Universidade do Minho
Escola de Engenharia

Rita Maria Pereira Rodrigues

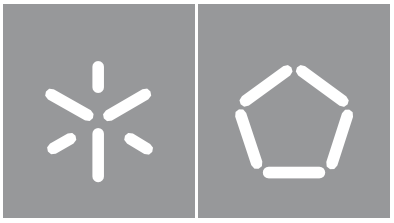
Motion Planning for Autonomous Sorting Tasks using the Collaborative Saywer Robot

setembro de 2022

Motion Planning for Autonomous Sorting Tasks using the Collaborative Sawyer Robot

Rita Rodrigues

UMinho | 2022



Universidade do Minho
Escola de Engenharia

Rita Maria Pereira Rodrigues

Motion Planning for Autonomous Sorting Tasks using the Collaborative Saywer Robot

Dissertação de Mestrado
Engenharia Eletrónica Industrial e Computadores
Controlo, Automação e Robótica

Trabalho efetuado sob a orientação do
Professora Doutora Estela Bicho

setembro de 2022

DIREITOS DE AUTOR E CONDIÇÕES DE UTILIZAÇÃO DO TRABALHO POR TERCEIROS

Acknowledgements

This dissertation could not have been completed without the help of several people who supported and helped me with this project.

First and foremost, I would like to thank my supervisor, Prof. Dr. Estela Bicho, for introducing me to this project and providing me with academic advice and support and availability throughout this project.

I would also like to thank Prof. Dr. Luís Louro and Prof. Dr. Sérgio Monteiro for all the advice and suggestions they gave me during our meetings.

A big thank you goes to Gianpaolo Gulleta for all the help you gave me. You were always available to answer my doubts and helped me solve several problems that came up. Thank you for your availability and support.

All this would not have been possible without my parents. Thank you for all your support and especially for always believing in me and giving me the opportunities to achieve this accomplishment.

I would also like to thank the rest of my family and friends for all the support during this journey.

Finally, I would like to thank my boyfriend for his emotional support, comfort, and understanding during this project. Thank you for always being there for me.

STATEMENT OF INTEGRITY

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

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

Resumo

Planeamento de Movimentos para tarefas de *sorting* usando o Robô Colaborativo Sawyer

Na década passada, pesquisa e desenvolvimento da área da robótica tem desenvolvido soluções para otimizar o processo de fabricação usando robôs colaborativos. Estes robôs são seguros e podem coexistir com os operadores humanos, trabalhando lado a lado em diferentes tarefas. Por exemplo, *Pick & place & pack* e *sorting*. No entanto, apesar de ser facilmente programáveis, estes robôs só fazem a mesma tarefa e não conseguem autonomamente adaptar a variações no espaço de trabalho. Considerando este problema associado com a ausência de autonomia nos robôs colaborativos, esta dissertação tem o objetivo de fazer com que o robô colaborativo sawyer consiga se programar e autonomamente gerar os seus movimentos, numa maneira adaptada á tarefa e ao espaço de trabalho.

Um dos objetivos principais é a geração autónoma de movimentos, livre de colisões, do robô Sawyer com caraterísticas cinemáticas similares aos movimentos feitos por humanos, em tarefas colaborativas como *Pick & Place & Sort*, em que o robô deve alcançar, agarrar, transportar e ordenar os diferentes tipos de objetos em simulação e num ambiente real. Para realizar este objetivo, um modelo baseado em caraterísticas dos movimentos dos membros superiores humanos, previamente desenvolvido por uma equipa de pesquisa do Mar Lab na universidade do Minho foi utilizado. Este modelo permite um planeamento de movimentos independente que se divide em dois subproblemas, a seleção da postura final e trajetória.

A validação deste modelo foi feita numa simulação onde o robô foi instruído a fazer a uma tarefa de *Pick & Place*. Os resultados mostram que o modelo de planeamento de movimentos implementado permite que o robô gere trajetórias similares às de humanos, evitando colisões com objetos no espaço do trabalho.

palavras-chave: robô colaborativo, planeamento de movimentos, *sorting*, manipuladores robóticos.

Abstract

Motion Planning for Autonomous Sorting using the Collaborative Sawyer Robot

In the past decade, research and development in the robotics area have developed solutions to optimize the fabrication processes using collaborative robots. These robots are safe robots that can coexist with the human operators, working side by side in different tasks, e.g. Pick & Place & Pack and Sorting. However, despite being easily programmable, these robots always perform the same task, they cannot autonomously adapt to variations in the workspace. Considering this problem associated with the absence of autonomy in collaborative robots, this dissertation aims to endow the collaborative robot Sawyer with the capacity to self-program and autonomously generate its movements, in an adapted way to the task and the workspace.

One of the main objectives is the autonomous generation of a collision-free movement of the robot with kinematic characteristics similar to the movements made by humans, in collaborative tasks such as Pick & Place & Sort, in which the robot should reach, grasp, transport, and sort different types of objects in a simulation and in a real environemnt. To achieve this, a model based on the movement characteristics of human upper limbs, previously developed by the research team at the MAR Lab at the university of Minho, was deployed. This model allows for an independent motion planning process which divides into two subproblems, selection of the final posture and trajectory.

The validation of the model was made in a simulation software where the robot was instructed to make a sorting pick & place task. The results show that the motion planning model implemented allows the robot to generate trajectories similar to movements made by humans, as well as avoiding collisions with objects in the workspace while realizing the given task.

keywords: collaborative robot, motion planning, sorting, robotic manipulators.

Contents

List of Figures

List of Tables

List of Abbreviations

CNS	Central Nervous System.
DOF	Degrees of Freedom.
FTPBD	Fault-Tolerant Programming by Demonstration.
HUMP	Human-Like Upper-Limb Motion Planner.
ISO	International Organization for Standardization.
MAR Lab	Mobile and Anthropomorphic Robotics Lab.
NJS	Normalized Jerk Score.
NMU	Number of Movement Units.
PBD	Programming by Demonstration.
PRM	Probabilistic Roadmap Method.
ROS	Robot Operating System.
RRT	Rapidly-Exploring Random Tree.
V-REP	Virtual Robot Experimental Platform.

Chapter 1: Introduction

In the past decades, the field of robotics focused on the development of new solutions that made it possible to meet the technological needs of people and improve their quality of life. Initially, robotic platforms were only used in assembly and manufacturing processes, performing repetitive, precise, and risky tasks for human operators. The implementation of this work methodology in industrial production lines contributes to the growth of various companies by allowing an increase in daily production, a reduction in labor costs, an increase in the quality of manufactured products, and a reduction in workplace accidents (Garcia et al. (2007); Schaal (2007)). According to Gladden (2019), there has been an evolution of human society featured by:

(i) Society 1.0 - A hunter-gatherer society;

(ii) Society 2.0 - Agrarian society;

(iii) Society 3.0 - Industrial Society;

(iv) Society 4.0 - Information Society.

We are in Society 4.0, described by Fukuyama (2018) as "an information society that realized increasing added value by connecting intangible assets as information networks". Gladden (2019) promotes Society 5.0, which "aims to create a cyber-physical society in which citizens' daily lives will be enhanced through increasingly close collaboration with artificially intelligent systems." With the technological development focused on artificial intelligence, the first robotic mechanisms capable of sharing the workspace and/or interacting with human operators were developed and commercialized in the early XXI century (Schaal (2007)). The scientific community encouraged the creation of new solutions outside the industrial sector, related to domestic activities, space exploration, construction, search and rescue, and general assistance in daily life situations. Furthermore, there are research groups specifically focused on the development and integration of social robots into various everyday tasks. These systems are used as partners, colleagues, and human assistants in countless tasks that address assistance and health needs, such as therapy sessions for neurological disorders, playful and educational activities, physical therapy, and service for elderly people with limited mobility (Fong et al. (2003)).

In addition to the systems already mentioned, several companies have begun to develop and market collaborative robots for industrial production lines. These robots are safe robots that can coexist with human operators by working side by side on different tasks, e.g., pick and place and pack and Bin Picking, or work together on the same task, e.g., assembly tasks. However, although these robots are easily programmable, they always perform the same task; they cannot autonomously adapt to changes in the workspace. Sawyer (Rethink Robots), YuMi (ABB), and UR10(Universal Robots) are some collaborative robot platforms currently available in the market.

Considering the problem associated with the lack of autonomy of collaborative robots, this dissertation aimed to deploy the planner HUMP - Human-like Upper Motion Planner - in the collaborative robot Sawyer in order to endow it with the ability to program itself to sort and place different objects on a shelf while avoiding obstacles in the workspace. Moreover, in section 1.1, the main motivations for the realization of this dissertation are presented, highlighting the motion planning projects previously developed in the Mobile and Anthropomorphic Robotics Lab (MAR Lab) of the University of Minho. Finally, the 1.2 section presents the structure of this dissertation.

1.1 Motivation and Objectives

Nowadays, industrial companies developing collaborative robotic platforms usually rely on easily programmable systems. Therefore, the robot's movements are defined by the human operator's demonstration of the movement. In the specific case of the Sawyer robot, launched in March 2016, the trajectories executed by the robot during a given task are defined by the Intera 5.2 software platform provided by the manufacturer - Rethink Robotics.

In the past decade, research and development in the robotics area have developed solutions to optimize the fabrication processes using collaborative robots. These robots are safe robots that can coexist with the human operators, working side by side in different tasks such as Pick & Place & Pack and Bin Picking, or working in collaboration on the same task such as Assembling tasks. However, despite being easily programmable, these robots always perform the same task, they do not have the capacity to autonomously adapt to variations in the workspace. This is a limitation that seriously restricts the collaborative potential of these robots in inherently dynamic tasks with unpredictability such as the changing the objects' position, obstacles can show up in the workspace that need to be avoided and also the physical characteristics of the human operators can be different.

Considering this problem associated with the absence of autonomy in collaborative robots, this disser-

tation aims to endow the collaborative robot Sawyer with the capacity to self-program and autonomously generate its movements, in an adapted way to the task and the workspace. One advantage, beyond its flexibility, is the reduction of down-time. Since we have the objective that the robot realizes tasks in direct collaboration with its humans' colleagues, an additional and important requirement for a more efficient human-robot interaction is that the generated movements be similar to the movements made by the human operators. Human-like movements allows the human coworkers to more easily understand the robot actions and predict the intentions, and hence to start earlier to make the complimentary action e.g. in a bin-sorting task, the prediction of what object the robot is going to grasp may be used by the human coworker to immediately place the corresponding container. Another important requirement is obstacle avoidance in order to avoid collisions with the human coworkers, with objects present in the workspace and with the robot itself.

One of the main objectives was the autonomous generation of a collision-free movement of the collaborative robot sawyer with kinematic characteristics similar to the movements made by humans, in collaborative tasks such as pick & place, as well as sort different types of objects and implement it in the collaborative robot Sawyer. This involved an intensive study of the functionalities and characteristics of the Sawyer robot (Chapter 4), as well as a study of the motion planning problem and its various applications which can be seen in chapter 2. There was also be a study of the ROS (Robot Operating System) functionalities and of the HUMP library (Chapter 5).

1.2 Dissertation Structure

This dissertation is divided in seven chapters, organized by the following:

- **Chapter 1** will present an introduction, the motivations and goals for this dissertation.
- **Chapter 2** will present a literature review of the motion planning methods in robotics manipulators.
- **Chapter 3** will present the main characteristics of human upper limbs movements.
- **Chapter 4** will present the basic specifications of the Sawyer collaborative robot, used in the implementation and validation of the project.
- **Chapter 5** will present the proposed motion planning method, making mention the movement classification system and planning strategy used.

- **Chapter 6** will present the scenario used to validate the motion planner as well as the obtained results.
- **Chapter 7** will present the conclusion of this dissertation.

Chapter 2: State of the Art

This chapter presents the literature review on motion planning in robotic manipulators. In section 2.1 a small introduction is given, in section 2.2 it is presented what a collaborative robot is and its importance in the industry, in section 2.3 some concepts related to motion planning are presented highlighting the different classifications. In section 2.4, different methods for solving motion planning problems are presented. Finally, section 2.5 presents some methods for bin sorting and bin sorting for anthropomorphic robots.

2.1 Introduction

The motion planning problem for a robotic manipulator is to find a trajectory for a point that represents the configuration of the manipulator in d-dimensional configuration space. An important concept in motion planning is configuration space - C-space. Each point in C-space corresponds to a unique configuration - q - of the manipulator and each configuration of the robot can be represented as a point in C-space. A feasible path is entirely in free C-space - C_{free} - which consists of all configurations where the robotic manipulator neither collides with obstacles in its path nor exceeds its joints limits (Lynch and Park (2017)). The Probabilistic Roadmap (PRM) and the Rapidly-Exploring Random Tree RRT may be the most commonly used sampling-based approaches (Elbanhawi and Simic (2014)).

2.2 Collaborative Robots

With the advent of Industry 4.0, collaborative robots have dominated the production and manufacturing industry as they have made significant advances over the past few decades. When one compares collaborative robots to industrial robots, one notices an increase in productivity, flexibility, versatility, and safety.

Industrial robots are defined as programmable, self-controlled devices consisting of electronic, electrical, or mechanical robots that perform a series of complex actions. Physically, they are large, heavy bodies used for tasks that can be difficult and potentially dangerous for human operators, such as carrying heavy loads around a factory. Typically, they are isolated from human operators and have their own workspace (Sherwani et al. (2020)). In contrast, collaborative robots are designed to perform tasks alongside human

operators and usually share the same workspace. In the figure 2.1, we can see that the collaborative workspace must be designed in such a way that the operator can perform their respective tasks while being completely safe from any safety hazards. These collaborative robots are much lighter devices, made of lightweight, high-strength metals. To best mimic the human-like movements of humans, they use intrinsically flexible actuators. The figure 2.2 shows the main features of a collaborative robot. Due to its characteristics, although it offers better mobility due to its lightweight, it also offers great flexibility, as a single robot can perform a variety of tasks and thus can be used in a wide range of industries. In addition, they are easily programmable and have large computing capacities that allow them to work safely and efficiently alongside human operators (Mihelj et al. (2019)).

Even though these robots are extremely useful in industry, they cannot work completely on their own. To have a highly efficient production line, the strengths of robots and humans must be combined. Robots have the advantage of repeatability, they have higher accuracy and endurance, so they can carry heavy loads without effort and save human operators from health problems in the future. But human operators have cognitive abilities that respond to unpredictable situations such as defective components or changing necessary process parameters. For this reason, there are collaborative robots that share the workspace with human operators to perform tasks of varying complexity. (Sherwani et al, 2020)

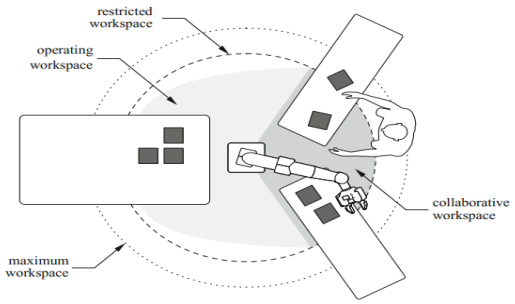


Figure 2.1: Workspace of a collaborative robot. [1]

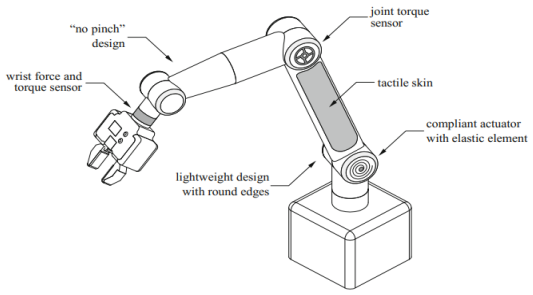


Figure 2.2: Characteristics of a collaborative robot. [2]

Some of the most well-known collaborative robots are Sawyer and Baxter from Rethink Robots (Figure 2.3), YuMi from ABB, LBR iiwa from KUKA (Figure 2.4), UR10 from Universal Robots (Figure 2.5), and Cr35-iA from FANUC, designed for a wide range of automated processes such as welding, packaging, quality control, assembly or logistics.

Unlike traditional robots, collaborative robotic systems are designed to work and interact with human operators by collecting a set of specifications that reduce accidents during working hours. Equipped with force limitation and potency, these robots are programmed to quantify the forces applied to the joints.

[2]Image taken from Mihelj et al. (2019)
[2]Image taken from Mihelj et al. (2019)
[3]Images taken from Google Images.

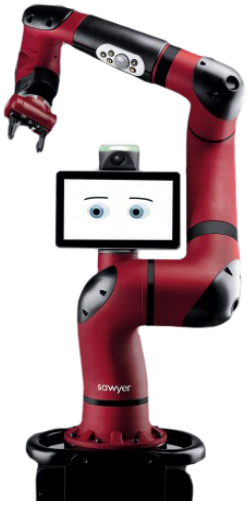


Figure 2.3: Sawyer Robot from Rethink Robots.

Figure 2.4: LBR iiwa from KUKA.

