theta_0 + \frac{\pi}{2} + \frac{\pi}{2}(-1)^s) & 0) \\ \sin(\theta_0 + \frac{\pi}{2} + \frac{\pi}{2}(-1)^s) & 0 & -\cos(\theta_0 + \frac{\pi}{2} + \frac{\pi}{2}(-1)^s) & 0) \\ 0 & 1 & 0 & L_{Waist}(-1)^{s+1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

Equation 3.4: G_{01} with DH parameters: $a = 0$, $d = L_{Waist}(-1)^{s+1}$, $\alpha = \frac{\pi}{2}$ and $\theta = \theta_0 + \frac{\pi}{2} + \frac{\pi}{2}(-1)^s$.

$$G_{12}(\theta_1) = \begin{bmatrix} \cos(\theta_1) & 0 & \sin(\theta_1) & 0 \\ \sin(\theta_1) & 0 & -\cos(\theta_1) & 0 \\ 0 & 1 & 0 & L_{Trunk}(-1)^s \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

Equation 3.5: G_{12} with DH parameters: $a = 0$, $d = L_{Trunk}(-1)^s$, $\alpha = \frac{\pi}{2}$ and $\theta = \theta_1$.

$$G_{sL2} = G_{00}G_{01}G_{12} \tag{3.6}$$

Regarding Equation (3.4) and Equation (3.5), both depend on their respective angle variation, θ_0 and θ_1. However, as it is possible to check in Figure 3.2, both have a very narrow range of values that are capable of performing, which led to an assumption that both would be considered equal to 0 in order to compute the inverse kinematics. Consequently, G_{01} and G_{12} matrixes were updated with the new values as follows:

$$G_{01}(0) = \begin{bmatrix} \cos(\frac{\pi}{2} + \frac{\pi}{2}(-1)^s) & 0 & \sin(\frac{\pi}{2} + \frac{\pi}{2}(-1)^s) & 0) \\ \sin(\frac{\pi}{2} + \frac{\pi}{2}(-1)^s) & 0 & -\cos(\frac{\pi}{2} + \frac{\pi}{2}(-1)^s) & 0) \\ 0 & 1 & 0 & L_{Waist}(-1)^{s+1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.7}$$

Equation 3.7: G_{01} with DH parameters: $a = 0$, $d = L_{Waist}(-1)^{s+1}$, $\alpha = \frac{\pi}{2}$ and $\theta = \frac{\pi}{2} + \frac{\pi}{2}(-1)^s$.

$$G_{12}(0) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & L_{Trunk}(-1)^s \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.8}$$

Equation 3.8: G_{12} with DH parameters: $a = 0$, $d = L_{Trunk}(-1)^s$, $\alpha = \frac{\pi}{2}$ and $\theta = 0$.

By analysing Equation (3.7) and Equation (3.8), it is possible to verify that G_{01} represents a transformation on xy plane, while G_{12} represents a transformation only influencing the z component. However, adding to the problem the assumptions considered before that $\theta_0 = \theta_1 = 0$, allow to considered that z_1 has the z component fixed and only changes the x and y values if the base changes. For now, since the base is fixed to the origin, its values are already known by Equation (3.6).

At the same time, it is also possible to understand that both equations depend on a variable s, which is a constant that is responsible for identifying which arm is being used, where the constant $s = 0$ represents the right side, while $s = 1$ represents the left one. This constant will appear during all homogeneous transformation matrices that lead to the end-effector.

3.2.1.B Shoulder's side choice

By having the position of each shoulder, the choice of the arm used to perform the inverse kinematics will affect all the process afterwards. There are 2 factors that will have direct impact on the choice of the side, such as the energy cost and the obstacles on the path. Since at this point, obstacles aren't being considered, the turning point was the distance between the shoulders S_1 already acquired and the end-effector, which means that the shoulder that is closer to the target is the one that is going to be used afterwards.

3.2.1.C Position and Orientation of the Shoulder S_2

At this point, it is important to take into consideration that eq. (3.6) only allows the calculation of S_1, not the shoulder S_2. To get S_2, it is necessary to have G_{34} (transformation that goes from the insider part of the shoulder (z_2) to the furthest one(z_3)). This transformation is calculated with θ_2, which is still unknown, since θ_2 is going to be found out only when the inverse kinematics is computed.

However, as it is explained in [16], in contrast to the human body, *Vizzy* doesn't have the Scapulothoracic joint, which is the responsible for the elevation of the shoulder [34]. This fact means that G_{00}, G_{01} and G_{12} are the only responsible for the z component of the joint z_2.

Since z_1 is already known (Position and Orientation of the Shoulder S_1) and, at the same time knowing that G_{34} also represents a transformation on the xy plane, this means that, not only $z_{S_2} = z_{S_1}$, but also allows to determine the exact position of S_2 (which is independent from the value of θ_2, since only

impacts its orientation, not its position). These calculations are made through Equation (3.9) and Equation (3.10) (with an arbitrary θ_2), that represent the transformations that go from S_1 to S_2 and from the base to S_2, respectively, which implies that the forth column of this matrix G_{sL3} contains the position of the shoulder S_2 that is pretended (calculated in eq. (3.11)).

$$G_{34}(\theta_2) = \begin{bmatrix} \cos(\theta_2 - \frac{\pi}{2}) & 0 & \sin(\theta_2 - \frac{\pi}{2}) & 0 \\ \sin(\theta_2 - \frac{\pi}{2}) & 0 & -\cos(\theta_2 - \frac{\pi}{2}) & 0 \\ 0 & 1 & 0 & L_s(-1)^s \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

Equation 3.9: G_{34} with DH parameters: $a = 0$, $d = L_s(-1)^s$, $\alpha = \frac{\pi}{2}$ and $\theta = \theta_2 - \frac{\pi}{2}$.

$$G_{sL3} = G_{sL2}G_{34}; \tag{3.10}$$

$$S_2 = (G_{sL3})_P; \tag{3.11}$$

For now, the position of the shoulder that is going to be used to perform the inverse kinematics is already known, only missing its orientation. To obtain it, it is necessary to acquire the position of the wrist as it is going to be calculated in section 3.2.1.D.

3.2.1.D Position of the Wrist

As the input of this system, the robot will receive both position and orientation of the end effector, which will consequently allow obtaining the position of the wrist. The orientation of the goal is beneficial because it announces where the wrist is going to be through its x coordinate (as it can be shown with an example in fig. 3.7).

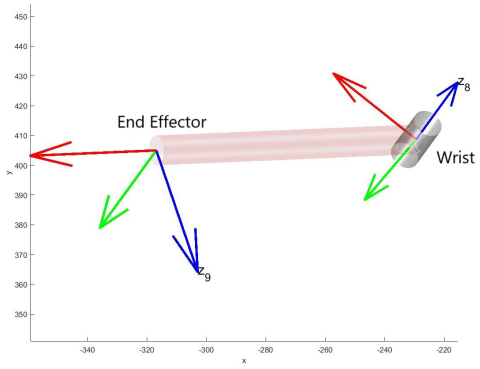


Figure 3.7: Wrist Position X - Y view in the world reference frame

Thus, the wrist's position is deductible as follows:

$$W = (T_{EE})_P - L_H(T_{EE})_{\overrightarrow{e_x}}; \tag{3.12}$$

Where W represents the position of the wrist and T_{EE} represents the end-effector's transformation matrix.

3.2.1.E Position of the Elbow

From the section 3.2.1.C, the position of the shoulder is well known. At the same time, from the previous subsection (Position of the Wrist), the position for the wrist is already computed as well. This means that there is only one position that is not yet calculated: the elbow.

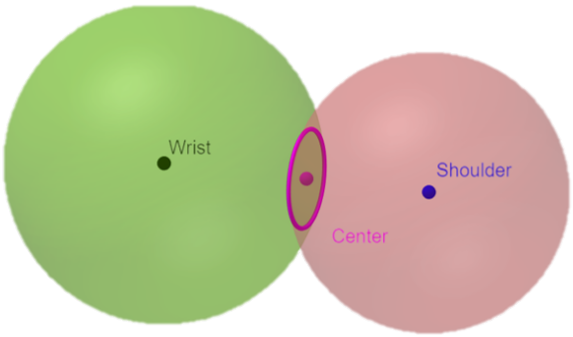


Figure 3.8: Possible Positions for the elbow with Spheres' Intersection

The possible positions for the elbow are easy to induce from the fact that the elbow position has to belong to the sphere centered in the shoulder S_2 and radius equals to the size of the upperarm(L_u), but also belong to the sphere centered in the wrist with radius equals to the size of the forearm(L_F), fig. 3.8. It can be shown that the intersection of these two spheres results in a circumference or a point, hence the segway platform was placed in a position that doesn't allow the null intersection. In the Section 3.2.2, will be explained how the algorithm will handle the case where the intersection is null.

Besides resulting in a circle or in a point, it is also possible to know where is its center. This results can be derived as follows:

Let's consider C as the circumference resulting from the intersection of both spheres, α as the plane where C is included and S_2, W as the positions of the shoulder and the wrist, respectively. So:

$$C \in \alpha \implies \forall point G \in C : \|\overrightarrow{S_2G}\| = \|\overrightarrow{WG}\| \frac{L_u}{L_F} \tag{3.13}$$

In order to simplify the nomenclature, let's assume:

$$\|\overrightarrow{r_u}\| = \|\overrightarrow{S_2G}\|; \|\overrightarrow{r_F}\| = \|\overrightarrow{GW}\| \tag{3.14}$$

$$\|\overrightarrow{r_w}\| = \|\overrightarrow{S_2M}\| + \|\overrightarrow{MW}\| \tag{3.15}$$

for M as the center of the circumference, $\overrightarrow{r_u}$ as the vector that goes from the shoulder until the elbow, $\overrightarrow{r_F}$ as the vector that goes from the elbow until the wrist and $\overrightarrow{r_w}$ as the vector that goes from the shoulder until the wrist, as it can be seen in fig. 3.9.

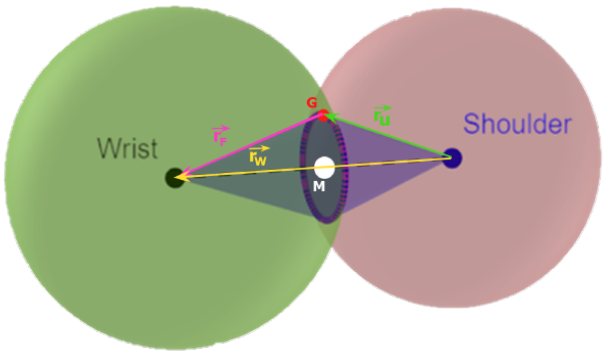


Figure 3.9: Possible positions for the elbow with vectors considered

By analysing eq. (3.13), it allows to define 2 cones with the base in α and slant heights equals to $\|\overrightarrow{r_u}\|$ and $\|\overrightarrow{r_F}\|$, as shown in fig. 3.10. Since the bases of the cones are exactly the same, it is guaranteed that both heights must be perpendicular to plane α , which implies that both heights are in the same straight line. At the same time, is also possible to assume that:

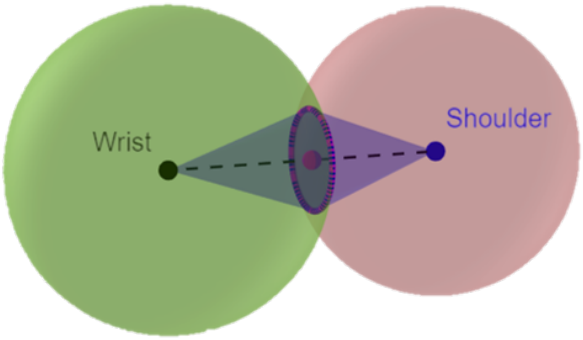


Figure 3.10: Cones in Spheres

$$\overrightarrow{MW} + \overrightarrow{GM} = \overrightarrow{r_F} \tag{3.16}$$

Overall, after realising that the center of the circle belongs to the straight line that goes through the shoulder and the wrist, it is also possible to find out 3 equations that lead to the computation of the radius:

$$(\overrightarrow{r_W} - \overrightarrow{S_2M}) + \overrightarrow{GM} = \overrightarrow{r_F} \Longrightarrow (\|\overrightarrow{r_w}\| - \|\overrightarrow{S_2M}\|)^2 = \|\overrightarrow{r_F}\|^2 - \|\overrightarrow{GM}\|^2 \tag{3.17}$$

$$\overrightarrow{S_2M} + \overrightarrow{MG} = \overrightarrow{r_u} \iff \|\overrightarrow{S_2M}\|^2 = \|\overrightarrow{r_u}\|^2 - \|\overrightarrow{MG}\|^2 \tag{3.18}$$

$$\|\overrightarrow{GM}\|^2 = \|\overrightarrow{MG}\|^2 \tag{3.19}$$

Therefore, the radius of the circumference can finally be found by solving the 3 previous equations as shown is equation eq. (3.20).

$$\left(\|\overrightarrow{r_w}\| - \sqrt{\|\overrightarrow{r_u}\|^2 - \|\overrightarrow{MG}\|^2}\right)^2 = \|\overrightarrow{r_F}\|^2 - \|\overrightarrow{MG}\|^2 \iff$$

$$\iff \|\overrightarrow{r_w}\|^2 - 2\|\overrightarrow{r_w}\|\sqrt{\|\overrightarrow{r_u}\|^2 - \|\overrightarrow{MG}\|^2} + \|\overrightarrow{r_u}\|^2 - \|\overrightarrow{MG}\|^2 = \|\overrightarrow{r_F}\|^2 - \|\overrightarrow{MG}\|^2 \iff$$

$$\iff R = \|\overrightarrow{MG}\| = \sqrt{\|\overrightarrow{r_u}\|^2 - \left(\frac{\|\overrightarrow{r_u}\|^2 - \|\overrightarrow{r_F}\|^2 + \|\overrightarrow{r_w}\|^2}{2\|\overrightarrow{r_w}\|}\right)^2} \tag{3.20}$$

Regarding the position of the center of the circumference, through eq. (3.15), it can be shown that

$$\overrightarrow{S_2M} = \overrightarrow{r_w}\frac{L_u}{L_u + L_F} \tag{3.21}$$

$$M = S_2 + \overrightarrow{S_2M} \tag{3.22}$$

At this point, the circle is completely defined with the radius and the center of the circumference. However, the final position for the elbow is still needed, because this circumference generates infinite possibilities, which brings another redundancy to evaluate. This implies that something must be assumed in order to keep going. For this purpose, the variable α was created and assumed as represented in fig. 3.11.

Furthermore, the application of α is going to be studied, but for now, α is going to be an input of the system, and is going to adjust the position of the elbow. By knowing α, and by expressing r_w in spherical coordinates as:

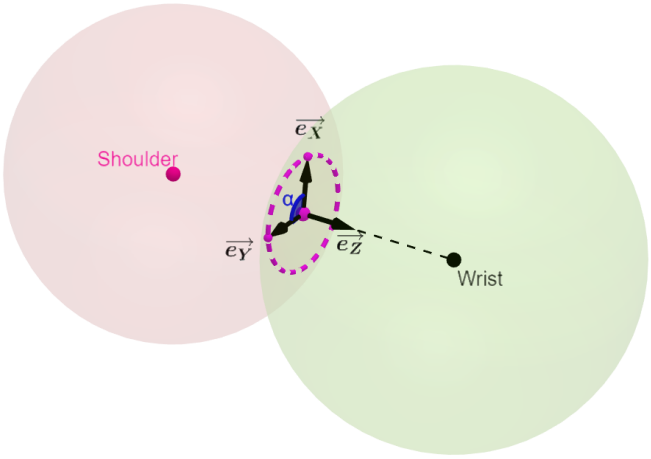


Figure 3.11: Representation of α

$$r_w = \begin{bmatrix} \vartheta_w \\ \varphi_w \\ \|\overrightarrow{r_w}\| \end{bmatrix} \tag{3.23}$$

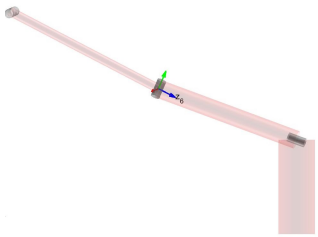
The elbow's position is obtained by

$$E = (R_z^{\varphi_w} R_y^{\vartheta_w} R_z^{\alpha} e_x)R + M \tag{3.24}$$

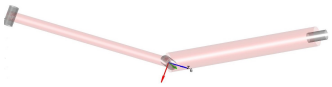
where R_z represents r_w's the rotation matrix around z-axis. At this point, the position of the elbow is known, however, the unit vectors of the frame z_5 (fig. 3.5) attached to the elbow must be established.

3.2.1.F Orientation of the elbow

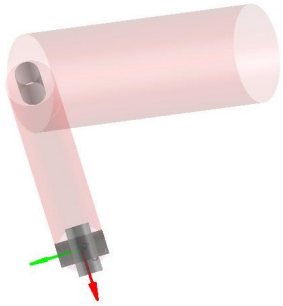
The orientation of the elbow is going to be one of the most important in the process of the inverse kinematics. To acquire the orientation of the elbow, let's consider $(\overrightarrow{e_x}, \overrightarrow{e_y}, \overrightarrow{e_z})$ as the unit vectors of the frame attached to the elbow, z_5.



(a) X-Y View in the world reference frame.



(b) X-Z view in the world reference frame.