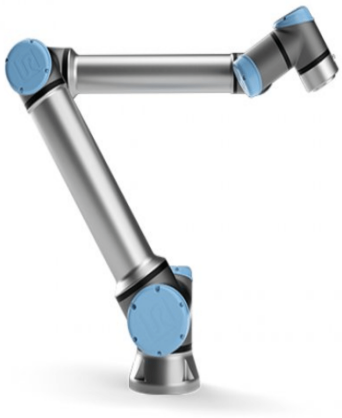


Figure 2.5: UR10 from Universal Robots. [3]

This feature allows the detection of abnormal forces generated by physical contact with objects or people, activating some mechanisms that make the system stop automatically and move in the opposite direction.

Physically, the collaborative robots present rounded surfaces, necessary to:

(i) Dissipate forces after impacts with large surfaces;

(ii) Hide the joints motors;

(iii) Guarantee the physical integrity of the human operators and the workspace.

In this context, to ensure the necessary security levels during the execution of the interaction tasks, the International Organization for Standardization (ISO) has defined a set of requirements for the collaborative systems. For these mechanisms, the international standard ISO 10218, applicable to all industrial robot

platforms, and the technical specifications ISO /TS 15066[4] (Bogue (2017)), which are defined for collaborative robots only, specifying the maximum values of potency, force, and velocity. These mechanisms can also be delineated by safety barriers applied to the manipulation of toxic or hazardous product processes.

The already mentioned characteristics of collaborative robots, combined with their flexibility and adaptability to different tasks, enable the implementation of powerful automation. Considering the current market trends, the commercialization of these robotic platforms is expected to grow exponentially. According to the International Federation of Robotics, global robot installations have decreased by 12% to 373,240 units in 2019[5], yet the number of industrial robots deployed is significantly higher compared to 2009 when only 60000 units were installed in total. According to these statistics, there are five major markets for industrial robots: China, Japan, the United States, the Republic of Korea, and Germany. These countries account for 73% of all global robot installations. We can also note that the automotive industry is the main consumer of industrial robots, followed by the electrical/electronics industry, the metal and machinery industry, the plastics and chemical industry, the food industry, and others.

2.3 The Motion Planning Problem

Path and trajectory are often used as synonyms, but this is incorrect. A **path** is a purely geometric description of motion, giving the location and orientation of points in joint space or Cartesian space. Meanwhile, a **trajectory** is a path on which a time law is specified, which can be given in terms of velocities and/or accelerations at each point (Silva (2011)).

Human-like motion planning methods can be classified according to primary features that provide general insights into concepts, ideas, and decisions behind the genesis of a motion. A primary classification is concerned with how knowledge about the robot's workspace is used to plan a possible path. A technique is called **global** if knowledge about the entire workspace is processed offline before a motion is executed. Conversely, **local** methods plan an online movement considering only the sections of the workspace that are of interest. Although the difference between these two methods is not always clear, they concern the implementation of collision avoidance strategies. A second classification focuses on the variables used within the motion planning process. Some methods use only kinematic variables such as position, velocity, and acceleration, while other methods also use dynamic variables such as forces and torques. A proper selection of these variables defines the state space of a robot and can be determined by the expected interaction with the environment or with a human operator. Regularities of human upper

[4]The international norms ISO 10218 and the technical specifications ISO /TS 15066, can be found in https://www.iso.org/.
[5]study available in https://ifr.org/.

limb motion can be replicated at different levels, depending on the space in which the robot operates. Some applications may limit the dimensionality of this space, which consequently affects the complexity of a planning algorithm. Considering these reasons, a third classification focuses on the space where the generation of a movement takes place since it can be the operation space or the joint space of the manipulation (Gulletta et al. (2020); Siciliano et al. (2009)).

Mathematically, the motion planning problem consists of determining the control inputs $u(t)$ that yield a trajectory $q(t)$ that avoids obstacles, brings the system to the desired goal state, and finally optimizes an objective function (Choset et al. (2005)). According to Hwang and Ahuja (1992), motion planning can be classified as either **gross-motion planning** or **fine-motion planning**. Gross-motion planning deals with the problems involving a space larger than the size of the objects and the position error of the robot. Fine-motion planning, on the other hand, deals with the problem of moving objects in confined spaces where the required motion accuracy exceeds the positioning accuracy of the robot. The focus of this dissertation is on gross-motion planning. This problem can have various classifications mainly related to the following:

(i) The type of information available about the obstacles;

(ii) The existence of possible alterations in the robot's workspace or on the configurations of the objects;

(iii) The application of constraints during the motion planning.

Motion planning can be **static** or **dynamic**, depending on the available information about the obstacles. In a static problem, all information about obstacles is known a priori, and the robot's motion is planned based on the information obtained. However, in dynamic planning, only partial information about the obstacles is available. Therefore, during the motion execution, the robot receives real-time information about the obstacles, such as position, orientation, and size, which allows it to update the representation of the workspace and modify the motion accordingly. If the workspace or object configurations are prone to change, the planning is classified as a time-varying problem.

In the case of constraints, the motion planning problem can be called **constrained** or **unconstrained**, depending on the existence of inherent constraints on the motion of robots, e.g., constraints that occur for reasons other than collisions with obstacles. These include constraints on a robot's velocity and acceleration. The trajectory planning of any physical system is constrained because the actuator has finite power (Hwang and Ahuja (1992)).

2.4 Motion Planning Methods

There are numerous suitable approaches for handling motion planning problems in robotics, some of which present endless permutations of the same problem, while others present a restricted number of applications. According to Lynch and Park (2017), there is no single planner that is suitable for all problems of motion planning.

Some older motion planning approaches such as classical or combinatorial methods include roadmap, cell decomposition, and potential fields (Choset et al. (2005)). The majority of motion planning problems can be solved using these methods, and in addition, their combinations are often used to develop new motion planning methods. The implementation of these methods requires a high computational cost in environments with large dimensions and a large number of obstacles. This problem occurs because these methods depend on an explicit representation of the obstacles of the configuration space (LaValle (2006)).

Global motion planning methods are abundant since they take into account all the available information of the robotic manipulator's workspace. Therefore, its motions are determined from the initial configuration to the final configuration. The sampling-based algorithms such as roadmap and cell decomposition are examples of global methods. The local methods such as potential fields are projected to avoid obstacles near the robot so that only the information of the nearest objects is used. In general, these methods are implemented as a safety resource to avoid unexpected obstacles that are not represented in the configuration space but can be detected by sensors during the execution of the movements.

According to Lynch and Park (2017) there are several methods of motion planning, which are presented below.

- **Complete Methods** which focus on exact representations of the geometry or topology of the configuration space and ensure completeness.
- **Grid Methods** which discretize the configuration space into a grid and search in this grid for a motion from q_{start} to a grid point in the goal position. An example of this method is the A* grid-path planner used by Tanzmeister et al. (2016) and Han et al. (2021)..
- **Sampling Methods** that rely on a random or deterministic function to select a sample from the configuration space, such as PRM and RRT.
- **Virtual Potential Fields** which create forces on the robot that pull it toward the goal and push it away from the obstacles.

The next sections present details on some of the most popular motion planning methods including sampling methods and potential fields.

2.4.1 Probabilistic Roadmap Method - PRM

The Probabilistic Roadmap Method (PRM) created by Kavraki et al. (1996) is a sampling-based method originally developed for collision-free planning of trajectories in a static environment, involving robotics manipulators with various DOF. The implementation of this method is processed in two different phases:

(i) Learning phase;

(ii) Query phase.

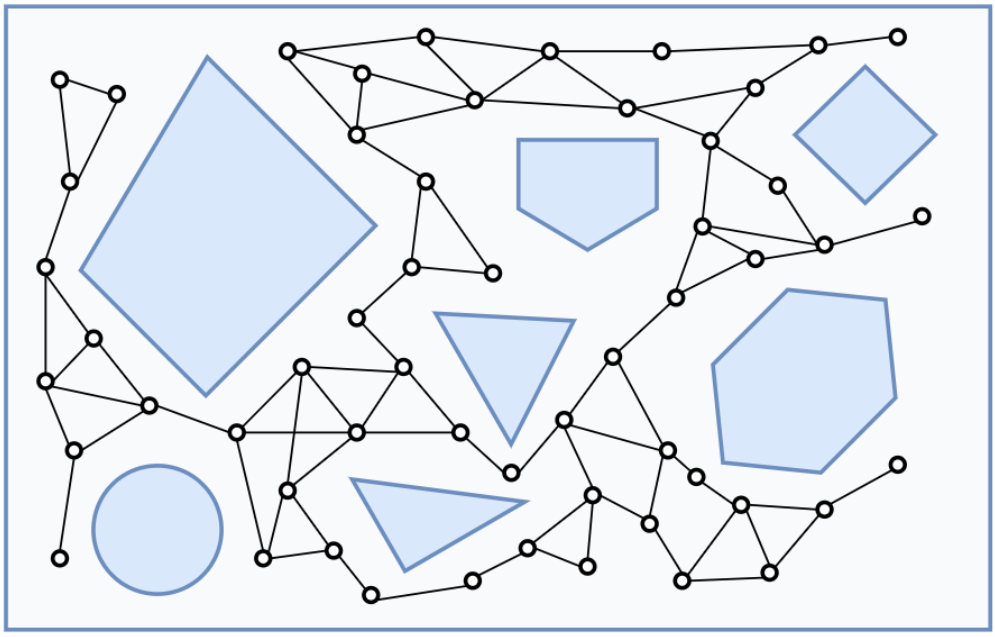


Figure 2.6: Example of a PRM roadmap where the grey shapes are the obstacles, the white is the free configuration space and the circles represent the nodes. [6]

. In the learning phase, a probabilistic roadmap is created by repeatedly generating and connecting random free configurations of the robot. As shown in figure 2.6, the roadmap is stored in an undirected graph where the nodes and edges correspond to the configurations and the free collision paths generated between them, respectively. These paths are computed by a local planner. After the learning phase, several queries can be answered. One query requests a path between two free configurations of the robot. To process this query, the method first tries to find a path from the start to the goal configurations of two

[6] Image adapted from Lynch and Park (2017)

nodes on the roadmap. Then, a graph search is performed to find a sequence of edges connecting these nodes in the roadmap.

The experimental results of Santiago et al. (2017), in which they compare a generic motion planning algorithm and the PRM on the same problem, show that the PRM can efficiently generate a collision-free path for complex maps. However, the trajectories used could be smoother despite its lower computation time and processing results.

In recent decades, there was a need to expand on the dynamic motion planning, particularly in partially know workspaces. According to Short et al. (2016), a variety of sampling-based strategies have been developed for these dynamic issues. One of the most common variants is the Lazy PRM of Bohlin and Kavraki (2000). Collisions are only verified in this approach after the path between the initial and final configurations has been discovered. Essentially, this method aims to minimize the number of collision checks performed during the planning and thus minimize the runtime of the planner. Unlike PRM, this method first assumes that all nodes and edges in the roadmap are collision-free, and searches for the shortest path between the initial and final nodes. The nodes and edges along the path are then checked for collisions. Figure 2.7 shows a high-level flowchart of Lazy PRM.

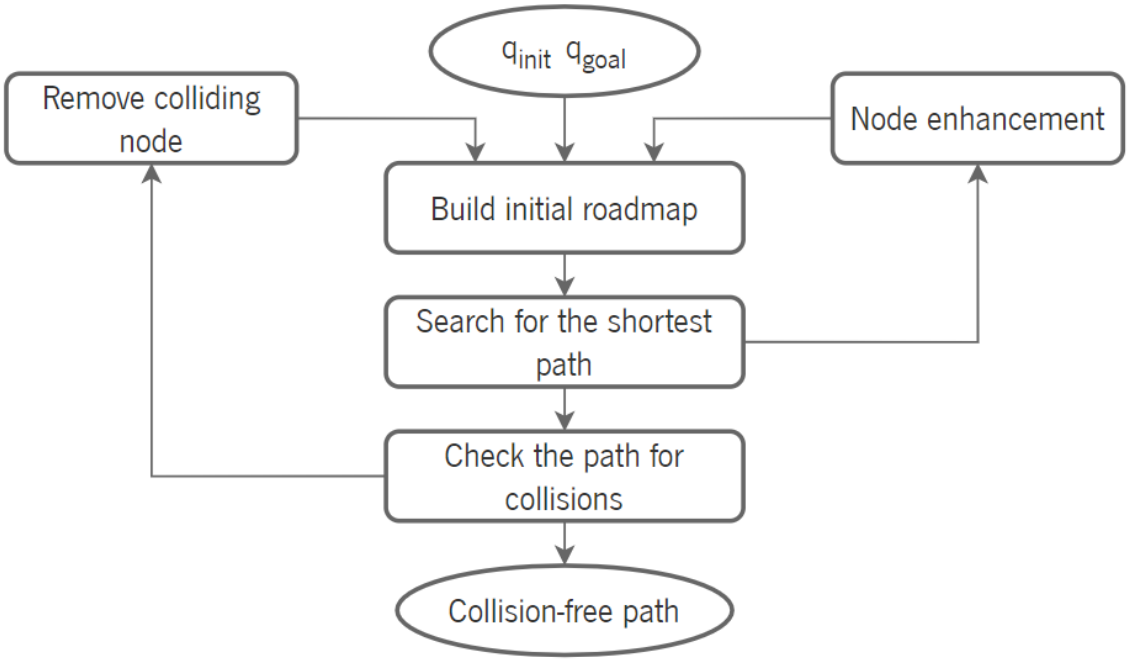


Figure 2.7: High-level flowchart of Lazy PRM [7]

In experiments with the PRM and the Lazy PRM, Hauser (2015) run each planner ten times with

[7] Image adapted from Bohlin and Kavraki (2000).

various random seeds. Although it initially performed poorly because the lazy roadmap made optimistic jumps over obstacles, the lazy PRM outperformed the non-lazy strategy in challenging the narrow route. However, the lazy technique gradually closes the performance gap by performing fewer edge checks. The author concludes that this technique reduces the computational cost per sample because it avoids checking edges that are unlikely to improve the existing optimal path. Although initially performing worse in workspaces with fewer obstacles, the lazy PRM can eventually outperform the non-lazy variant due to a higher convergence rate.

2.4.2 Rapidly-Exploring Random Tree (RRT)

LaValle et al. (1998) developed the Rapidly-Exploring Random Tree (RRT), a sampling-based strategy for coping with algebraic and differential constraints on robotic platforms with various degrees of freedom. Since it is not necessary to create a roadmap or establish connections between pairs of configurations, this method can be used directly for holonomic, nonholonomic, kinematic, and dynamic motion planning problems, unlike the PRM method (Gasparetto and Zanotto (2010)).

According to Lynch and Park (2017), the RRT algorithm searches for a collision-free path from an initial state called x_{start} to a goal set called X_{goal}. It can be used to solve kinematic problems where the state x is simply the configuration q, as well as dynamic problems where the state includes velocity. Starting from x_{start}, the RRT generates a single tree.

In a usual implementation of the kinematic problem, where x is simply q, the sampler randomly selects x_{samp} from a nearly uniform distribution over X, with a tiny preference for states in X_{goal}. The node in the search tree T that minimizes the Euclidean distance to x_{samp} is $x_{nearest}$. On the straight line between $x_{nearest}$ and x_{samp}, the state x_{new} is chosen as the state with a modest distance d from $x_{nearest}$. Because of the small size of d, a simple local planner, such as one that generates a straight-line motion, will often find a motion that connects $x_{nearest}$ and x_{new}. The new state x_{new} will be added to the search tree T if the motion is collision-free. The net effect is that the nearly uniformly distributed samples "pull" the tree toward itself so that it rapidly explores X_{free}. This can be seen in the figure 2.8.

The choice of sampling method, distance measure, and local planner has a significant impact on the performance of the basic RRT algorithm. Apart from these options, there are two other more common RRT versions.