(c) Perpendicular view to r_F

Figure 3.12: Elbow Orientation

By analysing the Figure 3.12, the vector $\overrightarrow{e_z}$ lies along the straight line defined by the vector $\overrightarrow{EW}$ (the forearm) and points on the opposite direction of the wrist for the right arm (the opposite happens for the left arm). The vector $\overrightarrow{e_y}$ is perpendicular to $\overrightarrow{e_z}$ and also to $\overrightarrow{r_u}$. By computing the right-handed coordinate frame, the last vector, $\overrightarrow{e_x}$, is obtained. This can be be formulated as:

$$\overrightarrow{e_z} = (-1)^{s+1} \frac{\overrightarrow{EW}}{\|\overrightarrow{EW}\|} \tag{3.25}$$

$$\overrightarrow{e_y} = \frac{\overrightarrow{e_z} \times \overrightarrow{r_u}}{\|\overrightarrow{e_z} \times \overrightarrow{r_u}\|} \tag{3.26}$$

$$\overrightarrow{e_x} = \overrightarrow{e_y} \times \overrightarrow{e_z} \tag{3.27}$$

3.2.1.G Elbow's Inverse Kinematics

The calculation of all positions of *Vizzy*'s arm and its correspondent orientations, will allow the determination of all the *Vizzy*'s angles with Inverse Kinematics. In order to obtain those angles, a 4x4 homogeneous transformation matrix was created to both shoulder and elbow by using its position and orientation.

At this specific context, elbow's transformation matrix is also obtained through the product of the homogeneous transformation matrices (as represented in eq. (3.28)), which will allow to define the

transformation of the elbow as follows:

$$\begin{bmatrix} \overrightarrow{x_4} & \overrightarrow{y_4} & \overrightarrow{z_4} & E \\ 0 & 0 & 0 & 1 \end{bmatrix} = T_E = G_{00}G_{01}G_{12}G_{34}G_{45}G_{56}G_{67} \tag{3.28}$$

Where G_{45}, G_{56} and G_{67} can be described as:

$$G_{45}(\theta_3) = \begin{bmatrix} \cos(\theta_3 + 11\frac{\pi}{18}) & 0 & \sin(\theta_3 + 11\frac{\pi}{18}) & 0 \\ \sin(\theta_3 + 11\frac{\pi}{18}) & 0 & -\cos(\theta_3 + 11\frac{\pi}{18}) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.29}$$

Equation 3.29: G_{45} with DH parameters: $a = 0$, $d = 0$, $\alpha = \frac{\pi}{2}$ and $\theta = \theta_3 + 11\frac{\pi}{18}$.

$$G_{56}(\theta_4) = \begin{bmatrix} \cos(\theta_4 + \frac{\pi}{2}) & 0 & \sin(\theta_4 + \frac{\pi}{2}) & 0 \\ \sin(\theta_4 + \frac{\pi}{2}) & 0 & -\cos(\theta_4 + \frac{\pi}{2}) & 0 \\ 0 & 1 & 0 & L_u(-1)^s \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.30}$$

Equation 3.30: G_{56} with DH parameters: $a = 0$, $d = L_u(-1)^s$, $\alpha = \frac{\pi}{2}$ and $\theta = \theta_4 + \frac{\pi}{2}$.

.

$$G_{67}(\theta_5) = \begin{bmatrix} \cos(\theta_5 + 5\frac{\pi}{180}) & 0 & \sin(\theta_5 + 5\frac{\pi}{180}) & 0 \\ \sin(\theta_5 + 5\frac{\pi}{180}) & 0 & -\cos(\theta_5 + 5\frac{\pi}{180}) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.31}$$

Equation 3.31: G_{67} with DH parameters: $a = 0$, $d = 0$, $\alpha = \frac{\pi}{2}$ and $\theta = \theta_5 + 5\frac{\pi}{180}$.

From eq. (3.28), T_E depends on 4 variables: θ_2, θ_3, θ_4 and θ_5. The solution for this problem is too complex and an alternative was imperative to find. This alternative required the division of the problem in sub problems easier to solve. The first solution was a simple modification on the eq. (3.28), which resulted in:

$$\underbrace{(G_{34})^{-1}(G_{12})^{-1}(G_{01})^{-1}(G_{00})^{-1}T_{elbow}}_{L_1} = \underbrace{G_{45}G_{56}G_{67}}_{R_1} \tag{3.32}$$

Regarding eq. (3.32), L_1 only depends on 1 variable, θ_2, so after analysing R_1, it was possible to find out that the position (3,4) of R_1 is a constant for whatever angle. Therefore, this analyses provides the calculation of θ_2, since:

$$L_1(3,4) = R_1(3,4) \iff L_1(3,4) = L_u(-1)^s cos\Big(\frac{\pi}{2}\Big) \iff L_1(3,4) = f(\theta_2) = 0 \tag{3.33}$$

Once eq. (3.33) is solved and θ_2 acquired, L_1 starts to be a constant matrix. Consequently, the same

criteria can be used to obtain the angle θ_3, which have 2 equations that must be fulfilled:

$$L_1(1,4) = R_1(1,4) \iff L_1(1,4) = L_u(-1)^s sin\Big(\frac{11\pi}{18} + \theta_3\Big) \tag{3.34}$$

$$L_1(2,4) = R_1(2,4) \iff L_1(2,4) = L_u(-1)^{s+1} cos\Big(\frac{11\pi}{18} + \theta_3\Big) \tag{3.35}$$

At this point, θ_2 and θ_3 values are already acquired, however using eq. (3.32) without any alteration would bring a very hard computational problem to solve θ_4 and θ_5. Besides that, the time to estimate would be very extensive which led to a small change on eq. (3.32):

$$\underbrace{(G_{45})^{-1}(G_{34})^{-1}(G_{12})^{-1}(G_{01})^{-1}(G_{00})^{-1}T_{elbow}}_{L_2} = \underbrace{G_{56}G_{67}}_{R_2} \tag{3.36}$$

Duplicating the strategy used for θ_2, through eq. (3.36), and adding the fact that L_2 is a constant matrix as well (G_{45} is the homogeneous matrix related to θ_3, already calculated as well), θ_4 and θ_5 were possible to obtain. Since there wasn't any element of R_2 that only depended on θ_4, θ_5 was calculated first, as follows:

$$L_2(3,1) = R_2(3,1) \iff L_2(3,1) = sin\Big(\frac{\pi}{36} + \theta_5\Big) \tag{3.37}$$

$$L_2(3,3) = R_2(3,3) \iff L_2(3,3) = -cos\Big(\frac{\pi}{36} + \theta_5\Big) \tag{3.38}$$

Last but not the list, the only angle that wasn't found yet was θ_4 that could easily be found with eq. (3.39) which required to fulfill the following restrictions:

$$L_2(1,2) = R_2(1,2) \wedge L_2(2,2) = R_2(2,2) \tag{3.39}$$

At this point, all the angles that provide the forward kinematics from shoulder to elbow were found out, however, it's quite important to refer that, for each angle that is going to be find out, a function called "Check Constraints" is always called to guarantee that, not only, the solution is possible, but also, that the values obtained are feasible for *Vizzy*'s characteristics (not only for the actual but also for the previous angles). This function was created having in account the table referred before ('Range of human movements vs. *Vizzy*'s range of movements. All the angular values are in degrees' on page 17).

3.2.1.H Wrist's Inverse Kinematics

Contrarily to the elbow, the transformation matrix for the wrist is not yet known, owing to the fact that from subsection (Section 3.2.1.F) it was not possible to acquire its orientation since the position of the

elbow wasn't yet obtained.

So, the first step to obtain its transformation was the definition of the homogeneous transformation matrices that would transform the elbow on the wrist:

$$G_{78}(\theta_6) = \begin{bmatrix} \cos(\theta_6) & 0 & \sin(\theta_6) & 0 \\ \sin(\theta_6) & 0 & -\cos(\theta_6) & 0 \\ 0 & 1 & 0 & L_F(-1)^{s+1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.40}$$

Equation 3.40: G_{78} with DH parameters: $a = 0$, $d = L_F(-1)^{s+1}$, $\alpha = \frac{\pi}{2}$ and $\theta = \theta_6$.

$$G_{89}(\theta_7) = \begin{bmatrix} \cos(\theta_7 + \frac{\pi}{2}(-1)^{s+1}) & 0 & \sin(\theta_7 + \frac{\pi}{2}(-1)^{s+1}) & 0 \\ \sin(\theta_7 + \frac{\pi}{2}(-1)^{s+1}) & 0 & -\cos(\theta_7 + \frac{\pi}{2}(-1)^{s+1}) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.41}$$

Equation 3.41: G_{89} with DH parameters: $a = 0$, $d = 0$, $\alpha = \frac{\pi}{2}$ and $\theta_7 + \frac{\pi}{2}(-1)^{s+1}$.

$$G_{910}(\theta_8) = \begin{bmatrix} \cos(\theta_8 + \pi s) & 0 & (-1)^{s+1}\sin(\theta_8 + \pi s) & L_H\cos(\theta_8 + \pi s) \\ \sin(\theta_8 + \pi s) & 0 & -(-1)^{s+1}\cos(\theta_8 + \pi s) & L_H\sin(\theta_8 + \pi s) \\ 0 & (-1)^{s+1} & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.42}$$

Equation 3.42: G_{910} with DH parameters: $a = L_H$, $d = 0$, $\alpha = \frac{\pi}{2}(-1)^{s+1}$ and $\theta = \theta_8 + \pi s$.