[8]Image adapted from Lynch and Park (2017)

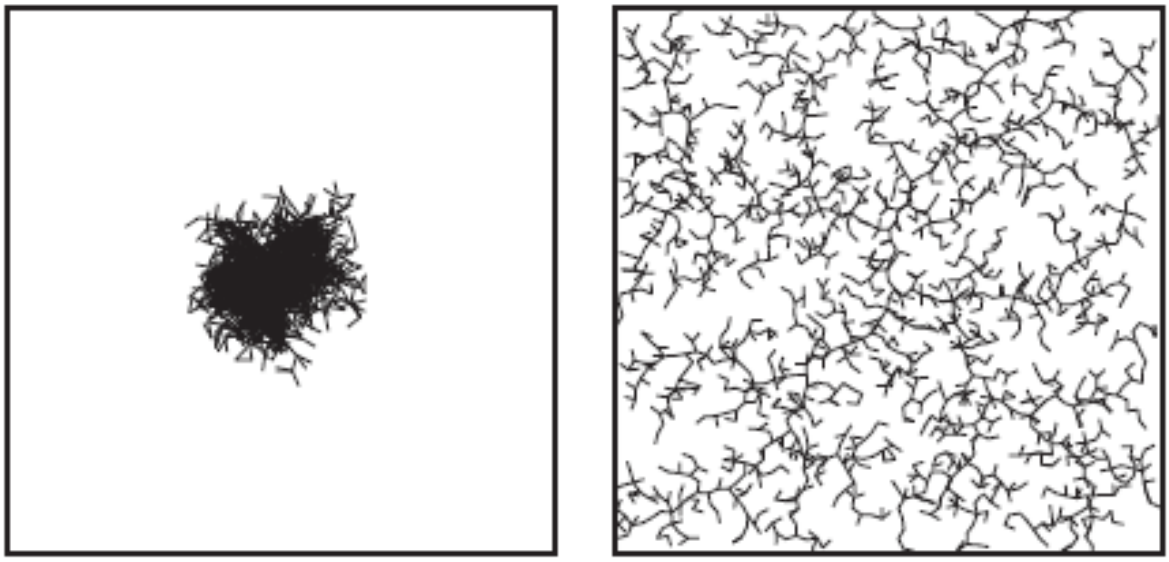


Figure 2.8: On the left, you see a tree created by applying a uniformly-distributed random movement from a randomly selected tree node. On the right, you see a tree created by the RRT method using randomly selected samples from a uniform distribution. There are 2000 nodes in both trees..[8]

Bidirectional RRT Probabilistic algorithms with a single query are designed to quickly solve a particular case of motion planning. Unlike multiple-query planners such as PRM, these techniques do not rely on building a roadmap exhaustively representing the connectivity of the free configuration space. Instead, they tend to explore only a subset of the free configuration space that is relevant to solving the problem at hand. This further reduces the time required to compute a solution. The bidirectional RRT is an example of a single-query probabilistic planner. (Siciliano et al. (2009))

The bidirectional RRT builds two trees: one "forward" from x_{start} and one "backward" from x_{goal}. The algorithm alternately builds the forward tree and the backward tree, and occasionally tries to join the two trees by selecting x_{samp} from the other tree. This strategy has the advantage of achieving a single goal state x_{goal} rather than just a goal set X_{goal}.We can see in figure 2.9 the process to extend the tree on the left as well as the linking of the two trees. Another advantage is that in many situations the probability that the two trees will find each other is much greater than that a single "forward" tree will find a target set. The main problem is that the local planner may not be able to accurately link the two trees.

RRT* Once a movement to X_{goal} is detected, the basic RRT method is successful. Another option is to let the algorithm continue running and stop it only when another termination condition is achieved (e.g., a maximum running time or a maximum tree size). The movement with the lowest cost can then be returned. This allows the RRT solution to improve over time. On the other hand, the RRT does not always

[9]Image adapted from Siciliano et al. (2009)

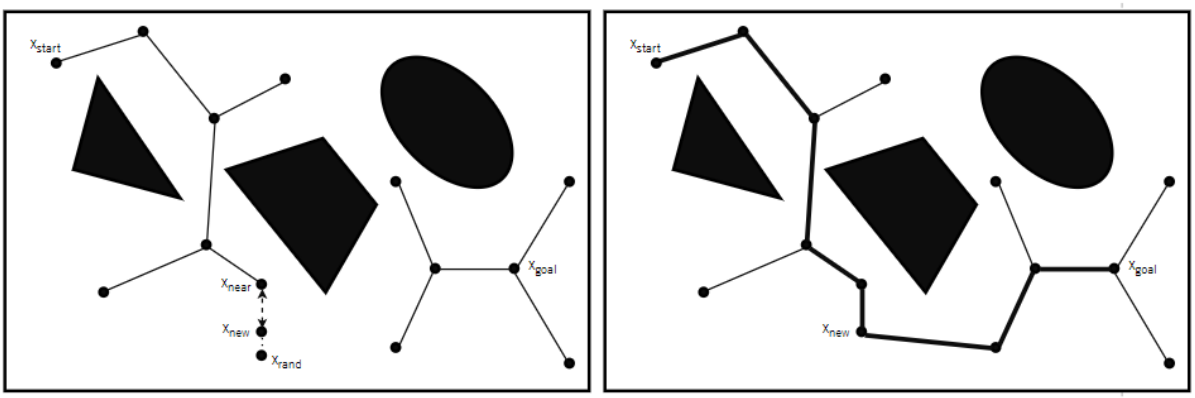


Figure 2.9: In a two-dimensional configuration space, the bidirectional RRT technique shows the randomized process for extending a tree on the left and the extension procedure for linking the two trees on the right.[9]

converge to an optimal solution because the edges in the tree are never destroyed or changed (Karaman and Frazzoli (2011)).

The RRT* method is a variant of single-tree RRT that constantly rewrites the search tree to ensure that the shortest path from x_{start} to each node in the tree is always encoded. The basic technique allows precise paths from any node to any other node in the configuration space without movement boundaries.

? compared the RRT and RRT* algorithms by running them in a square environment and the cost function is set to the Euclidian path length. The figure 2.10 shows that the RRT approach does not significantly improve the solution, but the RRT* algorithm converges to an optimal solution by identifying a feasible solution of the homotopy class in which the optimal path exists. Therefore, in a congested environment, the cost of the first feasible solution discovered by the RRT or RRT* algorithms may be significantly higher than the ideal cost. Although the RRT* method effectively improves the solution over time, the RRT algorithm tends to get stuck at the first solution found.

2.4.3 Potential Fields

Potential energy fields in nature, such as gravitational and magnetic fields, have inspired virtual potential field methods. Khatib (1986) developed the potential field's approach, which is the first local method for motion planning. Its application in real-time, without prior knowledge of the configuration space, could allow robotic platforms to autonomously plan their motions. In robot motion control, the goal configuration qgoal has a negative virtual potential - attractive - while obstacles have a positive virtual potential - repulsive - as can be seen in figure 2.11. The potential field is defined by the total of the above potentials, and its negative gradient determines the trajectory of the manipulator's end-effector. When a force proportional

[10] Image taken from **?**

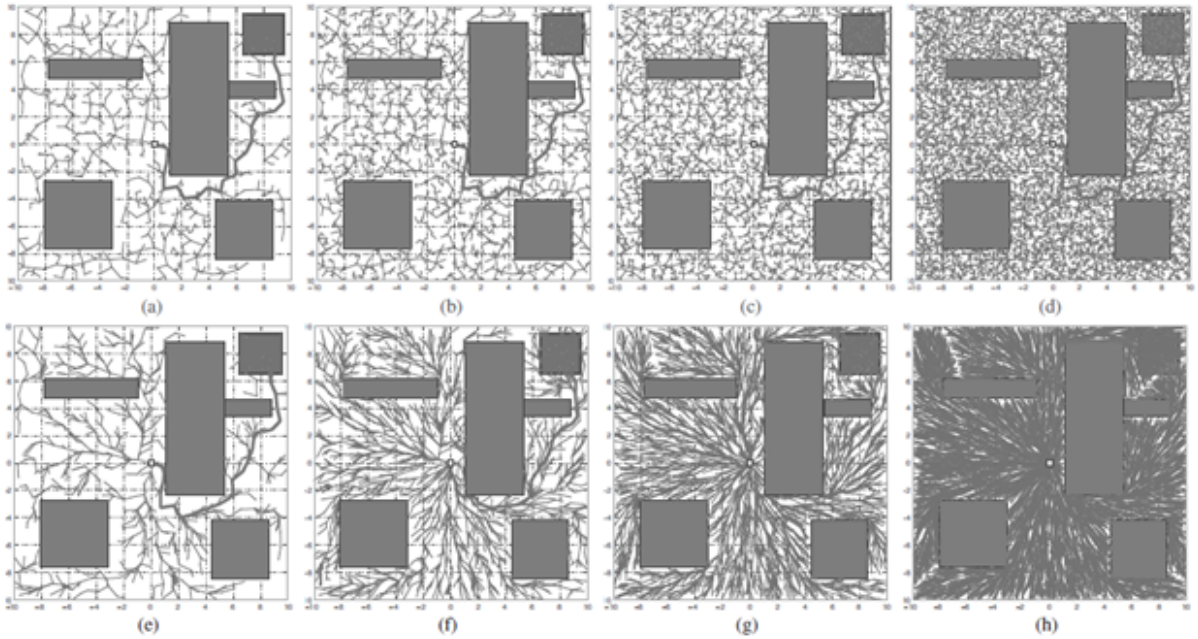


Figure 2.10: On a simulation example, the RRT and RRT* algorithms are compared. The RRT algorithm's tree is shown in (a)-(d) in various stages, whereas the RRT* algorithm's tree is displayed in (e)-(h) in various stages. (a), (e) are 1000 iterations, (b), (f) are 2500 iterations, (c), (g) are 5000 iterations, and (d), (h) are 15,000 iterations.[10]

to the negative gradient of the virtual potential acts on the robot, it automatically pulls it toward the target and away from obstacles (Lynch and Park (2017)).

Since the gradient of the field can usually be calculated quickly, the motion can be calculated in real-time (reactive control) and does not need to be planned a priori. The approach can even handle obstacles that shift or arrive unexpectedly if appropriate sensors are used. This basic method has the disadvantage that the robot can get stuck in the local minima of the potential field.

When the attractive potential of the goal and the repulsive potential of the obstacles are equal, the gradient vector of the potential field becomes zero and the robot is stuck in a deadlock. According to Siciliano et al. (2009), a simple solution to the local minima problem is to use a best-first method developed under the assumption that the free configuration space has been discretized with a regular grid.

2.4.4 Attractor Dynamics

Autonomous reaching, grasping, transporting, and handing over must occur under certain constraints in robotic assistance. Since the human operator may be working at a shared workplace, the environment is constantly changing. Dynamic conditions are important not only for obstacle avoidance, but also for transport movements, e.g. when the robotic manipulator has to hand over an object or place an object on

[11] Image adapted from Lynch and Park (2017)

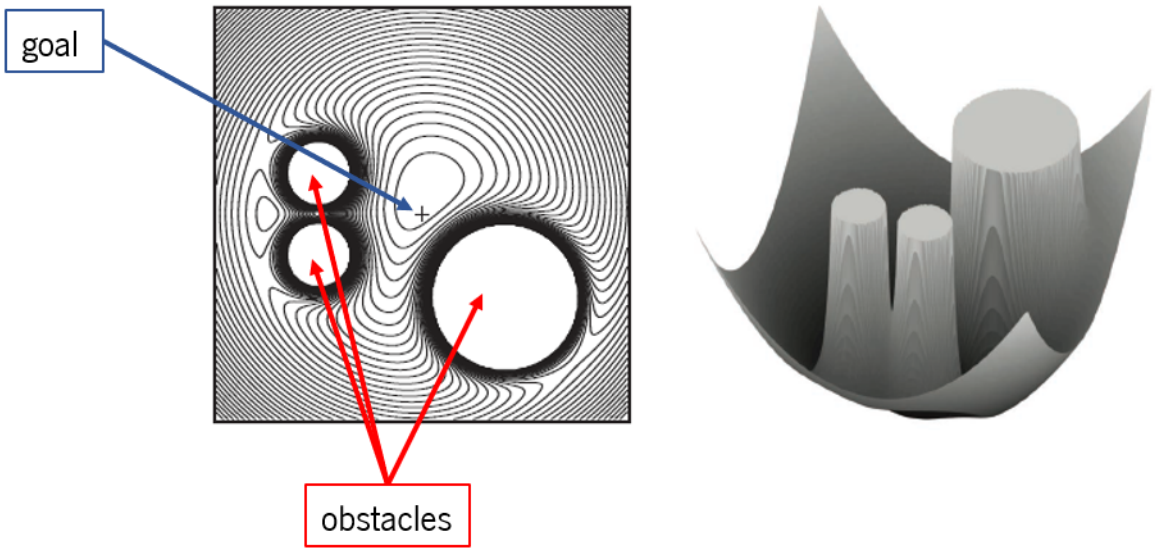


Figure 2.11: Various representations of the potential fields method, where there are three obstacles and a goal point (+). The potential field is the sum of the attractive and the repulsive potentials.[11]

a part that the human operator is holding (Iossifidis and Schöner (2004)).

The potential fields method, as mentioned earlier (2.4.3), is probably the heuristic strategy best suited to deal with changing situations and imprecise data. An alternative technique based on attractor dynamics has been researched since 1992. The attractor dynamics method is a mathematically formalized variant of the potential field method. The motion plan is generated when the planning variable is at the minimum (the attractor), rather than by the temporary downward solution as in the standard potential fields method, using the direction of the movement rather than the Cartesian position of the vehicle.

Two angles of heading direction are used to generalize the attractor dynamics approach to the control of robotic manipulators. These determine the direction in which the end-effector moves in three-dimensional space. Figure 2.12 shows two different heading direction angles. These can be defined using the spherical coordinates of a coordinate system associated with the end-effector but aligned with an arbitrary but fixed world reference frame. The angle between the direction in which the end-effector is moving and the vertical axis is called the elevation, $\theta(t)$. The azimuth, $\varphi(t)$, is the angle formed by projecting the direction of the end-effector onto the meridional plane and the arbitrary but fixed x-axis of the world coordinate system. The third variable is the path velocity, which is kept constant throughout the path. The result is a relatively simple method based on two equations in which the target acts as an attractive force and the obstacles as repulsive forces, and the end-effector follows a collision-free path.

The benefit of using the directional approach is that it provides a straightforward way to avoid some

[12] Image adapted from Iossifidis and Schöner (2006)

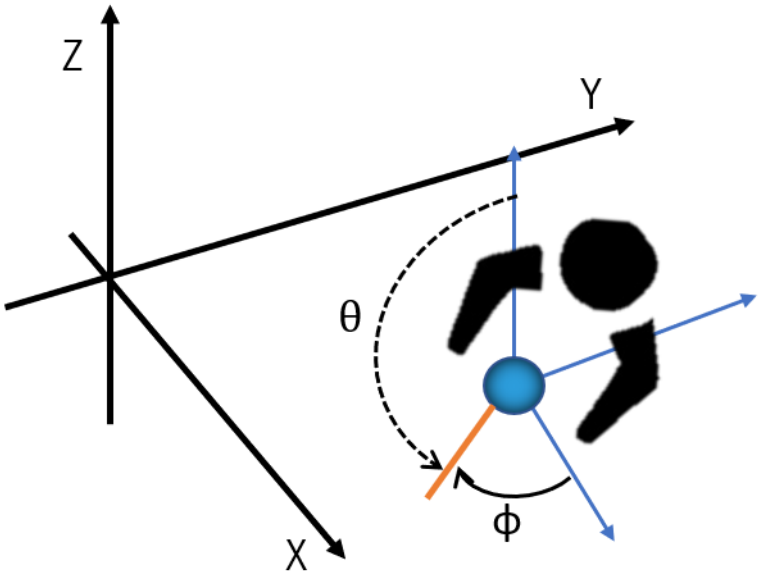


Figure 2.12: The position of the tip of the manipulator (sphere) is represented in the world coordinate system, fixed on the surface of the workspace $z = 0$. The direction of the current navigation is represented by relative angles to the vertical axis (elevation, θ) and to the x-axis (azimuth, ϕ). [12]

of the problems associated with potential fields methods, such as local minima and oscillations, without incurring the high processing costs associated with accurate solutions. The attractor dynamics approach, based on bifurcation theory, allows decision-making to be controlled and designed in the presence of several minima. As a result, the motion plan preserves temporal continuity, which is critical in higher-dimensional motion control problems in robotic manipulators. In comparison with the potential fields method, the attractor dynamics approach is simpler because it does not require calculation and control of forces or torques, but it is limited to kinematics (Iossifidis and Schöner (2006)).

Reimann et al. (2010) used the attractor dynamics method to prevent all segments of the robot CORA from colliding with the numerous obstacles in the workspace. The obtained results showed that this method accurately planned trajectories of different complexity levels and with different numbers of obstacles. However, some related problems were identified, such as a cancellation effect in which the contributions of different obstacles push the joint configuration in the joint space in opposite directions, effectively weakening the avoidance effect, and insufficient velocity reduction in the null-space task, which makes it impossible for the system to select valid trajectories to reach the target position.

2.4.5 Optimization Methods

In robotics, motion planning is a critical component. It is vital to plan a smooth and collision-free trajectory while manipulating an object with a robotic manipulator to accomplish a task. However, existing

motion planning methods often have drawbacks, such as the fact that a motion planner can only develop a single solution for a given setting and that target configurations must be defined by the user (Osa (2020)).

Optimization approaches provide dynamically feasible motions that maximize a performance criterion while satisfying a set of constraints. This is an interesting method for nonlinear and unstable systems, such as legged or balanced robots, where an optimal solution cannot be found analytically from the required optimality requirements (Pardo et al. (2016)).

The fundamental drawback of these approaches, such as sampling methods, is that the computed path contains unwanted detours and must be post-processed before it can be executed by a virtual or real robot. However, there are other possibilities. Obstacle avoidance is managed by constraints or costs by calculating the distance to the nearest obstacle during path optimization planning. Most of these planners use nonlinear optimization under constraints. For simple problems, such planners produce near-optimal paths and take less time to compute. Ideal random samples are also close to an optimal answer, although the computation takes much longer than with traditional methods.

The optimization process is always focused on one or more criteria. Most commonly, motion planning considers path length, where deviations are penalized, obstacle clearance for safety, and execution time, which is determined by the number of via points that introduce velocity discontinuities. To properly execute the action of a robot, one can additionally propose minimizing the acceleration or jerk peaks (Campana et al. (2016)). Fiorini and Shiller (1996); Shin and Mckay (1985); Bobrow (1988); Lin et al. (1983); presented the first optimization algorithms for motion planning that used minimization of execution time as an objective function. However, several of these algorithms produced trajectories with discontinuous acceleration and torque values at the joint level, which led to dynamic problems in the execution of the planned motions.

The development of alternative methods based on the formulation of new performance criteria such as minimization of energy consumption during motion execution and minimization of acceleration fluctuations is the result of the extension of optimization methods in trajectory generation. Currently, approaches based on the latter criteria are widely used in the industry and offer several advantages, including the reduction of errors in joint positioning, the generation of coordinated and smooth motions, the limitation of the resonant frequencies of the robot, and the reduction of the wear of the manipulator actuators and their respective structures (Gasparetto and Zanotto (2010)).

Khatib's work on the use of potential fields, published in 1986, is considered the fundamental work on optimization-based motion planning. CHOMP (Zucker et al. (2013)), STOMP (Kalakrishnan et al. (2011)), Tra-jOpt (Schulman et al. (2014)), and GPMP (Mukadam et al. (2018)) are some of the more modern

optimization-based methods. These methods explicitly optimize the trajectory with respect to the objective function.

2.5 Sorting in Anthropomorphic Robots

One of the most important goals in the field of robotics is to implement more intelligent, flexible, and independent robots that are able to interact with their environment (Liu et al. (2012)). Thus, there is an increasing need for autonomous and flexible robotic systems that are able to grasp, manipulate, and sort randomly oriented, clustered, and unknown parts into the appropriate place. These systems are very important in industry as they perform a task that is very monotonous, strenuous, and dangerous for humans. According to Tauro et al. (2010), the most complex sorting task is called bin-picking or more commonly known as bin sorting.

2.5.1 Pick & Place in Sorting Tasks

There are various sorting tasks, and they may vary depending on the implementation and system. However, there is a common ground in all of these systems, and that is the inclusion of pick and place tasks. We will examine different approaches to sorting systems.

Amin developed an embedded system which can increase the speed of a color sorting procedure

Patel et al. (2017) focused on programming robots that use human pointing gestures to automatically sort objects into bins. In many industries, programming robots is a time-consuming task with a high degree of difficulty and cost that requires specialized training. Therefore, the researchers developed methods for learning actions and behaviors for robots to learn through human demonstration - Programming by Demonstration (PBD). The system the authors used consisted of an RGB-D camera (Microsoft Kinect 2.0), a six-axis robotic arm from CRS Robotics, and an electromagnetic gripper. They presented a Fault-Tolerant Programming by Demonstration (FTPBD) system whose goal is to automatically sort industrial objects into bins after the human operator indicates which bin the object belongs in. Patel et al. (2017) divided the tasks of the system into the learning task and the sorting task. Since it is a fault-tolerant system, the authors assume various errors that can come from both the human and the robot, such as inaccurate pointing during the learning task, an unclassified object being identified during the sorting task, the human is in the robot's workplace during the sorting task and others. Also, various assumptions are made such as the objects and bins are within the field of view of the camera, the shapes of the objects are different from each other, the objects do not overlap, and more. We can see the results obtained by the author in

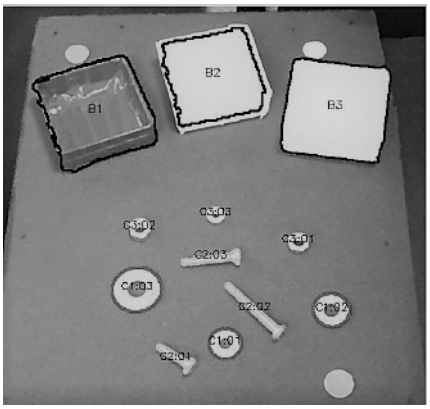


Figure 2.13: Objects identified by human pointing gesture. [13]

the figure 2.13.

The authors concluded that the proposed method can be executed quickly, the human-robot interaction is effective, and the experiments with industrial objects were successful. However, they acknowledge that many improvements are possible, such as the existence of overlapping objects. Although this is an effective implementation, it is prone to human error, so autonomous sorting tasks would be more beneficial overall.

Nieuwenhuisen et al. (2013), however, used a non-traditional method not normally seen in the industry. Typically, sorting and pick and place tasks are performed in a static environment where the robotic manipulator is fixed in a specific location. So the authors took a different approach and demonstrated sorting using an autonomous anthropomorphic mobile robot. Objects were recognized in a series of images using a shape primitive approach. Grasping and arm motion was planned in an efficient local multi resolution height map.

They considered the scenario where unordered parts had to be grabbed from a transport box and placed in the processing station. The robot used was the Cosero cognitive service robot, which navigates on an eight-wheeled omnidirectional base and has a 7DOF robotic manipulator that ends with two-finger Festo FinGripper grippers. They used a 3D object recognition method based on subgraph.

The object recognition method used by Nieuwenhuisen et al. (2013) was compared with the method used by Papazov and Burschka (2014) using point-pair features. They concluded that their approach found slightly fewer objects but did not produce false positives. In the case of mobile sorting, in thirty-two runs of picking up and delivering a pipe connector, the robot successfully picked up the pipe connector and delivered it to the correct station in twenty-eight of those runs. In future work, the authors hope to improve the object recognition performance. A full video explaining the entire process and results can be viewed on the authors' website[14].

[13] Image adapted from Patel et al. (2017)

[14] www.ais.uni-bonn.de/ActReMa.

Two different sorting methods have been explored, one sorting method that focused on human gestures and another sorting method that did not take place in a static environment. However, there is a common feature between them - the objects being sorted are very similar to each other and do not have much variety. The literature review on object sorting focused on the use of sensing and grasping technologies that are used to pick a single type of object that is randomly positioned and scattered. Zhang et al. (2018) noticed this pattern and motivated the authors to present an industrial robot application called Gilbreth that picks up objects from a moving conveyor belt and sorts these objects into bins according to the different types.

Gilbreth is an application where different types of objects arriving on a moving conveyor belt are identified, grasped, and picked up by an industrial UR10 robot manipulator and then sorted into bins according to the type of object. They combine object detection, segmentation and recognition, robot pose planning, robot motion planning, and robot execution by a robot controller using ROS and ROS-Industrial packaged. The Gilbreth setup can be seen in figure 2.14.

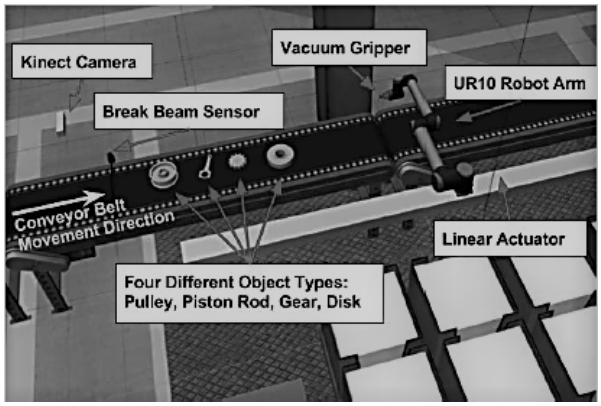


Figure 2.14: Gilbreth setup: sensors, objects, conveyor belt and UR10 robot arm. [15]

The authors conclude that it is possible to implement a pick and place application combining computer vision with motion planning to provide the robotic manipulator with the ability to position itself and its end-effector to pick up objects from a conveyor belt, move to the appropriate bin, and place the object inside. They conducted two different experiments. In the first experiment, they concluded that the robot's execution time for the task of picking up and placing takes about the same amount of time as the time required for object detection. In the second experiment, they found that the object recognition time could decrease with fast servers. For future work, they plan to support multiple robots, implement a dynamic position pickup policy, and generally improve Gilbreth performance and expand the object types.

In addition to motion planning, this dissertation also has an element of sorting. Therefore, a litera-

[15] Image adapted from Zhang et al. (2018)

ture review of current sorting methods is important to better understand which method would be more appropriate for the current problem.

2.6 Discussion

The motion planning methods presented in the previous sections have led to a greater integration of computer-controlled robotic systems into industrial production lines in recent decades. Sampling-based approaches such as PRM (Kavraki et al. (1996)) and RRT (LaValle et al. (1998)) are widely used to solve very complex motion planning problems in this domain. Despite their advantages, scientific experiments show that these methods are inefficient for real-time applications on robotic platforms with different DOF. According to Khatib (1986), the potential fields approach is a simple, effective, and computationally sustainable planning process. However, as discussed earlier in the chapter, there are problems with local minima that can be overcome with a best-first method created under the premise that the free configuration space has been discretized with a regular grid (Siciliano et al. (2009)). Gasparetto and Zanotto (2010) used optimization methods to predict the trajectory of manipulators based on performance criteria. Although these methods involve a selection process constrained by constraints to avoid collisions with existing obstacles in the workspace and to respect the overall physical limits of the manipulator, the predicted trajectories do not have the same characteristics as those observed in experimental work with human upper limbs. Motion planning solutions that focus on efficiency and computation time to increase productivity and reduce costs in industry are the most used methods in robotics. With the integration of robotic platforms into shared workspaces with human operators, robotics is now focusing on the research and development of technologies that enable human-like motion planning. In this case, we want robotic manipulators to be able to reach, grasp, and manipulate objects in the same way as humans.

One of the most common tasks in the industry is pick and place tasks, as they can be very tedious and repetitive. One of the applications of pick and place tasks is bin sorting. In a bin sorting system, pick and place tasks are performed by the robot to pick an object from a cluster of different objects, transport it, and place it in the designated area. However, this can present many difficulties as the appearance of the objects may be distorted due to overlapping parts, illumination variations, or reflections. Many methods have been proposed for image processing for bin sorting, such as object recognition, shape recognition of orientation features using 2D or 3D image sensors (Kim et al. (2012)).

These sorting tasks can be applied and implemented in various industrial systems. Patel et al. (2017)

[15]Image adapted from Nieuwenhuisen et al. (2013)

implemented sorting tasks using human pointing gestures to automatically sort objects into bins. Nieuwenhuisen et al. (2013) took a different approach and used a mobile anthropomorphic robot to perform the sorting tasks, which is an unusual method since in the industry robots tend to be static in one place. Zhang et al. (2018) applied sorting methods to a conveyor belt, differentiating in the objects used as he examined more different objects rather than the same object which is usually more common. The work made by Zhang et al. (2018) is the one that is closest to the goal of this dissertation since a robotic manipulator of various DOF was used, and also the different objects to be sorted is a very important similarity to what we want to do.

Chapter 3: Human Motor Control

The motion planner deployed in this dissertation, i.e. HUMP, which was developed to generate smooth and human-like trajectories, was heavily inspired by the understanding of human upper-limb functional models and their properties. Hence, in this chapter, the properties of human arms are presented with details on arm kinematics and hand properties, which are essential for understanding the grasping of objects depending on the task to be performed. In addition, this chapter presents different modeling approaches to motion planning problems in humans, the main characteristics of upper-limb movements, and finally the experimental work related to upper-limb movements. This chapter also follows presentations already made by de Sá (2018) and Silva (2011).

3.1 Human Arm Characteristics

The human body is a complex system of muscles, bones, and tissues that together allow the body to move. According to Freivalds (2011), human movements, being three-dimensional and centered around joints, are defined by three anatomical planes that divide the human body, namely: sagittal plane, coronal or frontal plane, and transverse plane, as we can see in the figure 3.1.

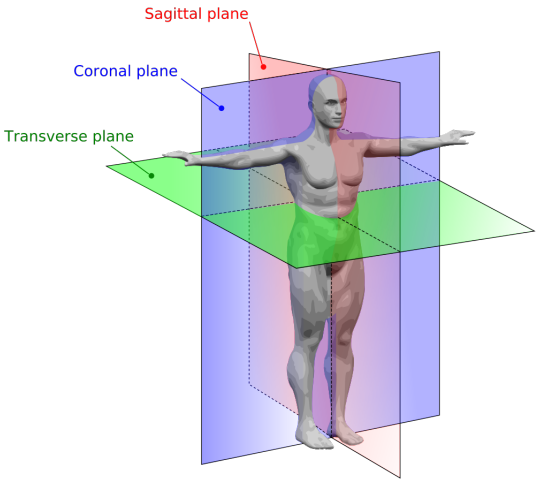


Figure 3.1: Model of the human anatomical planes.[1]

A transverse plane is an imaginary plane that divides the body horizontally into an upper and a lower part. The sagittal plane is the imaginary vertical plane that divides the body into the left and right parts

[1]Image taken from website: https://courses.lumenlearning.com/austincc-ap1/chapter/practice-anatomical-location/.

of the body. Finally, the coronal plane divides the body into the anterior and posterior parts. According to human kinesiology, arm and hand movements depend on the arm joints, namely the shoulder, elbow, and wrist. Human anatomy considers seven main arm movements (Freivalds (2011); Maciejasz et al. (2014)):

Shoulder - the shoulder allows for the following movements:

(i) Flexion and Extension movements;

(ii) Adduction and Abduction movements;

(iii) Internal and External rotation.

Elbow - the elbow allows for the following movements:

(i) Flexion and Extension movements;

(ii) Pronation and Supination movements.

Wrist - the wrist allows for the following movements:

(i) Adduction and Abduction movements;

(ii) Flexion and Extension movements.

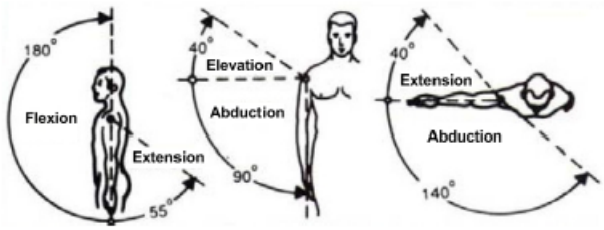
For a better understanding of the previously mentioned movements, the figure 3.2 illustrates the movements for each joint.

The amplitude of these movements can vary from person to person, as we humans are not all the same and do not all have the same flexibility. Since each joint can have more than one DOF, a joint is capable of several different movements.

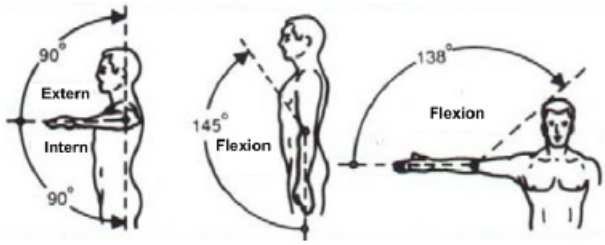
3.1.1 Arm Kinematics

The human arm consists of three main segments - the upper arm segment, the forearm segment, and the hand segment (Kim et al. (2012)). The upper arm segment is connected to the shoulder via the shoulder joint, the forearm segment is connected to the upper arm segment via the elbow joint, and finally, the hand segment is connected to the forearm segment via the wrist joint, giving a total of 7 DOFs, as shown in figure 3.3. The arm consists of a shoulder joint with 3 DOFs, an elbow joint with 1 DOF and a wrist joint with 3 DOFs.

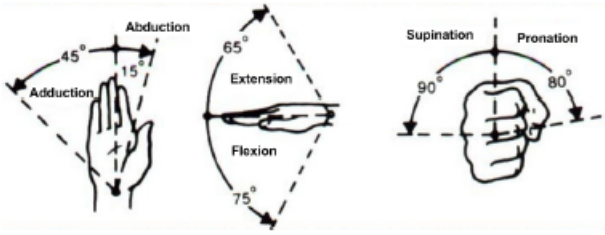
[2]Image taken from Silva (2011)
[3]Image taken from Silva (2011)



(a) Upper Arm Flexion/Extension, Elevation/Abduction and Extension/Abduction



(b) Upper Arm Rotation and Elbow Flexion



(c) Wrist Adduction/Abduction, Wrist Extension/Flexion and Wrist Supination/Pronation

Figure 3.2: Joints of the human arm and corresponding movements.[2]

The human arm is a redundant subsystem because it has 7 DOFs. However, it only needs 6 DOFs to position and orient the hand. This extra DOF manifests as an internal movement of the limb segments that does not move the hand, hence the kinematic redundancy (Perry et al. (2009)). Considering this redundancy, many joint angles can produce the same position, the specification of the hand position is not suitable to specify the joint angles of the arm. Despite the mathematical problems, it has an advantage in that it allows the arm to have different configurations that can help to deviate from obstacles that may occur in the trajectory.

3.1.2 Hand Characteristics

The human hand is a highly complex system with more than 20 DOF, about 4 DOF per finger and 5 DOF per thumb (Kemp et al. (2008)). Because of its 5 fingers, the human hand can control all of them, resulting in a large number of possible manipulation movements. Napier (1956) was one of the first to

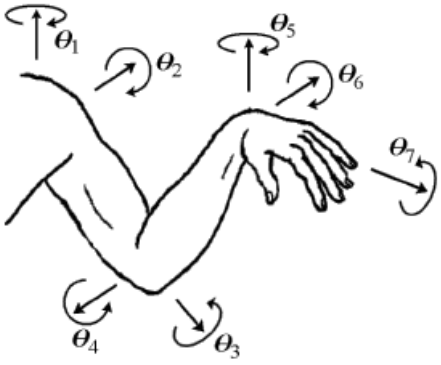


Figure 3.3: Model of a human arm and hand.[3]

study and classify these movements, resulting in two main groups:

Prehensile Movements - movements in which an object is grasped and held wholly or partly in the hand;

Non-Prehensile Movements - Movements in which objects are not grasped or seized but are manipulated by pushing or lifting movements of the hand.