Similarly to what was made on eq. (3.28), the same could be done at this point by assuming:

$$T^{''} = G_{78}G_{89}G_{910} \tag{3.43}$$

Which can be applied to the following equation:

$$T_{EE} = T_E T^{''} \iff T^{''} = \big(T_E\big)^{-1} T_{EE} \tag{3.44}$$

From eq. (3.43) and eq. (3.44), a readjustment is possible to be made, giving origin to:

$$\underbrace{\big(G_{78}\big)^{-1}\big(T_E\big)^{-1}T_{EE}}_{L_3} = \underbrace{G_{89}G_{910}}_{R_3} \tag{3.45}$$

Regarding eq. (3.45), L_3 only depends on 1 variable, θ_6, which would simplify to find θ_6 if there was any element on R_3 that would be constant. That hypothesis happens for the position (3,2) as it's shown in eq. (3.46).

$$\begin{aligned} L_3(3,2) &= R_3(3,2) \\ &= cos\Big(\frac{\pi}{2}\Big)sin\Big(\frac{\pi}{2}\Big)cos\Big(\pi s + \theta_8\Big) - cos\Big(\frac{\pi}{2}\Big)sin\Big(\frac{\pi}{2}\Big) \\ &= 0 \end{aligned} \tag{3.46}$$

After replacing θ_6, acquired before in L_3, θ_8 has to be calculated before θ_7, as it happened for the elbow where θ_5 was previously obtained than θ_4. Therefore, the calculation of θ_8 comes from:

$$L_3(3,1) = R_3(3,1) \iff L_3(3,1) = sin\Big(\frac{\pi}{2}\Big)sin\Big(\pi s + \theta_8\Big) = sin\Big(\pi s + \theta_8\Big) \tag{3.47}$$

$$L_3(3,3) = R_3(3,3) \iff L_3(3,3) = \Big(sin\Big(\frac{\pi}{2}\Big)\Big)^2 cos\Big(\pi s + \theta_8\Big) = cos\Big(\pi s + \theta_8\Big) \tag{3.48}$$

And, last but not least, it is possible to obtain the last angle, θ_7:

$$L_3(1,2) = R_3(1,2) \wedge L_3(2,2) = R_2(2,2) \tag{3.49}$$

In sum, all angles can be calculated in order to perform the Inverse Kinematics, admitting the fact that base is always at the origin, as explained in all section 3.2.1. Thus, in the next section 3.2.2, the same inverse kinematics is going to be developed but considering the possibility of moving the base.

3.2.2 Segway Platform

Regarding the base, the Inverse Kinematics must find a relation that combines both base and end-effector, so that the platform may be placed on a space where its computation is possible. As it was mentioned before, on these types of problems there are multiple approaches due to the degrees of freedom of the robot.

However, as it was proved before in fig. 3.8, the base's position is dependent on the possibility of the intersection of both spheres centered in the wrist and in the shoulder S_2. Therefore, to study the platform's base possible positions, the viable positions for S_2 must be evaluated.

3.2.2.A Position of the Shoulder (S_2)

Since $\theta_0 = \theta_1 = 0$ and the base belongs to the xOy plane, it is once more demonstrable that the shoulder belongs to the plane $z = z_{S_1} = z_{S_2}$ (Section 3.2.1.C). This aspect reduces the possible positions for the shoulder, once it only requires the study of the possible positions for shoulder's x and y values. At the same time, the position of the wrist (section 3.2.1.D) is known as before.

The fundamental idea is to reduce the shoulder's x and y positions of the robot through a circle (fig. 3.13), where is possible to ensure that, inside it, all positions are possible for the shoulder.

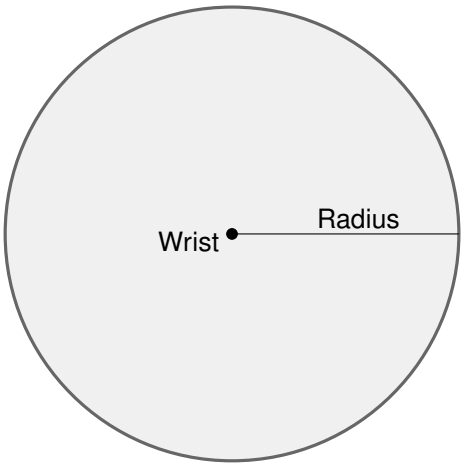


Figure 3.13: Circle of possible positions of the shoulder in the X-Y view in the world reference frame.

To clarify: this perspective does not ensure that there will be a position for the shoulder at all the possible positions inside the circle, but ensures that outside the circle there won't be any solution. Thereby, the value of the radius of that circle, through pythagorean theorem (as shown in fig. 3.14), could be obtained by the following equation:

$$Radius = \sqrt{\left(L_u + L_F\right)^2 - \triangle Z^2} \tag{3.50}$$

where $\triangle$Z represents the difference between the heights of the shoulder and the wrist, while $|L_u + L_F|$ represents the maximum distance between the shoulder and the wrist. This calculation would allow to bound the allowed positions of the shoulder.. This calculation would allow to bound the allowed positions of the shoulder.

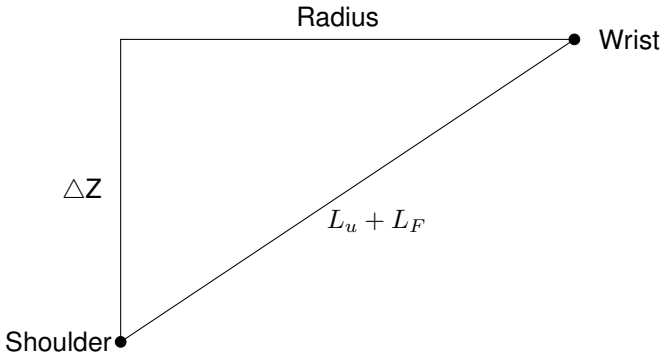


Figure 3.14: Triangle to obtain the radius of the circle

At this point, there is only a circle where the shoulder could be, but its position is yet unknown. However, since it was considered that $\theta_0 = \theta_1 = 0$ (section 3.2.1.C), it is possible to create a correspondence

between both positions and orientations, through the following equation:

$$(G_{34})^{-1}(G_{12})^{-1}(G_{01})^{-1}T_{S_2} = G_{00} \tag{3.51}$$

Where T_{S_2} represents the transformation that goes from the platform's base until S_2. In conclusion, by finding out a viable position and orientation for the shoulder, it would consequently be possible the determination of the position and orientation for the base (the reverse also happens as well), as it was wanted to be proved.

3.2.2.B Approach for the base

Although there are infinite positions for the base, it would be affected, not only by the circle of possible positions of the shoulder in xy view (since they are correlated by section 3.2.2.A), but also by the initial position of the robot, which would also affect the way the robot would move. At the same time, it is obviously always known that the position of the base is of the form $(x, y, 0), \forall x, y \in \mathbb{R}$.

Regarding the orientation of the base, G_{00}, it represents a transformation matrix where $\overrightarrow{e_y}$ always points upwards. This means that the only angle that will have a direct influence on this orientation is the one represented by $angle\_base$ ín fig. 3.6. This angle will be responsible for the rotation of the robot by itself with the help of its wheels.

One of the potential ways to approach this problem would be through a discretization of the xOy plane to different positions for the base and, at the same time, by applying a discretization for the $angle\_base$ as well [28]. This process was done at section 4.5, however, since the Kinematics is very sensible because the Kinematics by itself is not a continuous function, every slight shift could modify not only the solution but also the possibility of finding suitable Inverse Kinematics. Consequently, there was a risk of failing during this process, which led to the development of a different approach. As it was referenced on [32], there is always the possibility of defining a hybrid approach, using not only an Analytical Method to perform the Inverse Kinematics but also combining the results with a Data-driven approach.

Our approach consists of considering pre-learned postures generated through Forward Kinematics in order to find the closest transformation(s) to the end-effector. These pre-computed values are designed for 2 scenarios: Select just the $angle\_base$ value or retrieve the full pre-computed configuration when the IK solution is not found.

The first scenario would be important on the choice of the $angle\_base$ to compute the Inverse Kinematics since it would be impossible to compute for all $anglebase$ possible (from 0-360º), adding the fact that by computing the Inverse Kinematics, no solution was guaranteed. The choice of the transformation(s) that would be considered to compute the inverse kinematics would be made through L1 norm, as it can be shown in :

$$\min\|X\|_1 = \min_{1\leq j\leq m} \sum_{i=1}^{n} |x_{ij}| \tag{3.52}$$

Where:

$$X = AcquiredTranformation_R - EE_R \tag{3.53}$$

From eq. (3.53) it is possible to understand that the computation will use only the rotation matrices (without considering the T_P, referred in eq. (3.2), that would be adjusted at the end) from the pre-computed values, represented by AcquiredTranformation, and the End-Effector.

Since the purpose was only to find the closest, only would be considered the matrices that would fulfill the following requisition:

$$AcquiredTranformation_{P_z} - EE_{P_z} < 50 \tag{3.54}$$

Where eq. (3.54) defines that the highest difference between the high of the acquired transformation and the high of the end-effector could not ever be more significant than 50 mm. This 5 cm was considered because it represents the highest difference that guarantees that the robot will catch the end-effector due to its sizes and limitations. Nevertheless, for the closest acquired transformation(s), the value of the $angle\_base$ would be saved to be used in the analytical inverse kinematics. Regarding the position of the base, the value of x and y would be used as follows:

$$\begin{bmatrix} PlatformBase_{P_x} \\ PlatformBase_{P_y} \end{bmatrix} = \begin{bmatrix} EE_{P_x} - AcquiredTranformation_{P_x} \\ EE_{P_y} - AcquiredTranformation_{P_y} \end{bmatrix} \tag{3.55}$$

Which would adjust the difference between the position of the base and the position of the end-effector. If the IK would not work for those values, there was always the security of using the initial pre-computed values by only adjusting the platform's base. Always essential to notice that this security would always be the worst option because the analytical method acquires an exact inverse kinematics while this one always has an associated error. This error comes from the error associated with P_z, which can be under 5 cm, and the orientation, as it will be seen in section 4.5.

4

Results

Contents

One of the advantages of creating the Inverse Kinematics through an Analytical Method over a Control-based Method is that, by applying the calculations correctly and choosing wisely the redundancies, the efficiency is 100% for achievable end-effectors.

In order to prove this fact, this chapter will be divided into 2 subsections. The first one will test the efficiency of the algorithm created by comparing it with trac_ik. In the second subsection, the base will be introduced, and the tests for the base will be implemented as well.

4.1 Test with Inverse Kinematics

As it was explained in 3.2.1.E, the Inverse Kinematics method was released by knowing the exact α that would allow to identify the position of the elbow that has originated the forward kinematics. Thus, the tests were separated in two different parts: the first one with α known and the second one with α unknown.

4.1.1 Inverse Kinematics' tests with α known

In this case, for the algorithm developed in section 3.2.1, it was only necessary to define the end effector and the original position. In order to compare with the trac_ik algorithm later, it was considered that both initial conditions would be the same and equivalent to the ones defined in the previous chapter (which means that $\theta_0 = \theta_1 = angle\_base = 0$), so that the system would only have 7 DOF.

Due to the fact that these tests were made only with positions that were achievable by the robot, the choice of the angles was generated through forward kinematics, where we chose 3 possible values for each angle, and were made all the possible combinations with those 21 values for the angles, combining 2187 tests, as it is presented in the following table:

Theta	First Value	Second Value	Third Value
θ_2	25	-50	75
θ_3	25	50	75
θ_4	-27	50	-75
θ_5	30	60	90
θ_6	-27	54	81
θ_7	-10	20	-30
θ_8	-10	20	-30

Table 4.1: Combination of the angles that created the end-effectors to test the Analytical IK and the track_ik algorithms.

With the assumptions and with the positions and orientations generated in the Table 4.1, the following

results were obtained:

Algorithm	Alpha known?	Number of tests	Success Tests	Efficiency
Inverse Kinematics	Yes	2187	2187	100%

Table 4.2: Tests on IK with alpha known

As it can be seen, the algorithm works perfectly for these conditions, with an efficiency of 100%. However, it is known that trac_ik wouldn't have any parameter as input so, in order to compare them and both being in the same stage of equity, a possible value for the α angle needed to be found.

4.1.2 Inverse Kinematics' tests with α unknown

In order to compare with trac_ik algorithm, a value for α needed to be obtained. The solution momentously found for this problem was through the discretization of α in the range of $[-\pi, \pi]$.

In order to set the values, it was considered the following algorithm: α would start at $-\pi$ and would be added $\frac{\pi}{6}$ until a solution was found. If α assumed a value bigger than π, it would be considered π and subtracted half of its initial incrementation ($\frac{\pi}{12}$) until a solution could be found. If it wasn't possible to acquire α while α assumed a value higher than $-\pi$, the same procedure would be implemented by admitting $\alpha = -\pi$ again and the incrementation would be half as well ($\frac{\pi}{24}$). This method would be used until a solution was acquired and it was implemented as defined follows:

Algorithm 4.1: Alpha Momentously Solution

$[solutions, \alpha] = InverseKinematics(\alpha, EE)$

begin

  if $\underline{not(isPossible(solutions))}$ **then**

    if $\underline{alpha > \pi}$ **then**

      $step = -\frac{step}{2}$;

      $alpha = \pi$;

    else if $\underline{alpha < -\pi}$ **then**

      $step = -\frac{step}{2}$;

      $alpha = -\pi$;

    else if $\underline{step == \frac{1}{32} || step == -\frac{1}{32}}$ **then**

      $disp('FAILED')$;

    $[solutions, \alpha] = InverseKinematics(\alpha + step * \pi/6, EE)$;

It was considered to stop at a variation lower than $\frac{\pi}{96}$ just for precaution, because there was always the possibility of the program stay stucked on an infinite loop. However, it never happened. Although this method isn't the most sophisticated, it recursively get through α values until it finds the results and, by using the same data to test the program as in Section 4.1.1, the final results with this momentously solution were the following:

Algorithm	Alpha known?	Number of tests	Success Tests	Efficiency
Inverse Kinematics	No	2187	2187	100%

Table 4.3: Tests on IK with alpha unknown

By analysing Table 4.3 it is possible to verify that the system was always capable to obtain the final solution with a 100% solution as well. It is important to remember that this solution was once more acquired with the utilization of data tests created by the forward kinematics. This means that the solution by this data was always possible to find, which can not be true in real cases since it may not have a solution for the end-effector presented.

4.2 Test with Trac_ik

In order to run Trac_ik, the program needed a few input values as it can be shown in Listing 4.1:

```
    def get_ik(self, qinit,
               x, y, z,
               rx, ry, rz, rw,
               bx=1e-5, by=1e-5, bz=1e-5,
               brx=1e-3, bry=1e-3, brz=1e-3):
        """
        Do the \ac{IK} call.

        :param list of float qinit: Initial status of the joints as seed.
        :param float x: X coordinates in base_frame.
        :param float y: Y coordinates in base_frame.
        :param float z: Z coordinates in base_frame.
        :param float rx: X quaternion coordinate.
        :param float ry: Y quaternion coordinate.
        :param float rz: Z quaternion coordinate.
        :param float rw: W quaternion coordinate.
        :param float bx: X allowed bound.
        :param float by: Y allowed bound.
        :param float bz: Z allowed bound.
        :param float brx: rotation over X allowed bound.
        :param float bry: rotation over Y allowed bound.
        :param float brz: rotation over Z allowed bound.
```

Listing 4.1: Trac_ik input values

Besides the generation of the positions and orientations for the end-effector (that were used as the previous tests, defined by Table 4.1), the program also required the boundaries for both position (bx,by,bz) and orientation (brx,bry,brz as euler angles) of the end effector. There were made 3 different tests considering these two factors and the results were the following:

Algorithm	Bound Position (mm)	Bound Orientation (rad)	Number of tests	Success Tests	Efficiency
Trac_ik	0.001	0.1	2187	225	10.29%
Trac_ik	0.001	0.2	2187	316	14.45%
Trac_ik	0.001	9999.0	2187	2180	99.73%

Table 4.4: Tests with trac_ik

The inspection of this table allowed to conclude that the program is very efficient achieving a determined position, but only for positions which must not have a specific orientation.

The tests made in Section 4.2 were also done with higher bx, by and bz for the same positions, however, the results were similar, which allowed assuming that it would not be a predominant aspect in this case.

4.3 Example of IK solutions on an obstacle scenario

Obstacles have always been a difficult problem to handle in Robotics. The trac_ik's method will not be a good inverse kinematics to avoid obstacles since it only uses Newton's approaches based on calculations that would never be manipulable in order to avoid obstacles in the middle. However, due to *Vizzy*'s characteristics, it may perform the analytical method of the Inverse Kinematics in order to achieve it. This will only be possible because the only thing that the robot will need to know is which α angles are available to find the elbow's position without existing collapsing.