Despite the wide variance in grasping movements, the author concluded that only two patterns are good anatomical bases for all activities of this type:

(i) **Power grip** - the object is held in a clamp formed by partially flexed fingers and the palm (Figure 3.4(b));

(ii) **Precision grip** - the objects are clamped between the flexor sides of the fingers and the opposite thumb (Figure 3.4(a)).

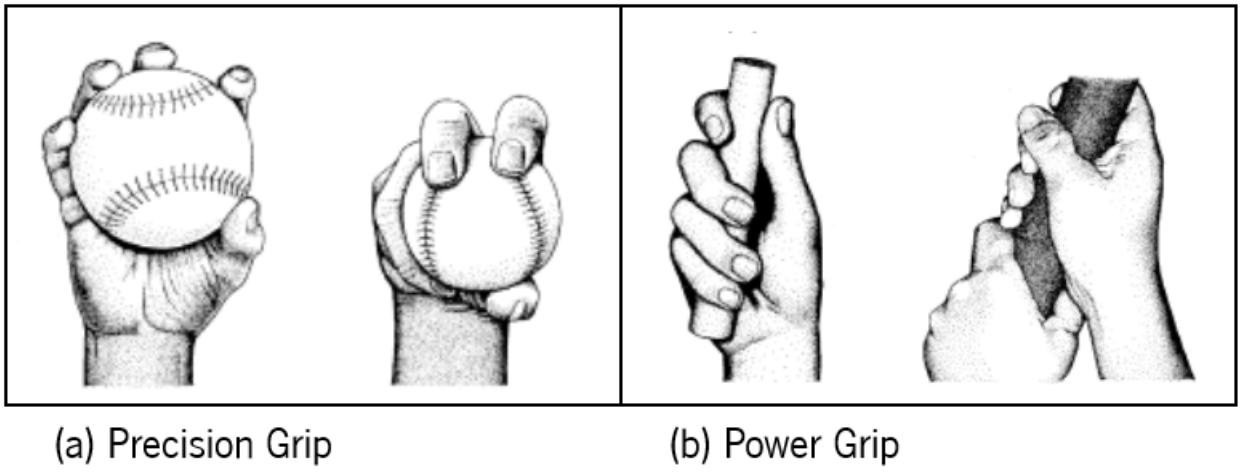


Figure 3.4: (a) Precision Grip; (b) Power Grip.[4]

According to Napier (1956), the type of grip chosen for a task is primarily influenced by the properties of the objects such as shape, size, weight, texture or temperature.

3.2 Computational Models of Motion Planning in Humans

To understand how the various DOFs of the human body are coordinated to perform a manipulation movement, we must consider the central problem of human motor control. Some studies have focused on human upper limb motion planning, specifically how the Central Nervous System (CNS) considers the various variables and factors to proceed to plan a particular movement.

According to Rosenbaum et al. (2001), movements are executed in response to various signals that may come from an external environment, such as the changing of a traffic light, or from an internal environment, such as thoughts and feelings. To execute a movement, a hierarchical structure with three levels has been proposed for the human motor control system: motor planning, motor programming, and motor execution. At the level of motor planning, the planning of movements is generated, at the level of programming, the appropriate motor commands are determined, and finally, at the level of motor execution, these commands are applied to the motor system. During the execution of the movement, these three levels constantly interact with each other, allowing the system to update and change trajectories (Andani and Bahrami (2012)).

According to Wolpert (1997), motor planning can be viewed as a computational process of selecting a single solution, taking into account that for the simplest task, such as moving the hand to a target location, there is an infinite number of possible trajectories, velocity profiles, and combinations of joint angles that the hand could follow. To address this problem, several computational approaches have been developed to model and describe motion planning in humans. According to Andani and Bahrami (2012), these approaches can be classified as:

Data-based models - A set of empirical behavioral data used to create the database for a typical movement. Using this database, forward and inverse kinematics models are developed to predict movement based on features such as initial and final position and anthropometric data (segmental mass and length);

Optimization based models - It is assumed that the CNS considers several performance criteria to plan a motion, such as minimum jerk, minimum torque change, or minimum angle jerk.

According to Wolpert (1997), the optimization-based models are distinguished into two categories:

[4]Image adapted from Young (2003)

(i) Kinematic based models

(ii) Dynamic based models

The cost function, the integral of the instantaneous cost over a certain time interval, considers only the geometric and time-based properties of the motions in kinematic models. The cost function of dynamic models depends on the dynamics of the arm and uses variables such as joint torques, forces acting on the hand, and muscle commands. The main difference between these two models is the separation between planning and execution. In kinematic models, the movement is specified in terms of position and velocity, for which it is necessary that the two processes are executed separately. In dynamic models, on the other hand, the process is executed simultaneously, since the motion is already specified by motor commands.

In the majority of cases, the optimization-based models produce better human-like motions. However, some basic features are not considered, such as real-time planning. Besides this drawback, these models have high computational costs (Andani and Bahrami (2012)).

3.2.1 Characteristics of Human Upper-Limb Movements

In human motor control, the focus tend to be on upper limb movements. Although the initial focus was on the redundancy problem, more studies have been concerned with the different phases of pick & place tasks (Kang and Ikeuchi (1994)). In this study, the pick and place task is divided into a series of phases and transitions by analysing the velocity of a human hand. This division can be seen in figure 3.5.

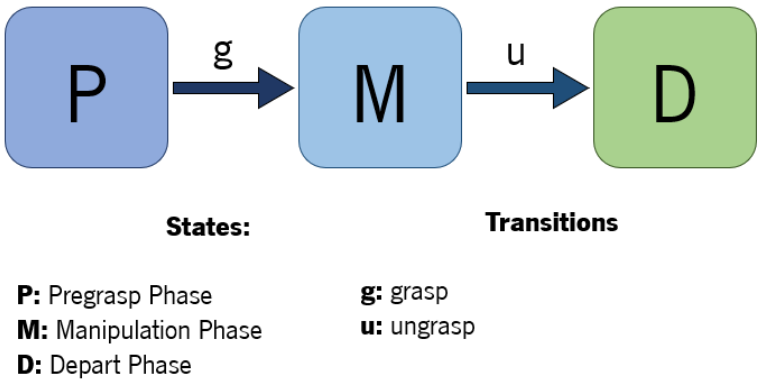


Figure 3.5: State Transition Diagram of a pick&place task.[5]

According to the author, we start with the pre-grasp phase, which ends when the end-effector touches and grasps the object, completing the grasp transition. In the manipulation phase, the object is transported to the desired position, after which the ungrasping transition begins, where the object is released. The

[5]Image adapted from Kang and Ikeuchi (1994)

picking up and putting down motion is completed once the end-effector moves away from the object, which is the departure phase. The trajectory of the hand is affected by the distance of the object to the hand, while the changes in the finger joint parameters depend on the shape of the object.

In pick & place tasks, humans tend to make several movements in which they pick up an object, then transport it, and finally place it in another location. One of the first experimental studies in humans examining upper limb behaviour suggests that humans tend to perform straight-line trajectories to the target location, generally exhibiting a bell-shaped velocity profile (figure 3.6). This behaviour appears to be independent of the movement execution workspace (Flash and Hogan (1985)).

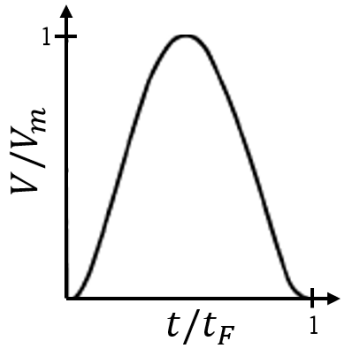


Figure 3.6: Bell-shaped velocity profile.[6]

Although the model by Flash and Hogan (1985) focused more on the properties of the movement to an already known final posture, Rosenbaum et al. (1995) focused on the means by which final postures are chosen - the Knowledge Model. The planning of the movement is based on an evaluation of the stored posture representations. The mathematical ideas of the model are shown in figure 3.7.

In this model, the postures are represented as vectors (Figure 3.7(A)) in a joint space, and the dimensions of the space are the hip, shoulder, and elbow angles. At the beginning of the reaching task, there are initially three stored postures in the example shown: P_1, P_2, and P_3. After the spatial target is identified, the weights w_1, w_2, and w_3 are assigned to the three posture vectors (Figure 3.7(B)). Next, the stored postures are summed vectorially, yielding a target posture (Figure 3.7(C)). Finally, a trajectory from the start posture to the target posture is created (Figure 3.7(D)).

A few years later, Rosenbaum et al. (2001) and Gréa et al. (2000) began new studies supporting the idea that the desired final posture is initially planned and used as a control variable by the central nervous system (CNS). Then Rosenbaum et al. (2001) proposed the division of the motion planning problem into two subproblems:

[6]Image adapted from **?**

(i) Final posture selection;

(ii) Trajectory selection.

. This model is the Posture-based motion planning method, which will be discussed in more detail in the next subsection.

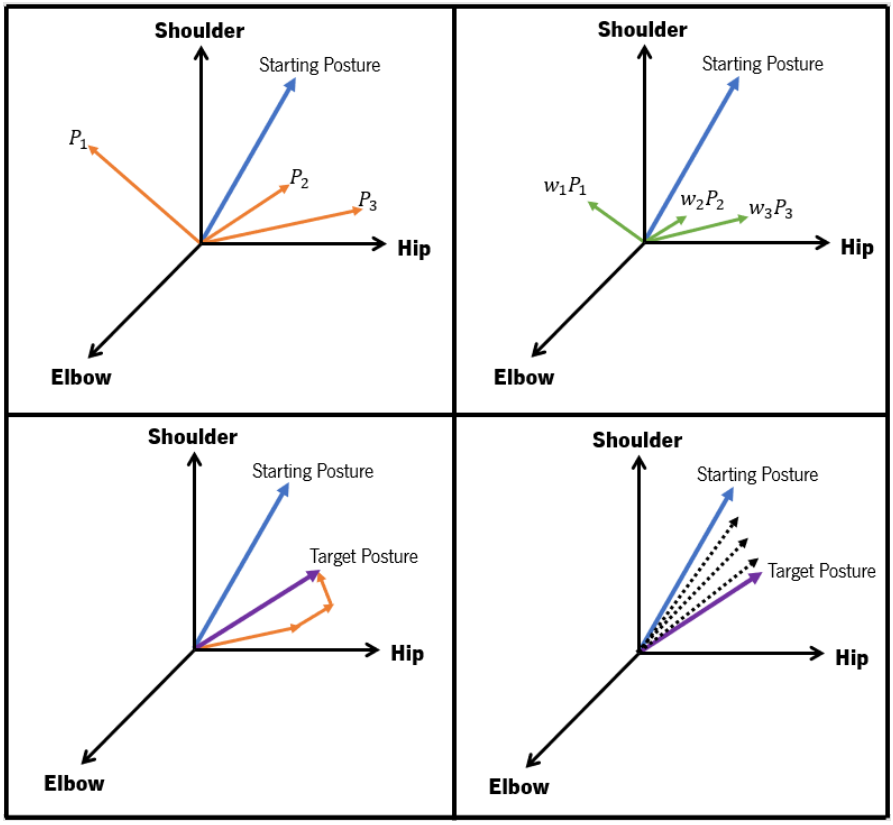


Figure 3.7: Representation of the Knowledge Model.[7]

3.2.2 Human Arm Movement in Obstacle Avoidance

In most of our daily tasks, we must use our upper limbs to avoid collisions with numerous objects in our environment. Rosenbaum et al. (2001) and Vaughan et al. (2006) studied this human ability and developed an experimental work-based model called posture-based motion planning that can produce smooth, fluid, and collision-free movements (Bicho et al. (2011); Gulletta et al. (2015); Silva (2011)).

The application of this motion planning method requires a set of constraints, prioritized and associated with the task to be done. The authors propose to divide motion planning into two subproblems to perform grasping and manipulating motions: final posture selection and trajectory selection.

[7] Image adapted from Rosenbaum et al. (2001)

The planning process in which postures are produced around a promising stored posture until time runs out is referred to as *diffusion 'til a deadline* (Figure 3.8(b)). The final posture is selected by comparing the performance of the new postures given the constraints of the task. The generated posture that satisfies the most constraints is selected as the final posture (Figure 3.8(c)).

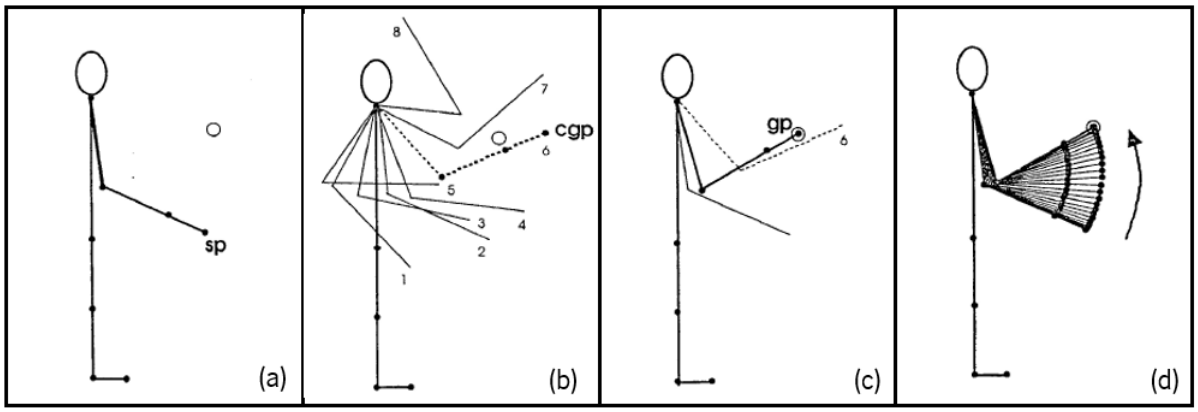


Figure 3.8: Representation of a planning a movement from an initial posture to touch a target: **(a)** Initial posture (sp) and target (circle); **(b)** Most adequate posture (cpg); **(c)** Generated posture (gp); **(d)** Direct movement to be executed between initial and final posture. [8]

Subsequently, if a collision-free path is confirmed, interpolation across joint space is used to perform direct movement between the initial and final postures (Figure 3.8(d)). However, if a collision is detected, a back-and-forth motion is performed according to the Rosenbaum et al. (2001) model. This motion is added to the direct movement to create a collision-free trajectory to the final posture that was previously calculated (Figure 3.9(d)).

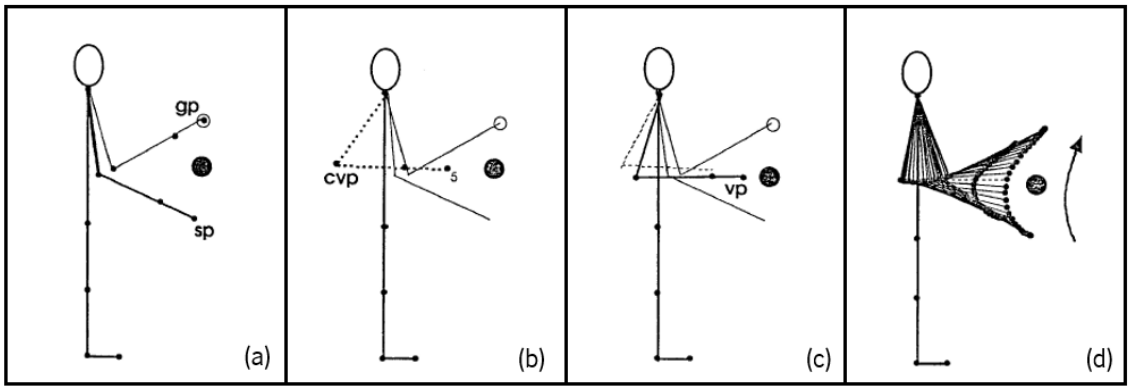


Figure 3.9: Representation of a planning a movement from an initial posture to touch a target when an obstacle is present: **(a)** Initial posture (sp) and final posture (gp) previously determined; **(b)** Most adequate stored posture (cvp) as bounce posture; **(c)** Generated posture (vp) as the bounce posture; **(d)** Movement to be executed between initial and final posture. [9]

[8] Image adapted from Vaughan et al. (2001)
[9] Image adapted from Vaughan et al. (2001)

In reality, the back-and-forth movement is divided into two sub- movements: from the initial position to the bounce posture; from the bounce posture to the starting position.

The most appropriate bounce posture is selected and generated by the same procedure as the final posture, departing from the given constraints (Figures 3.9(c) and 3.9(d)). According to Vaughan et al. (2006), an alternative to the obstacle avoidance procedure of Rosenbaum et al. (2001) is to split the movement into two sub- movements: movement from the initial posture to the bounce posture; movement from the bounce posture to the final posture.

Individual planning of each sub- movement, on the other hand, leads to an unrealistic trajectory characterised by several sudden fluctuations in the movement.

Chapter 4: Robotic Platform Sawyer

This chapter presents the Saywer Robotic Platforms used in the implementation and validation of this dissertation. It will start with some specifications of the robotic manipulator about its structure and characteristics (section 4.1). Then, the kinematic model of the manipulator will be presented (section 4.2), starting with a theoretical background (section 4.2.1), a representation of the robotic arm and its variables (section 4.2.2), and the computation of the direct kinematics (section 4.2.2.1).

4.1 Specifications

The collaborative Sawyer Robot is one of the most innovative and recent robotic platforms currently on the market. Developed by Rethink Robots and marketed since March 2016, Sawyer is designed to provide the industry with high-performance automation essential for performing a wide range of tasks that require high flexibility in workspaces with human operators.

Physically, the Sawyer robot consists of a pedestal (Figure 4.1(b)) and a 7DOF robotic manipulator (Figure 4.1(a)) that allows a maximum reach of 1260 mm and manipulation of objects weighing less than 4 kg. In addition, this platform has a vision system integrated on the head and wrist, which provides a positioning mechanism essential for dynamic reorientation of the manipulator and highly accurate and repeatable movements.

Unlike most industrial robots, Sawyer is controlled simultaneously by torque and joint positions, which monitors the applied force in different directions and ensures a task behavior similar to humans. Currently, this robotic platform does not exhibit autonomy in its movements, as these are executed via a trajectory replication of movements performed by the human operator using the Intera 3.3 software platform.

According to Rethink Robotics, Sawyer is suitable for various industrial applications such as pick and place, plastic injection and blow molding, PCB handling and ICT, testing and quality control, metal sampling and press operation, CNC machine operation, and co-packing and end-of-line packaging. The table 4.1 summarizes the key specifications of the Sawyer robot.

[1]Image taken from website: https://www.active-robots.com.

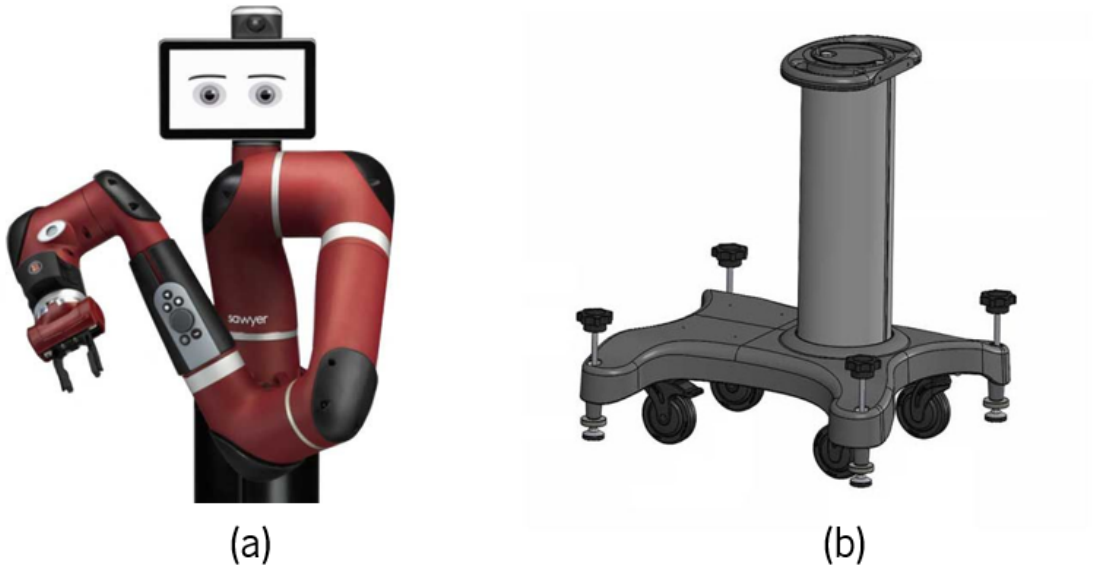


Figure 4.1: Sawyer Robotic Manipulator (a) with its pedestal (b).[1]

4.2 Kinematic Model of the Manipulator

The motion planning method used in this dissertation depends on the computation of the kinematic models of the manipulator's arm and hand. In the next sections, some theoretical concepts related to the kinematic models are introduced and the hardware specifications of the Sawyer robotic manipulator are presented.

Since the motion planning method used does not require inverse kinematics to obtain the joint angles of the manipulator, we will only compute the direct kinematics of the arm. In this method, the set of admissible postures is determined by nonlinear constraint-bound optimization problems, making the motion planning process faster and computationally less expensive.

4.2.1 Kinematics Theory

According to Craig (2005), kinematics is the study of the geometrical and time-based properties of motion, without regard to the forces that cause them. In the case of manipulators described as a set of segments connected by joints, the author says that the kinematic study can be handled by four established parameters for the Denavit-Hartenberg convention:

- a_{i-1}: the distance from $\hat{z}_{i-1}$ to $\hat{z}_i$ measured along $\hat{x}_{i-1}$;
- α_{i-1}: the angle from $\hat{z}_{i-1}$ to $\hat{z}_i$ measured along $\hat{x}_{i-1}$;

Table 4.1: Sawyer Robot Specifications

Degrees of Freedom	7
Maximum Reach	1260 mm
Payload	4 kg
Task repeatability	0.1 mm
Average Speed	1.5 m/s
Power Requirements	100.240VAC, 47-63Hz, 4A Max
Operating Temperature	0°C-40°C, 80% Relative Humidity
Communication	ModBus Remote I/O, PLC
Collaboration	ISSO 10218-1:2011

- d_i: the distance from $\hat{x}_{i-1}$ to $\hat{x}_i$ measured along $\hat{z}_i$;
- θ_i: the angle from $\hat{x}_{i-1}$ to $\hat{x}_i$ measured along $\hat{z}_i$;

The kinematic model can be direct if the position and orientation of the end-effector are represented in joint space, or inverse if they are represented in Cartesian space. As shown in figure 4.2, direct kinematics is the computation of the position and orientation of the end-effector from a set of given joint angles. Conversely, inverse kinematics is the computation of the joint angles by which the end-effector of the manipulator reaches a desired position and orientation. However, the inverse kinematics problem may be impossible to solve or it may have multiple solutions, making it a computationally expensive process.

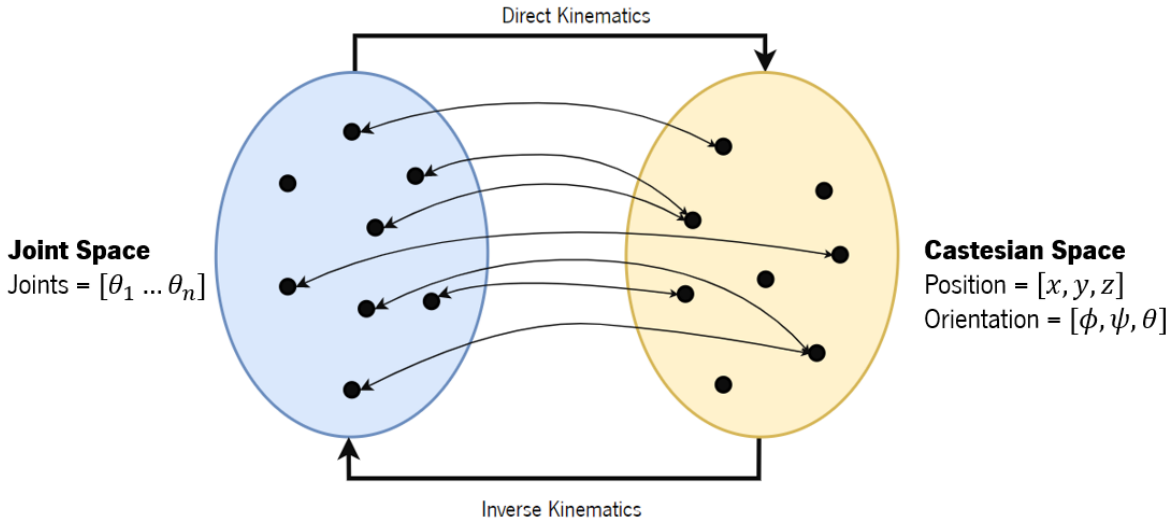


Figure 4.2: Relation between direct and inverse kinematics.[2]

[2]Image adapted from de Sá (2018).

4.2.2 Robotic Arm

The Sawyer robotic platform Sawyer is a robotic manipulator with 7DOFs that can reach and manipulate objects with a payload of less than 4 kg and a maximum reach of 1260 mm.

As can be seen in figure 4.3, the joints of the manipulator can be divided into two different categories according to its motion:

(i) *roll* joints - θ_1, θ_3, θ_5 and θ_7;

(ii) *pitch* joints - θ_2, θ_4 and θ_6.

The maximum limits of each joint can be consulted in table 4.2.

In addition to the joints of the robot manipulator, the figure 4.3 also highlights its segments, whose dimensions are given in the table 4.3, and their respective reference points. These reference points are classified according to a nomenclature adaptation defined for joints and extremities of the human arm: shoulder offset (S_off), shoulder (S), elbow offset (E_off), elbow (E), wrist offset (W_off), wrist (W) and hand (H).

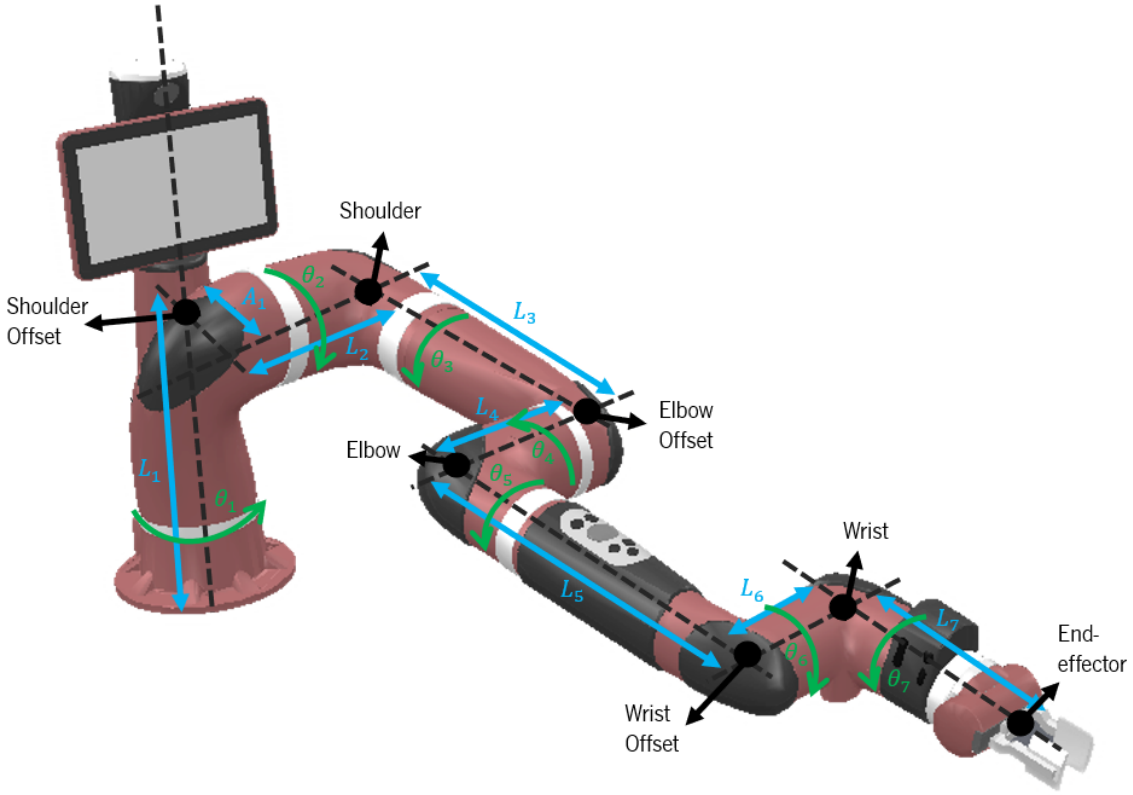


Figure 4.3: Robotic Manipulator Sawyer illustration, highlighting the reference points for the joints and end-effector. The rotations of the joints are shown in yellow and the segments of the robot in blue.[3]

[3]Image adapted from de Sá (2018).

Table 4.2: Joint Limits of the robotic platform Sawyer

Joint	Joint Limits (degrees)	Variation (degrees)
θ_1	[-174,8;174,7]	349,5
θ_2	[-218,8;130,7]	349,5
θ_3	[-174,8;174,9]	349,7
θ_4	[-174,8;174,9]	349,7
θ_5	[-171,0;171,0]	342,0
θ_6	[-171,0;171,0]	342,0
θ_1	[-269,9;269,9]	539,8

Table 4.3: Robotic platform Sawyer segments dimensions.

Segment	Dimension (mm)	Description
L_1	317,0	Distance from origin to θ_1 along z
A_1	81,0	Distance from origin to θ_1 along x
L_2	192,5	Distance from θ_1 to θ_2 along z
L_3	400,0	Distance from θ_2 to θ_3 along z
L_4	168.5	Distance from θ_3 to θ_4 along z
L_5	400,0	Distance from θ_4 to θ_5 along z
L_6	136,3	Distance from θ_5 to θ_6 along z
L_7	265,2	Distance from θ_6 to θ_7 along z

4.2.2.1 Direct Kinematics

When calculating the direct kinematics of the manipulator, the first thing we do is to place the manipulator in the null position, that is, all the joints have a value of zero. Then, we need to assign a coordinate axis ($\hat{x}_i\hat{y}_i\hat{z}_i$) to each arm segment according to the convention Denavit-Hartenberg.

As can be seen in the figure 4.4, in the Sawyer robotic platform, the first pair of coordinate axis ($\hat{x}_0\hat{y}_0\hat{z}_0$) is located in the base of the manipulator. The second and third pairs of coordinate axis ($\hat{x}_1\hat{y}_1\hat{z}_1$ and $\hat{x}_2\hat{y}_2\hat{z}_2$) are represented in the shoulder offset and shoulder. This procedure is repeated for the fourth, fifth, sixth and seventh coordinate axis pairs ($\hat{x}_3\hat{y}_3\hat{z}_3$, $\hat{x}_4\hat{y}_4\hat{z}_4$, $\hat{x}_5\hat{y}_5\hat{z}_5$ and $\hat{x}_6\hat{y}_6\hat{z}_6$) represented by the elbow offset, the elbow, the wrist offset and the wrist respectively. The last is the eighth pair of coordinate axis ($\hat{x}_7\hat{y}_7\hat{z}_7$) represented by the hand. The table 4.4 shows the Denavit-Hartenberg parameters for each joint of the manipulator.

[4]Image adapted from de Sá (2018).

Table 4.4: Denavit-Hartenberg parameters for the robotic platform Sawyer.

i	α_{i-1} **(degrees)**	a_{i-1} **(mm)**	d_i **(mm)**	θ_i **(degrees)**
1	0	0	L_1	θ_1
2	-90	A_1	L_2	θ_2 - 90
3	-90	0	L_3	θ_3
4	90	0	L_4	θ_4
5	-90	0	L_5	θ_5
6	90	0	L_6	θ_6
7	-90	0	L_7	θ_7

With these Denavit-Hartenberg parameters, it is possible to calculate the transformation between the different joints of the manipulator using the general expression 4.1.

$$^{i-1}T_i = Rot_x(\alpha_{i-1})Trans_x(a_{i-1})Trans_z(d_i)Rot_z(\theta_i)$$

$$= \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) & 0 & a_{i-1} \\ \sin(\theta_i)\cos(\alpha_{i-1}) & \cos(\theta_i)\cos(\alpha_{i-1}) & -\sin(\alpha_{i-1}) & -\sin(\alpha_{i-1})d_i \\ \sin(\theta_i)\sin(\alpha_{i-1}) & \cos(\theta_i)\sin(\alpha_{i-1}) & \cos(\alpha_{i-1}) & \cos(\alpha_{i-1})d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

Then we use the first row of the table 4.4 to compute the transformation between the base and the first joint of the manipulator defined by the matrix 4.2.

$$^{0}T_1 = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 & 0 \\ \sin(\theta_1) & \cos(\theta_1) & 0 & 0 \\ 0 & 0 & 1 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.2}$$

We then proceed to do the same for the remaining rows of the table 4.4 The transformations between