In order to show this fact, a specific example was created. The robot was positioned on the origin, and it had to catch a ball with orientation and position equal to:

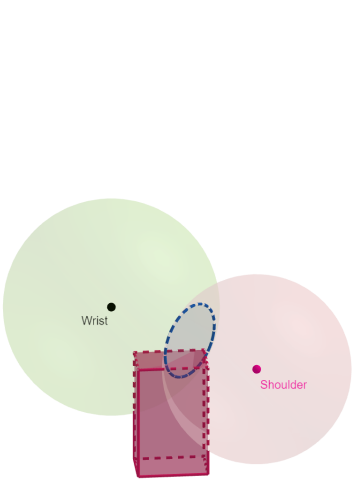
$$T_{EE} = \begin{bmatrix} -0.4819 & -0.7298 & -0.4850 & -277.6091 \\ 0.8070 & -0.1540 & -0.5702 & 489.3845 \\ 0.3414 & -0.6661 & 0.6631 & 272.3578 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

As it was demonstrated in chapter section 3.2.1.D, the position of the wrist is already known and with coordinates:

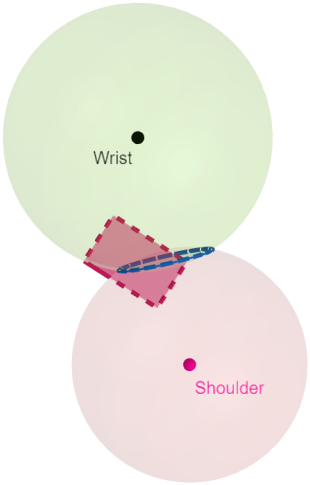
$$W = \begin{bmatrix} -229.4208 \\ 408.6868 \\ 238.2145 \end{bmatrix} \tag{4.2}$$

Between the robot and the end effector, there was placed an obstacle (for example, representing a bedside table), and it was placed with its upper base at z=200 and lower base at z=0 as it can be seen

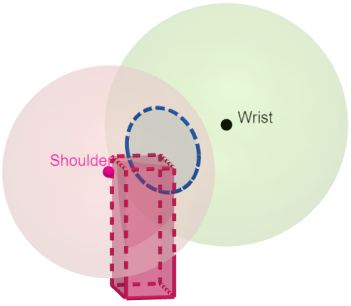
in the following images:



(a) Side View



(b) X - Y View in the world reference frame.



(c) Another Perspective

Figure 4.1: Example for an obstacle

For this example, the Inverse Kinematics was calculated through both algorithms, which allowed to obtain the following angles:

Algorithm	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8
Vizzy's Algorithm	64.92	38.97	-23.34	40	-78.31	-17.96	-21.98
$Trac\_ik$	75	7.5	50	40	0	0	0

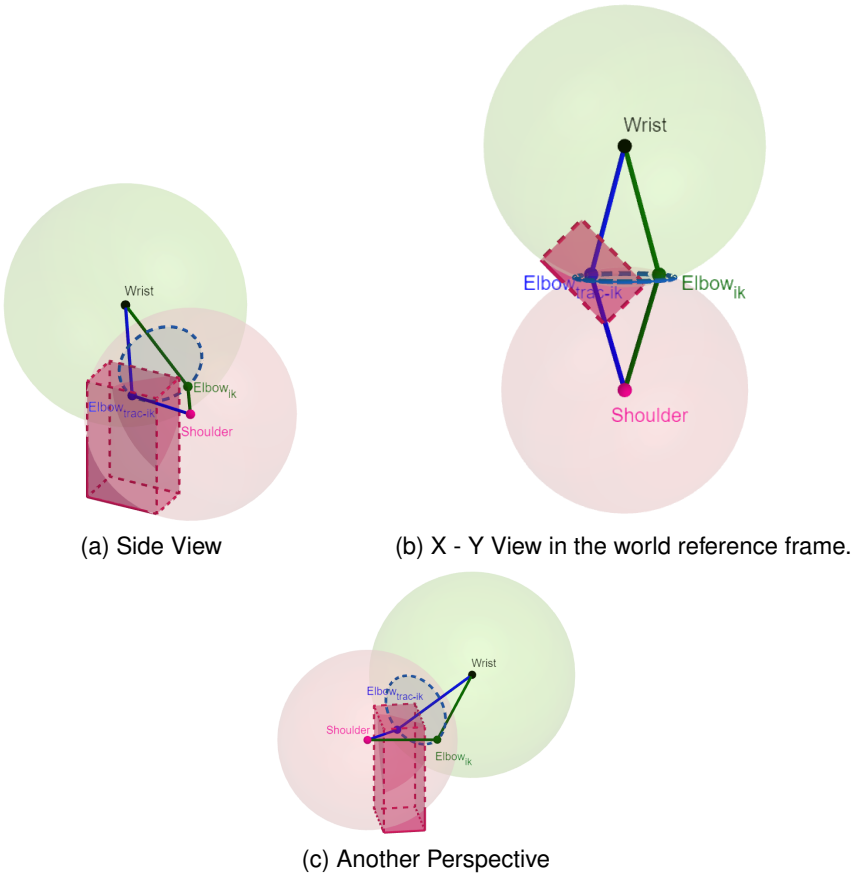
Table 4.5: Angles obtained with both Algorithms in Degrees

Therefore, were obtained 2 different positions for the elbow:

$$E_{trac\_ik} = \begin{bmatrix} -139.6220 \\ 258.3066 \\ 174.5884 \end{bmatrix} \tag{4.3}$$

$$E_{alg} = \begin{bmatrix} -76.0892 \\ 322.6969 \\ 176.3922 \end{bmatrix} \tag{4.4}$$

Thus, the final results are the following:



(a) Side View

(b) X - Y View in the world reference frame.

(c) Another Perspective

Figure 4.2: Example for an obstacle

By analysing the fig. 4.2, we can understand that trac_ik could not find a feasible position for the elbow because it collides with the obstacle, while the analytical approach created is able to solve it. It is also important to clarify that the algorithm created for the *Vizzy*'s IK was able to find a solution, and it always will as long as it exists.

4.4 Impact of elbow redudancy α on IK sucess

After comparing the Analytical method with trac_ik, the main difference is that the one proposed in this master thesis could achieve the target with the desired orientation. However, to obtain the Inverse Kinematics, a very extensive test was needed since there were times that a wide range of values for α was used and failed until it found a suitable solution. Thus, the purpose of this section is to study the impact of the choice of α on the efficiency of the IK.

In order to test the efficiency of $\alpha's$ values, an experience was made for attainable end-effectors. Then, there were chosen 128 targets, using Forward Kinematics and, therefore, the Inverse Kinematics

was computed with α assuming all integer values, starting in 0º until 360º. The results of this experience were compressed in the graphics represented in Figure 4.3 and Figure 4.4.

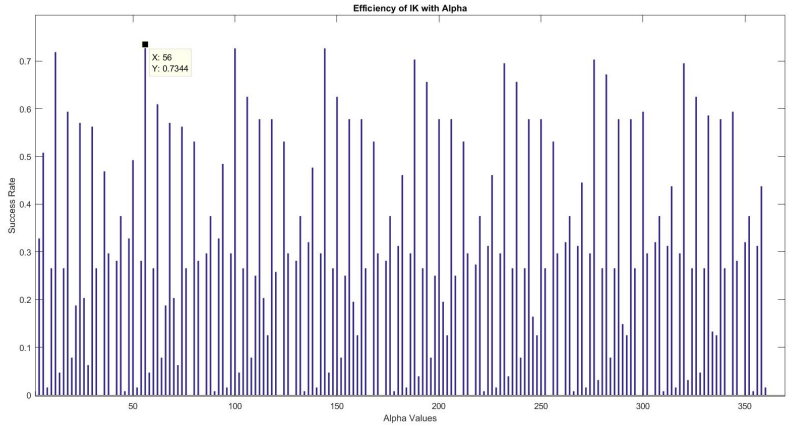


Figure 4.3: Efficiency of IK with Alpha (in degrees)

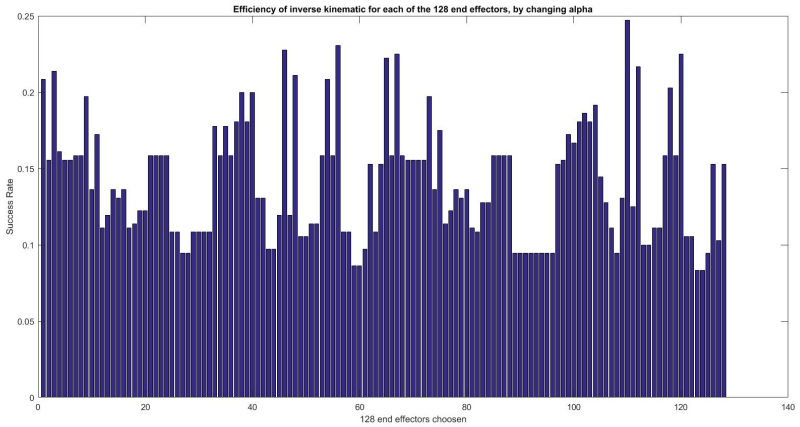


Figure 4.4: Efficiency of the Inverse Kinematics for each of the 128 end-effectors, by changing alpha.

By the analysis of the fig. 4.3, it is possible to understand that there was an angle that performed with the most efficiency and that was $\alpha = 56$, with an efficiency of 73.44%. Besides that, it is also possible to notice that there is a kind of a periodicity in the results, with the period of $\alpha = 34$. Important to emphasize that, only were used integer values for α, however, by fig. 4.4, it is possible to understand that, even if α values are only integer, the Inverse Kinematics was able to at least obtain a solution for all of them.

The results in this section are very important for 2 two reasons:

1. At the next chapter (section 4.5), there is only going to be used the α with higher efficiency ($\alpha = 56$). This will result in much faster tests with an efficiency closer to 75%;

2. For future work, there will be a chance to verify if this periodicity can lead to a combination of α that

allows a 100% chance of success since, at this point, we had a 100%, but only when examining all α available.