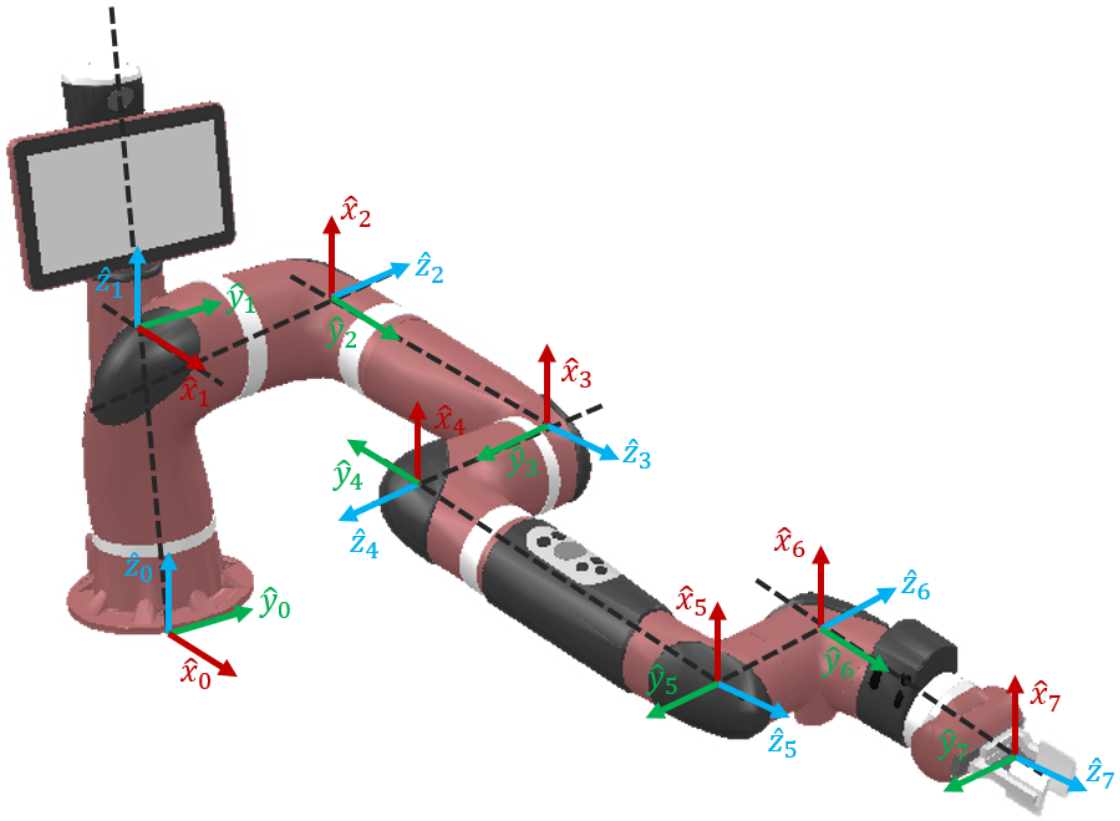


Figure 4.4: Robotic manipulator Sawyer in its null position and representation of coordinate axis pairs ($\hat{x}_i\hat{y}_i\hat{z}_i$) by convention Denavit-Hartenberg.[4]

the first and second joints to the sixth and seventh joints defined by the matrices (4.3) to (4.8).

$$ {}^{1}T_2 = \begin{bmatrix} \cos(\theta_2 - 90) & -\sin(\theta_2 - 90) & 0 & A_1 \\ 0 & 0 & 1 & L_2 \\ -\sin(\theta_2 - 90) & -\cos(\theta_2 - 90) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.3} $$

$$ {}^{2}T_3 = \begin{bmatrix} \cos(\theta_3) & -\sin(\theta_3) & 0 & 0 \\ 0 & 0 & 1 & L_3 \\ -\sin(\theta_3) & -\cos(\theta_3) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4} $$

$$ {}^{3}T_4 = \begin{bmatrix} \cos(\theta_4) & -\sin(\theta_4) & 0 & 0 \\ 0 & 0 & 1 & L_4 \\ -\sin(\theta_4) & -\cos(\theta_4) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.5} $$

$$ {}^{4}T_{5} = \begin{bmatrix} \cos(\theta_5) & -\sin(\theta_5) & 0 & 0 \\ 0 & 0 & 1 & L_5 \\ -\sin(\theta_5) & -\cos(\theta_5) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.6} $$

$$ {}^{5}T_{6} = \begin{bmatrix} \cos(\theta_6) & -\sin(\theta_6) & 0 & 0 \\ 0 & 0 & -1 & -L_6 \\ \sin(\theta_6) & \cos(\theta_6) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.7} $$

$$ {}^{6}T_{7} = \begin{bmatrix} \cos(\theta_7) & -\sin(\theta_7) & 0 & 0 \\ 0 & 0 & 1 & L_7 \\ -\sin(\theta_7) & -\cos(\theta_7) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.8} $$

Multiplying all the homogeneous transformation matrices presented before, we obtain the global transformation matrix (matrix 4.9) which represents the transformation from the base to the hand of the manipulator.

$$ {}^{0}T_{H} = {}^{0}T_{7} = {}^{0}T_{1} \cdot {}^{1}T_{2} \cdot {}^{2}T_{3} \cdot {}^{3}T_{4} \cdot {}^{4}T_{5} \cdot {}^{5}T_{6} \cdot {}^{6}T_{7} \tag{4.9} $$

However, via the described procedure it is also possible to calculate the remaining points of the manipulator referenced to the base (equations 4.10 to 4.15).

$$ {}^{0}T_{S_{off}} = {}^{0}T_{1} \tag{4.10} $$

$$ {}^{0}T_{E_{off}} = {}^{0}T_{3} \tag{4.11} $$

$$ {}^{0}T_{W_{off}} = {}^{0}T_{5} \tag{4.12} $$

$$ {}^{0}T_{S} = {}^{0}T_{2} \tag{4.13} $$

$$ {}^{0}T_{E} = {}^{0}T_{4} \tag{4.14} $$

$$ {}^{0}T_{W} = {}^{0}T_{6} \tag{4.15} $$

Using the homogeneous transformation matrices computed by the general equation 4.1, we can obtain

the orientation (ϕ, ψ, θ) and the position (x, y, z) of the various reference points of the arm with respect to a given axis, as can be seen in matrix (4.16). Then, using the global transformation matrix, we can compute the orientation (ϕ_H, ψ_H, θ_H) and position (x_H, y_H, z_H) from hand to base.

$$ {}^{i-1}T_i = \left[\begin{array}{c|c} Rot_{3\times3}(\phi,\psi,\theta) & Trans_{3\times1}(x,y,z) \\ \hline 0_{1\times3} & 1 \end{array}\right] \tag{4.16} $$

Chapter 5: Motion Planning

This chapter will describe the motion planning method deployed in this dissertation. As previously mentioned, this method relies on the transfer of human motor control knowledge to the robotics area and follows the presentation of in Silva (2011). (de Sá (2018), Gulletta (2021)).

In section 5.1 will introduce a few concepts to take in consideration regarding the motion planner deployed (HUMP) and also mathematically formalize the non-linear optimization problem, generalizing the final and bounce posture problems as well as the trajectory selection. Afterward, section 5.2 will describe and classify the different types of motion to be executed by the manipulator during a task, as well as the calculus that is involved in all of the different stages of a pick & place movement. Section 5.3 will explain the motion planning strategy adopted in this dissertation, especially the implemented mechanism to avoid object collision. Next, section 5.4 will present the robots' model of the body, manipulator, and the models of the existing objects in the workspace. These models are used to define constraints to optimization problems, related to the obstacle avoidance mechanisms.Finally, section 5.5 will present optimization software used in problem resolutions previously mentioned.

5.1 HUMP - Overview

To accomplish a simple task such as reaching and grasping a given object, the manipulator may have numerous trajectories that take into account different time and velocity requirements. In this dissertation, a motion planning method called Human-like upper-limb motion planner (**HUMP**) is used (Gulletta (2021); Gulletta et al. (2015) Gulletta et al. (2021). The HUMP borrows from the Posture-based Motion Planner model by Rosenbaum et al. (2001), which is used to generate human-like movements in grasping and manipulation tasks. By selecting a final posture and then generating a collision-free trajectory, a motor redundancy problem is overcome. HUMP also has an obstacle avoidance mechanism based on the concept of bounce posture. By superimposing (i) a direct movement from the start to a final posture and (ii) a back-and-forth movement from the start to a bounce posture and back, we can achieve a human-like motion for obstacle avoidance (Section 3.2.2). Although based directly on the Rosenbaum et al. (2001) model, some adaptations were made to support fluid and efficient human-robot interactions, such as no explicit inverse kinematics, which can lead to a lack of generalization and is computationally expensive,

and integrating sequential motion phases, which may require high endpoint accuracy.

In figure 5.1 we see a simplification of how the motion planner works, which begins with the selection of a suitable final posture.

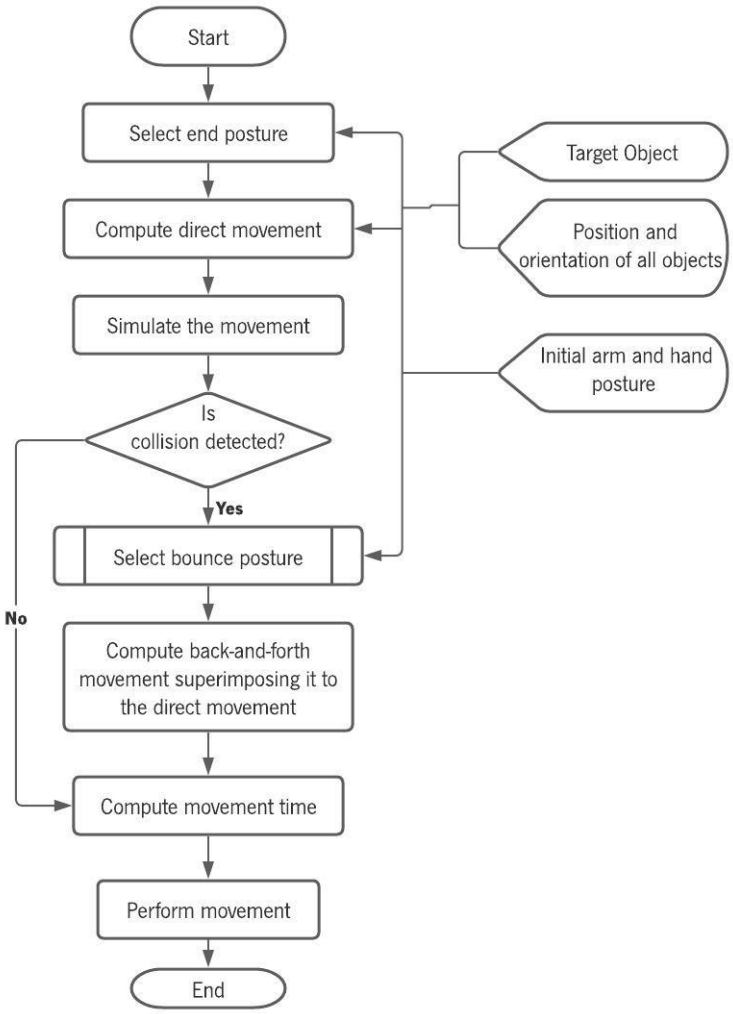


Figure 5.1: Flowchart of motion planning.[1]

To select this final posture, the initial posture of the hand and arm must be taken into account, as well as the information provided by the vision system about the objects present in the workspace. Then, for each joint, the trajectory is calculated, as well as the time history of position, velocity and acceleration from the initial posture to the final posture. This calculated trajectory defines the direct motion without considering collisions. To detect collisions with objects in the workspace, the planning system uses direct kinematics to internally simulate the motion from start to finish. If no collisions are detected, the motion is executed, but if they are detected, the system searches for a feasible motion. To find this feasible

[1] Flowchart adapted from Silva (2011).

movement, a suitable bounce posture must be chosen. The final position to be reached is the same as for the direct movement, only the way to get there is different to ensure collision avoidance.

To select the final posture, both the final arm posture and the final hand posture must be calculated. The final hand posture depends on the hardware of the robotic hand. The cost of moving the manipulator from its initial position to the selected final position is given by the equation (5.1):

$$C(\theta_{f,arm}) = \sum_{k=1}^{n} \lambda_k (\theta_{k,0} - \theta_{k,f})^2, \lambda_k \geq 0 \tag{5.1}$$

Where **n** is the number of DOFs of the robotic arm and λ_k are the joint cost factors. In addition to this objective function, there are also some constraints that establish relationships between the manipulator and the target, as well as constraints that prevent collision with obstacles and ensure that the joints do not exceed their limits.

Once the final posture is selected, the trajectory must be generated. First, the direct movement is calculated according to the following equation (5.2).

$$\mathcal{T}(t; \theta_0, \theta_f) = \theta_0 + \mathcal{T}_{direct}(t; \theta_0, \theta_f) \tag{5.2}$$

Where $\mathcal{T}_{direct}$ represents the direct movement. To avoid collisions, the bounce posture serves as a subgoal of the back-and-forth movement which can be calculated using the equation (5.3).

$$\mathcal{T}_{bkfor}(t; \theta_0; \theta_b) = \frac{\tau(1-\tau)}{t_b(1-t_b)}(\theta_b - \theta_0) \sin^2(\pi\tau^{\vartheta}) \tag{5.3}$$

Where $\tau = t/T$ is the normalized time, where T is the duration of the movement. The complete trajectory can be calculated from the composition of the direct movement and back-and-forth movement given by equation (5.4).

$$\mathcal{T}(t; \theta_0, \theta_f, \theta_b) = \theta_0 + \mathcal{T}_{direct}(t; \theta_0, \theta_f) + \mathcal{T}_{bkfor}(t; \theta_0, \theta_b) \tag{5.4}$$

To calculate the back-and-forth movement, the bounce posture must be calculated, a process very similar to the selection of the final posture. The bounce posture can be calculated using the equation (5.5).

$$C(\theta_b) = \sum_{k=1}^{n_j} \lambda_k (\theta_{k,0} - \theta_{k,b})^2, \lambda_k \geq 0 \tag{5.5}$$

Where n_j is the number of joints involved in the optimization problem and $\lambda_k \geq 0$ is the cost factor

for each manipulator joint. With this objective function, there is also a set of constraints that ensure that the manipulator joint limits are respected and that there are no collisions between the manipulator, the robot's body, the objects in the workspace, and the object to be grasped.

More details about can be found in Gulletta (2021).

5.2 Trajectory Generation in Sequential Phases

5.2.1 Classification and segmentation of a movement

Common object manipulation tasks can be divided into relevant action phases, such as approaching an object, grasping, lifting from a surface, moving, and repositioning. Different sensory events, such as making or breaking contact between the manipulator and an object or between an object and a surface, define the transition between the different phases. Temporal task segmentation indicates that a selected final posture of a particular motor function becomes the initial posture of the subsequent motor primitive in the posture-based motion planning model (Rosenbaum et al. (1995)). In Figure 5.2 it can be seen the temporal segmentation of a pick & place task into different phases of an upper-limb movement (Gulletta (2021)).

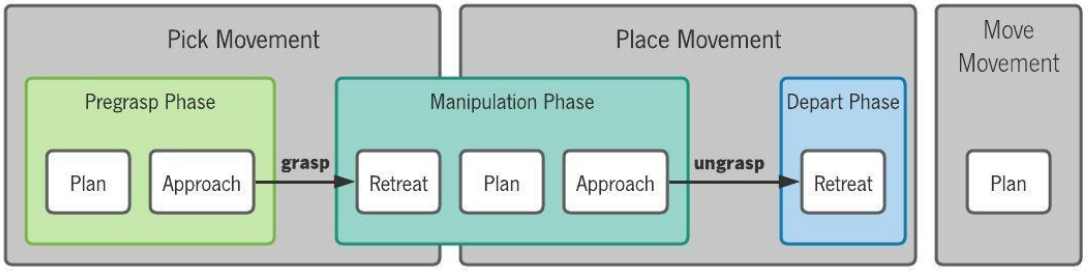


Figure 5.2: States transition diagram of pick and place and move movements.[2]

As we can see in figure 5.2 the movement can be divided into 3 different phases: Pick, Place and Move. Normally, these movements consist of the 3 stages (Plan, Approach and Retreat), except for the Move phase, which consists only of the Plan stage. The plan stage is responsible for bringing the end-effector close to the final posture. In the Approach stage, the end-effector is moved to the actual final posture. The retreat phase, the end-effector moves away from the final posture. Thus, the complete movement is performed as follows: the plan stage of the pick movement brings the end-effector close to the object to be grasped, the approach phase brings the hand into the grasping posture and actually grabs

[2]Flowchart adapted from Gulletta (2021).

the object (grasp), completing the pre-grasping phase. The manipulation phase begins with the retreat stage, which lifts the grasped object, and the place movement begins, where again the plan stage brings the object close to the desired position, the approach phase brings the object to the actual final position and releases it (ungrasp). After the object is released, the departure phase begins, which moves the end-effector away from the object and completes the Place movement. In addition, there are also movements in which no objects are manipulated, which consist only of the plan stage, in which the end-effector is merely moved to a desired position. A diagram of this complete movement is shown in Figure 5.3.