4.5 Tests with the base of the robot

At section 3.2.2, it was concluded that in the specific context of the Inverse Kinematics created, the position of the shoulder S_2 and the platform's base and orientation are directly connected for a specific end effector. Having in account these considerations, the purpose right now was to identify which discretization could be applied in order to find a suitable inverse kinematics. Thus, the following tests were responsible for identifying the response of the system to small changes in the base's positions/angles:

1. Add a small shift of 0.5cm to x or/and y coordinates of the position of the base.
2. Add a small angle to the orientation of the base (0.5º on the xOy orientation, fig. 4.5).

These tests were applied as represented in fig. 4.5. These results were surprising since these tests were made for more than 2000 end-effectors, and none of them was successful. This led to the fact that a slight variance in the initial positions of the robot could mean that the robot would not be able to find a suitable solution for the Inverse Kinematics.

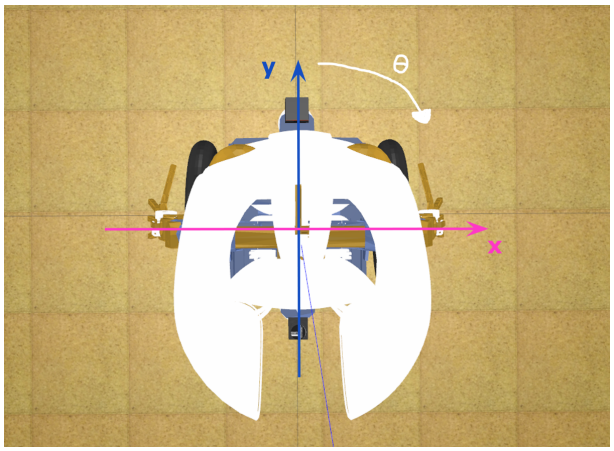


Figure 4.5: Top view from *Vizzy*

In contemplation of solving this problem, a different approach was implemented. As it was referenced on [32], a hybrid approach was developed by creating a Data-Driven approach as well. The search of the Inverse Kinematics would be computed by comparing the subtraction of all those matrices created through the pre-computed values from the end-effector positions by L1-norm.

In order to obtain those pre-learned postures, the angles responsible for the forward kinematics ($\theta_2 - \theta_8$ and $angle\_base$) were discretized, by considering the base at the origin. Therefore, that was

assumed that $\theta_2 - \theta_8$ would assume the values defined in table 4.1 (where were assumed 3 values for each angle, one closer to the maximum value, another closer to the minimum and the last one closer to the mean value) and the $angle\_base$ would be in the range of [0, 360º], assuming all integer values in that interval. This created 790507 different pre-computed values to test.

To compute and verify the results obtained through this strategy, 2 different variables were considered. The first one would be the number of transformations used from the pre-computed values. There were tests that considered 100% of the values, others 20% and even 5% of the table as well. The second one would be the number of inverse kinematics tested per experiment through the analytical method created. Those 2 strategies were developed and the results are represented in table 4.6, where "Percentage of the Table" shows the percentage of pre-computed values used from those 790507 different poses, the "No. IK Calculated" represents the number of Analytical Inverse Kinematics calculated using the best 1/5/10 minimum values of the L1-Norm and the "Percentage of IK Successes" exposes the percentage of those Inverse Kinematics that effectively found a solution. Regarding the L1-Norm, the efficiency of the algorithm was studied through the Maximum, Mean and Standard-Deviation of the minimum values of the L1-Norm, as it can be seen in table 4.6.

Test No.	Percentage of the Table	No. IK Calculated	Percentage of IK Successes	Time Per Experiment	Maximum $\min\|X\|_1$	Mean $\min\|X\|_1$	Std. Deviation $\min\|X\|_1$
1	100%	1	7.1%	92 s	0.0024	$1.415 \cdot 10^{-5}$	$6.8827 \cdot 10^{-5}$
2	100%	5	21.6%	180 s	0.0024	$1.25 \cdot 10^{-5}$	$6.7868 \cdot 10^{-5}$
3	100%	10	31.4%	186 s	0.0024	$1.145 \cdot 10^{-5}$	$6.7053 \cdot 10^{-5}$
4	20%	1	6.8%	84 s	0.0060	$1.6625 \cdot 10^{-4}$	$3.0423 \cdot 10^{-4}$
5	20%	5	21.5%	246 s	0.0060	$1.433 \cdot 10^{-4}$	$3.0229 \cdot 10^{-4}$
6	20%	10	31.6%	144 s	0.0060	$1.2865 \cdot 10^{-4}$	$3.0136 \cdot 10^{-4}$
7	5%	1	7.3%	86 s	0.0191	$7.9685 \cdot 10^{-4}$	0.0014
8	5%	5	22.7%	289 s	0.0191	$6.685 \cdot 10^{-4}$	0.0013
9	5%	10	32%	303 s	0.0191	$6.058 \cdot 10^{-4}$	0.0013

Table 4.6: Tests with the Base

From the analyses of the table is possible to conclude that the number of Inverse Kinematics that is performed for each experiment is directly proportional to the success rate of the program through Inverse Kinematics. At the same time, it is possible to deduct from the cases where there are more Inverse Kinematics being calculated, that both mean and standard deviation values from L1-Norm tend to be slightly smaller since the distances between the rotation matrices are lower, allowing to obtain better results. Regarding the Maximum minimum value of the L1-Norm, there is no influence on the results.

On the other hand, the number of transformations used from the pre-computed values allowed a significant improvement in the program. In particular, it is visible that the standard deviation and the Mean $\min\|X\|_1$ start with an order of magnitude of 10^{-5} for 100% of the pre-computed values, increases for 10^{-4} for 20% and the std. deviation still goes to 10^{-3} for 5%, which shows that the efficiency of the program decreases with the decreasing number of pre-tables values used. Regarding the Maximum L1-

Norm's minimum value, the conclusion obtained is exactly the same, which means that by using more pre-computed values, the associated errors tend to be smaller. On the other side, it is also possible to conclude that the percentage of Inverse Kinematics that was successfully computed does not change significantly by using more or less values from the pre-computed acquired before.

A critical analysis from table 4.6, that can be deducted from this experiment is that these nine tests allowed to verify that the algorithm computed the Inverse Kinematics efficiently, consistently achieving the primary goal. This efficiency was obtained because it only considered pre-computed values 5 centimeters ahead or below the target. The choice of those 50 mm was considered by studying the structure of *Vizzy* and understanding that with this difference, it would always be able to attain the end-effector. Besides that, it is clear that the main limitation of this approach is the time that the algorithm takes to find the solution. One aspect that clearly influenced the results was the implementation in Matlab, which a C++ implementation could replace, probably resulting in faster results.

5

Conclusions and Future Work

Contents

5.1 Conclusions

This master thesis proposes an Inverse Kinematics Method that obtains an accurate solution considering both position and orientation, which would make the robot *Vizzy* achieve its final goal, the target. This was achieved by separating the problem into two separate parts. The first one was responsible for the Inverse Kinematics from the shoulder until the hand, by considering the base fixed, and the second one, by including the base and finding the best position to place the base to compute the Inverse Kinematics.

Regarding the Analytical Method created for the Inverse Kinematics for the *Vizzy*'s arms, it demonstrates 100% efficiency for all positions and orientations achievable by *Vizzy*, however always dependent on the acquisition of the redundancy α. Therefore, for a system where α was known as an input of the system, it would be easier and faster to attain. However, if the system didn't provide α, it was possible through an algorithm that discretized the angle on the space, allowing the angle to be acquired. Thus, it allowed verifying that the Analytical Method could perform better than many control Based-Methods available, such as Orocos Kinematics and Dynamics Library (KDL) and trac_ik. The last one, trac_ik, revealed a very efficient algorithm in terms of achieving the goal for any position given as input, but, on the other hand, every time a specific orientation was requested, the algorithm presented very low efficiency.

About the approach used to perform the Inverse Kinematics for all *Vizzy*'s body, it required not only the understanding of how far the hand of the robot could go but also where the base could be placed so that *Vizzy* could perform the Inverse Kinematics due to its limitations. Both problems were solved by using a hybrid solution, which combined not only the Analytical Method developed but also a Data-Driven solution by the utilization of pre-computed values. This solution, with the usage of those values, was able to perform the inverse kinematics with values of the $angle\_base$ that already are known and are working solutions, which improves the efficiency of the system.

Another aspect that is important to emphasize is the lack of guarantees for the existence of the inverse kinematics for a specific placement of the platform due to its discontinuity. This is always a limitation since the robot never has the chance to understand if it did not find the solution because of the redundancies that were not being dealt with in the proper way (such as α) or if the solution did not even exist. Despite this point, *Vizzy* performed with high efficiency the Analytical Method proposed and found out always a solution that solved the problem.

5.2 Future Work

This approach to the Inverse Kinematics never is a finished implementation since the evolution of the technology always allows the development of new ways to implement and improve this mechanism. There are many contributions that this Master Thesis may have in future work, such as the avoidance of

obstacles and the future algorithms to plan the movement of the robot *Vizzy*.

5.2.1 Calculation of α through image processing

One approach that can be implemented in the future is the analysis of the circle generated through the intersection of the two spheres (section 3.2.1.E). At this point, in order to implement inverse kinematics for all *Vizzy*'s body, it considered the α that has the highest efficiency. At the same, it would never allow the avoidance of obstacles in this way. However, by image processing, the robot could be able to verify if those positions of the circle were available and therefore identify α (which positions were possible to the elbow of the robot attain), as it is shown in section 4.3.

5.2.2 Inverse Kinematics through threads

For future work, dealing with α through multiple threads may be a solution. This solution will allow splitting the problems into simpler problems. One of those solutions is the division of the arm into two parts: A thread that would be responsible for calculating the angles for the shoulder, while another thread would be the one that would calculate the following angles. This only would be possible because the position and orientation of the elbow are already known.

The usage of those threads would also simplify the algorithm 4.1, since it could run all of them and return only the final solution. Therefore, this would be an approach that would considerably low the time processing.

5.2.3 Planning algorithms for *Vizzy*'s movement

At this point, *Vizzy* only computes an Analytical Method for the Inverse Kinematics, however, the program only determines the position where the base should or not be placed. This will reacquire the study of planning algorithms in order to efficiently attain that position. On the other hand, since *Vizzy* may be a robot to be in people's houses helping people, it is vital that *Vizzy* can avoid obstacles in order not to collide with furniture that is everywhere in a house.

5.2.4 Usage of θ_0 and θ_1

In this master thesis, there was not possible to define an inverse kinematics that could assemble the Analytical Analysis with the usage of θ_0 and θ_1. This assumption implied that a lot of possible solutions would have to be disconsidered since the problem was too complex.

5.2.5 Usage of quaternion difference instead of L1-Norm

Regarding the placement of the base, the L1-norm was calculated not only to choose the closest pre-computed values to perform the IK, but also to verify the accuracy of the results. However, there are other formulas that potentially may perform with better results, such as L2-Norm [35] and the usage of the Distance between rotations, θ, [36].

.

Bibliography

[1] A. Aristidou and J. Lasenby, "Inverse kinematics: a review of existing techniques and introduction of a new fast iterative solver," 2009.

[2] T. Yoshikawa, "Manipulability of robotic mechanisms," *The international journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.

[3] C. Welman, "Inverse kinematics and geometric constraints for articulated figure manipulation," Ph.D. dissertation, Theses (School of Computing Science)/Simon Fraser University, 1993.

[4] V. Pilania and K. Gupta, "A hierarchical and adaptive mobile manipulator planner," in *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014, pp. 45–51.

[5] ——, "Mobile manipulator planning under uncertainty in unknown environments," *The International Journal of Robotics Research*, vol. 37, no. 2-3, pp. 316–339, 2018.

[6] A. Colome, "Smooth inverse kinematics algorithms for serial redundant robots," Ph.D. dissertation, Master Thesis. Barcelona: Institute de Roboticai Informatica Industrial (IRI), 2011.

[7] J. Wang, Y. Li, and X. Zhao, "Inverse kinematics and control of a 7-dof redundant manipulator based on the closed-loop algorithm," *International Journal of Advanced Robotic Systems*, vol. 7, no. 4, p. 37, 2010.

[8] S. Sharma and C. Scheurer, "Generalized unified closed form inverse kinematics for mobile manipulators with reusable redundancy parameters," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 58189. American Society of Mechanical Engineers, 2017, p. V05BT08A078.

[9] M. Shimizu, H. Kakuya, W.-K. Yoon, K. Kitagaki, and K. Kosuge, "Analytical inverse kinematic computation for 7-dof redundant manipulators with joint limits and its application to redundancy resolution," *IEEE Transactions on robotics*, vol. 24, no. 5, pp. 1131–1142, 2008.

[10] H. Su, W. Qi, Y. Hu, H. R. Karimi, G. Ferrigno, and E. De Momi, "An incremental learning framework for human-like redundancy optimization of anthropomorphic manipulators," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, pp. 1864–1872, 2020.

[11] M. Pfurner, "Closed form inverse kinematics solution for a redundant anthropomorphic robot arm," *Computer Aided Geometric Design*, vol. 47, pp. 163–171, 2016.

[12] P. Dahm and F. Joublin, *Closed form solution for the inverse kinematics of a redundant robot arm*. Ruhr-Univ., Inst. für Neuroinformatik, 1997.

[13] Y. Wang, "Closed-form inverse kinematic solution for anthropomorphic motion in redundant robot arms," Arizona State University, Tech. Rep., 2013.

[14] M. Bugday and M. Karali, "Design optimization of industrial robot arm to minimize redundant weight," *Engineering Science and Technology, an International Journal*, vol. 22, no. 1, pp. 346–352, 2019.

[15] R. Raja, A. Dutta, and B. Dasgupta, "Learning framework for inverse kinematics of a highly redundant mobile manipulator," *Robotics and Autonomous Systems*, vol. 120, p. 103245, 2019.

[16] T. Asfour and R. Dillmann, "Human-like motion of a humanoid robot arm based on a closed-form solution of the inverse kinematics problem," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, vol. 2. IEEE, 2003, pp. 1407–1412.

[17] X. Tian, Q. Xu, and Q. Zhan, "An analytical inverse kinematics solution with joint limits avoidance of 7-dof anthropomorphic manipulators without offset," *Journal of the Franklin Institute*, vol. 358, no. 2, pp. 1252–1272, 2021.

[18] "ROS robot operating system," https://www.ros.org//, accessed: 2022-03-07.

[19] "TRAC-IK kinematics solver," https://moveit.picknik.ai/galactic/doc/examples/trac_ik/trac_ik_tutorial.html, accessed: 2022-03-07.

[20] "IKFast kinematics solver," https://moveit.picknik.ai/galactic/doc/examples/ikfast/ikfast_tutorial.html, accessed: 2022-03-07.

[21] P. Beeson and B. Ames, "Trac-ik: An open-source library for improved solving of generic inverse kinematics," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2015, pp. 928–935.

[22] "TRACK-IK trac-ik," https://bitbucket.org/traclabs/trac_ik/src/master/, accessed: 2022-03-07.

[23] "KDL orocos kinematics and dynamics," https://www.orocos.org/kdl.html, accessed: 2022-03-07.

[24] R. Diankov, "Automated construction of robotic manipulation programs," 2010.

[25] A. Aristidou, J. Lasenby, Y. Chrysanthou, and A. Shamir, "Inverse kinematics techniques in computer graphics: A survey," in *Computer graphics forum*, vol. 37, no. 6. Wiley Online Library, 2018, pp. 35–58.

[26] A. Makhal and A. K. Goins, "Reuleaux: Robot base placement by reachability analysis," in *2018 Second IEEE International Conference on Robotic Computing (IRC)*. IEEE, 2018, pp. 137–142.

[27] J. Xu, K. Harada, W. Wan, T. Ueshiba, and Y. Domae, "Planning an efficient and robust base sequence for a mobile manipulator performing multiple pick-and-place tasks," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 11 018–11 024.

[28] A. H. B. Boutchouang, A. Melingui, J. J.-B. M. Ahanda, O. Lakhal, F. B. Motto, and R. Merzouki, "Learning-based approach to inverse kinematics of wheeled mobile continuum manipulators," *IEEE-ASME TRANSACTIONS ON MECHATRONICS*, 2022.

[29] K. Harada, T. Tsuji, K. Kikuchi, K. Nagata, H. Onda, and Y. Kawai, "Base position planning for dual-arm mobile manipulators performing a sequence of pick-and-place tasks," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2015, pp. 194–201.

[30] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Efficient inverse kinematics computation based on reachability analysis," *International Journal of Humanoid Robotics*, vol. 9, no. 04, p. 1250035, 2012.

[31] S. Sharma, G. K. Kraetzschmar, C. Scheurer, and R. Bischoff, "Unified closed form inverse kinematics for the kuka youbot," in *ROBOTIK 2012; 7th German Conference on Robotics*. VDE, 2012, pp. 1–6.

[32] A. Volinski, Y. Zaidel, A. Shalumov, T. DeWolf, L. Supic, and E. E. Tsur, "Data-driven artificial and spiking neural networks for inverse kinematics in neurorobotics," *Patterns*, vol. 3, no. 1, p. 100391, 2022.

[33] P. Moreno, R. Nunes, R. Figueiredo, R. Ferreira, A. Bernardino, J. Santos-Victor, R. Beira, L. Vargas, D. Aragao, and M. Aragao, "Vizzy: A humanoid on wheels for assistive robotics," in *Robot 2015: Second Iberian Robotics Conference*. Springer, 2016, pp. 17–28.

[34] "Scapulothoracic Joint description," https://www.physio-pedia.com/Scapulothoracic_Joint, accessed: 2022-03-17.

[35] X. Li, Y. Pang, and Y. Yuan, "L1-norm-based 2dpca," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 40, no. 4, pp. 1170–1175, 2010.

[36] “THETA distance between rotations,” http://www.boris-belousov.net/2016/12/01/quat-dist/, accessed: 2022-03-20.