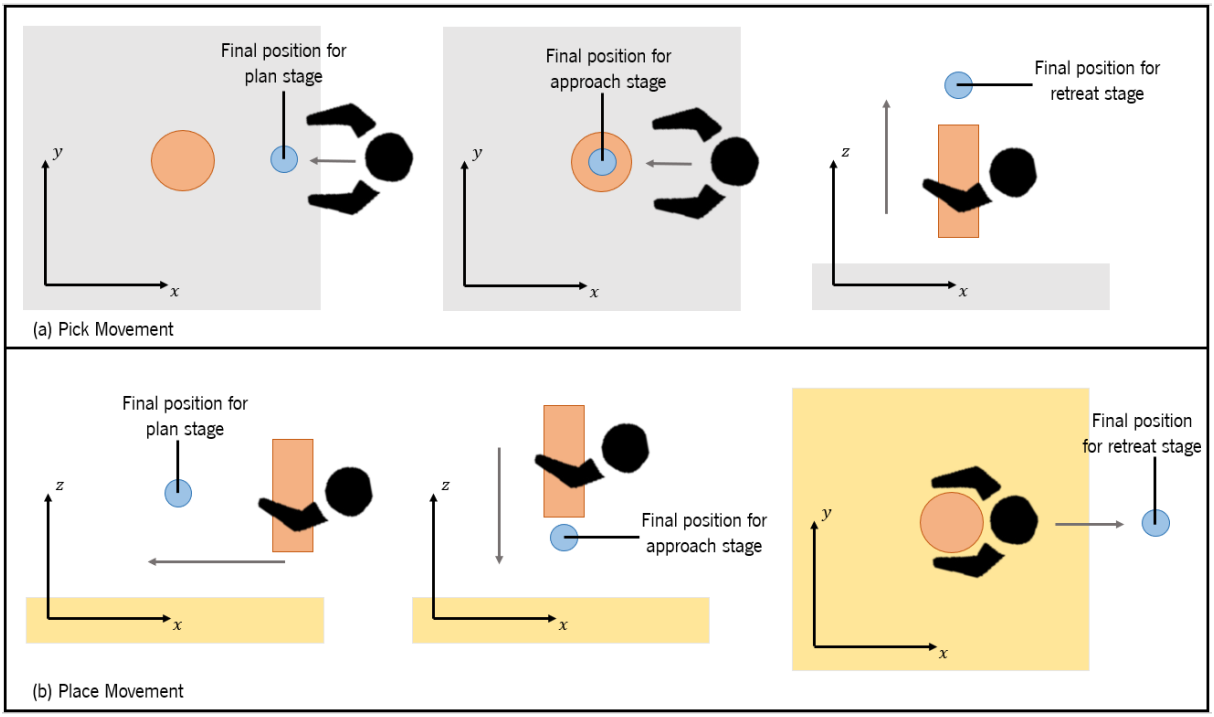


Figure 5.3: Diagram of the different stages of a pick & place movement where the orange cylinder is the object to be grasp, the gray are is where the object is to be grasped and the yellow are is the area where the object is to be placed. (a) Representation of the pick movement; (b) Representation of the place movement.[3]

5.2.2 Transport Phase

The transport final posture is a posture $\theta_{tran} = (\theta_{tran,arm}, \theta_{tran,hand})$, where $\theta_{tran,arm}$ is the arm final posture and $\theta_{tran,hand}$ is the hand final posture. The $\theta_{tran,arm}$ is selected as explained in Gulletta (2021) for a given transport final hand pose. The next sections will present the selection of the $\theta_{tran,hand}$ and the composition of the trajectory in the transport phase.

[3] Flowchart adapted from de Sá (2018).

Hand Transport Final Posture Selection For pick movements, the **hand transport final posture** for grasping is calculated considering the inverse kinematics of the gripper. When an approach phase follows, the values of $\theta_{tran,hand}$ are modified to obtain a slightly larger opening of the fingers than is necessary. This is done to emulate prehension and ensure a successful transition to grasp. In place movements, the **hand transport final posture** is limited because the object being manipulated must be held by the end-effector. Therefore, the **hand transport final posture** is equal to the **hand start posture**, i.e. $\theta_{tran,hand} = \theta_{0,hand}$. The move movement is characterized only by a transport phase of the hand, which is not directed towards objects. Therefore, the joint angle values in $\theta_{tran,hand}$ can be set randomly.

Transport direct movement The direct movement in the transport phase consists of a trajectory generated according to the principle of minimum angular jerk. The trajectories of the end-effector with a bell-shaped velocity profile are not explicitly represented, but result from the joint displacements (Rosenbaum et al. (1995)). The direct movement is calculated by minimizing the angular jerk during the course of the movement (equation (5.6)).

$$C_{dm}(\theta) = \frac{1}{2}\int_0^T \sum_{k=1}^{n} \left(\frac{d^3\theta_k(t)}{dt^3}\right)^2 dt \tag{5.6}$$

Where n is the number of joints and $\theta_k(t)$ is the k^{th} joint angle position at the instant t. The solution to the variational problem is a polynomial of order $5th$ whose coefficients can be calculated by applying boundary conditions to position, velocity, and acceleration. The direct transport movement given by equation (5.7) is the solution. (Gulletta (2021)).

$$\begin{aligned}\mathcal{T}_{tran}^{direct}(t;\theta_{tran}) = (\theta_{tran}-\theta_0)(10\tau^3-15\tau^4+6\tau^5)\\ +\omega_0 T(\tau-6\tau^3+8\tau^4-3\tau^5)+\omega_f T(-4\tau^3+7\tau^4-3\tau^5)\\ +\frac{1}{2}\alpha_0 T^2(\tau^2-3\tau^3+3\tau^4-\tau^5)+\frac{1}{2}\alpha_f T^2(\tau^3-2\tau^4+\tau^5)\end{aligned} \tag{5.7}$$

Where $\theta_0 = (\theta_{0,1},...,\theta_{0,n})^T$ is the vector with initial joint angles ($t = 0$), $\theta_{tran} = (\theta_{tran,1},...,\theta_{tran,n})^T$ is the vector with target joint angles $(t = T)$ and $\tau = t/T$ is the normalized time. ω_0 and ω_f are the initial and final joint velocities, while α_0 and α_f are the initial and final joint accelerations, respectively. To ensure that the joint angles are within their mechanical limits thoughout the entire movement we use equation (5.7) since the values of the transport final posture θ_{tran} are within their allowable range and

equation (5.7) is an evenly increasing or decreasing function that produces values from $\theta_{0,k}$ to $\theta_{tran,k}$ of each k^{th} joint. Thus, since both the initial and final postures are within the constraint limits of the joints, this also occurs throughout the trajectory. The angular velocities and accelerations of the transport phase can be calculated by deriving the equation (5.7). The movement of the joints follows a straight-line path in joint space with a bell-shaped velocity profile. In the task space, this can result in a straight trajectory or a slightly curved trajectory, both with bell-shaped velocity profiles, which is common with human upper-limb movements (Rosenbaum et al. (2001)). In this method, it is not necessary to explicitly model the bell-shaped velocity profile of the end-effector, but it results from the displacements of the joints.

Transport back-and-forth movement Next we must calculate the back-and-forth movement that is given by equation (5.3). This must avoid collisions with obstacles in the workspace, the robot's body, and, in the case of pick movements, with the object to be grasped, throughout the transport phase. During move movements, the bounce hand posture, $\theta_{b,hand}$, is selected by solving the optimization problem. For place movements, the object to be manipulated must be held by the end-effector throughout the transport phase. Therefore, the bounce hand posture is equal to the initial hand posture, i.e. $\theta_{b,hand} = \theta_{0,hand}$. This means that the contribution of the fingers to the back-and-forth motion in equation (5.4) is zero. For pick movements, the choice of the bounce hand posture $\theta_{b,hand}$ depends on the specific hardware of the robot hand.

Composite movement of the Transport Phase In the transport phase, the trajectory of each joint is obtained by superimposing the back-and-forth motion on the direct transport movement and is given by equation (5.8).

$$\mathcal{T}_{tran}(t, \theta_{tran}, \theta_b) = \theta_0 + T_{tran}^{direct}(t, \theta_{tran}) + T_{tran}^{bkfor}(t, \theta_b) \tag{5.8}$$

Where $\mathcal{T}_{tran}^{direct}$ and $\mathcal{T}_{tran}^{bkfor}$ are given by equation (5.7) and equation (5.3), respectively. Equation (5.8) provides a suitable and human-like collision-free trajectory with continuous angular velocity and acceleration. The joint boundaries are respected, and by setting the movement time T accordingly, the maximum allowable angular velocity and joint acceleration of each joint are never reached.

Velocities and accelerations in the transport phase are given by the equations (5.9) and (5.10) which can be obtained by deriving the equation (5.8) (Gulletta (2021)).

$$\begin{aligned}\dot{\mathcal{T}}_{tran}(t;\theta_{tran},\theta_b) = &\frac{30}{T}(\theta_{tran}-\theta_0)(\tau^2-2\tau^3+\tau^4)\\ &+\omega_0(1-18\tau^2+32\tau^3-15\tau^4)+\omega_{tar}(-12\tau^2-28\tau^3-15\tau^4)\\ &+\frac{1}{2}\alpha_0 T(2\tau-9\tau^2+12\tau^3-5\tau^4)+\frac{1}{2}\alpha_t arT(3\tau^2-8\tau^3+5\tau^4)\\ &+\frac{1}{Tt_b(1-t_b)}(\theta_b-\theta_0)\cdot\left[(1-2\tau)\sin^2(\pi\tau^\vartheta)+(1-\tau)\pi\vartheta\tau^\vartheta\sin(2\pi\tau^\vartheta)\right]\end{aligned} \quad (5.9)$$

$$\begin{aligned}\ddot{\mathcal{T}}_{tran}(t;\theta_{tran},\theta_b) = &\frac{60}{T^2}(\theta_{tran}-\theta_0)(\tau-3\tau^2+2\tau^3)\\ &+\frac{12}{T}\omega_0(-3\tau+8\tau^2-5\tau^3)+\frac{12}{T}\omega_{tar}(-2\tau+7\tau^2-5\tau^3)\\ &+\alpha_0(1-9\tau+18\tau^2-10\tau^3)+\alpha_{tar}(-3\tau-12\tau^2+10\tau^3)\\ &+\frac{2}{T^2t_b(1-t_b)}(\theta_b-\theta_0)\left[-\sin^2(\pi\tau^\vartheta)+\pi^2\vartheta^2(1-\tau)\tau^{2\vartheta-1}\cos(2\pi\tau^\vartheta)\right.\\ &\left.+(1-2\tau)\pi\vartheta\tau^{\vartheta-1}\sin(2\pi\tau^\vartheta)\right]\end{aligned} \quad (5.10)$$

5.2.3 Approach Phase

The transport phase is followed by the approach phase. The approach final posture is a posture $\theta_{app} = (\theta_{app,arm}, \theta_{app,hand})$, where $\theta_{app,arm}$ is the arm final posture and $\theta_{app,hand}$ is the hand final posture. The selection of $\theta_{app,arm}$ is done as explained in Gulletta (2021) for a given final hand posture of the approach phase.

Hand Approach Final Posture Selection The final posture of the hand, $\theta_{app,hand}$, is calculated differently for pick and place movements. For pick movements, the transport phase brings the hand close to the object to be grasped, given appropriate preformation of the fingers. Then, the approach phase brings the hand into the grasping posture and completes it. The values of the joints in the fingers are calculated according to the size of the object to be grasped. For place movements, the transport phase transports the grasped object to the immediate vicinity of its final position. Then, the approach phase completes the positioning of the object in its final pose. During this phase, the final posture of the hand remains constant in order to securely hold the object to be placed, i.e. $\theta_{app,hand} = \theta_{0,hand}$.

Approach Direct Movement Since it is assumed that there are no obstacles near the target, the transport phase brings the hand close to the target position with a reasonable and collision-free trajectory. However, since the paths to the final posture may be severely obstructed, there may be no solution to the planning problem. Therefore, in the approach phase, a direct movement is sufficient to obtain a collision-free trajectory. In this phase, the manipulator is moved from the final posture of the transport phase, θ_{tran}, (which is now the initial posture for the approach phase, θ_0) to the approach final posture, θ_{app}. The formulation of the direct approach movement, $\mathcal{T}_{app}(t;\theta_{app})$ is given by the equations (5.11) - (5.13) (Gulletta (2021)).

$$\mathcal{T}_{app}(t;\theta_{app}) = \theta_0 + \frac{1}{4}(\theta_{app} - \theta_0)(5\tau - \tau^5) \tag{5.11}$$

$$\dot{\mathcal{T}}_{app}(t;\theta_{app}) = \omega_0(1 - \tau^4) \tag{5.12}$$

$$\ddot{\mathcal{T}}_{app}(t;\theta_{app}) = -\alpha_0\tau^3 \tag{5.13}$$

Where $\omega_0 = \frac{5}{4T}(\theta_{app} - \theta_0), \alpha_0 = \frac{4\omega_0}{T}$, where T is the total duration of the approach phase and $\tau = t/T$ is the normalized time. The trajectory of the joints during the approach phase (equation (5.11)) is a uniformly increasing function if $(\theta_{app,k} - \theta_{0,k})$ is positive, or a decreasing function, when $(\theta_{app,k} - \theta_{0,k})$ is negative, resulting in values between $\theta_{k,0}$ and $\theta_{k,app}$ for each k^{th} joint. As a result, the overall approach direct movement respects the physical constraints of the joints, since both the initial and final postures are within their allowable range. The velocity and acceleration of the approach phase are modelled by the equations (5.12) and (5.13), respectively. The manipulator stops at the end of the approach phase to initiate the grasp/ungrasp transition, as shown in equation (5.12). At the beginning of the approach phase, the joints have a velocity ω_0 and an acceleration α_0 that must be equal to the velocity ω_{tar} and acceleration α_{tar} of the transport phase movement in equation (5.8). This implementation ensures that the transport and approach phases are properly composed.

5.2.4 Retreat Phase

The approach phase is followed by the retreat phase. The retreat target posture is a posture $\theta_{ret} = (\theta_{ret,arm}, \theta_{ret,hand})$, where $\theta_{ret,arm}$ is the arm final posture and $\theta_{ret,hand}$ is the hand final posture. The selection of the final posture for the retreat phase, $\theta_{ret,arm}$, is done as explained in Gulletta (2021) for a given retreat final hand posture.

Hand Retreat Final Posture Selection For pick and place movements, the final posture of the hand, $\theta_{ret,hand}$, is calculated differently. For pick movements, the final posture of the hand for the retreat phase is restricted because the object is held by the hand. Therefore, the retreat hand final posture is equal to the retreat hand initial posture, i.e. $\theta_{ret,hand} = \theta_{0,hand}$. For place movements in order to ungrasp successfully, the values for the final posture of the retreat hand final posture are calculated taking into account a slightly increased opening of the fingers with respect to the size of the object to be released.

Retreat Direct Movement The retreat phase, like the approach phase, is defined by a direct movement that moves the joints from their approach phase target (initial posture, θ_0) to the current final posture, i.e. the retreat final posture, θ_{ret}. For the same reasons of problem solvability as for the approach direct movement, it is reasonable to assume an obstacle-free region near the current target. The formulation replicates the increasing velocity of the joints until the retreat final posture is reached, since the retreat phase is the beginning of a transport phase in the next movement and is given by the equations (5.14) - (5.16) (Gulletta (2021)).

$$\mathcal{T}_{ret}(t;\theta_{ret}) = \theta_0 + \frac{1}{3}(\theta_{ret} - \theta_0)(5\tau^4 - 2\tau^5) \tag{5.14}$$

$$\dot{\mathcal{T}}_{ret}(t;\theta_{ret}) = \omega_{tar}(2\tau^3 - \tau^4) \tag{5.15}$$

$$\ddot{\mathcal{T}}_{ret}(t;\theta_{ret}) = \alpha_{tar}(3\tau^2 - 2\tau^3) \tag{5.16}$$

The trajectory of the joints in equation (5.14) is a uniformly increasing or decreasing trajectory according to the sign of $(\theta_{ret,k} - \theta_{0,k})$ for each k^{th} joint. Equation (5.14) ensures that the values of the trajectory remain within their physical limits, since θ_0 and θ_{ret} are within the allowable range of the joints. The velocity and acceleration of the retreat phase are modelled using equations (5.15) and (5.16). Immediately after the grasp/ungrasp transition, the manipulator begins with zero velocity and acceleration. The retreat phase ends with velocity ω_{tar} and acceleration α_{tar}, which must be specified as initial boundary conditions for the transport phase of the following movement. This option guarantees that the retreat phase and the subsequent transport phase are correctly composed.

5.3 Planning Method

As mentioned earlier, the HUMP is based on the Posture-based motion planning method of Rosenbaum et al. (2001) and the movement to be performed is planned in relation to the different stages described

in the previous section, as shown in the flowchart in Figure 5.2. At the beginning of the motion planning process, information about the manipulator, the workspace, and the task is determined, such as the initial and final joint velocities and accelerations, the position of obstacles in the workspace, the final position of each movement stage, and the type of grip that will be used for picking and placing. The plan stage begins with the selection of the most appropriate final posture for the task. This position is selected using a nonlinear optimization problem in which the joint position with the lowest cost must be selected. The minimum jerk criterion with respect to the maximum limit of the joints is used to define this motion. The obstacle avoidance method exists only in the plan stage since it assumes that the remaining stages are performed in obstacle-free areas given by the proximity of the target positions. The user can choose the direction and the distance to travel during the execution of the stage, but only provided that the movement is properly planned and has at least one obstacle-free path. The selection of the most suitable final posture and the definition of the direct movement begin the planning process in the approach and retreat stages. However, there are several elements associated with establishing a direct movement that are critical to ensuring that the trajectory between the various stages is smooth and fluid.

It is worth mentioning that the velocity of the hand gradually decreases during the approach phase until it reaches the target position. The values for joint velocity and acceleration at the beginning of this phase must match the values at the end of the plan stage to ensure a smooth movement. The same procedure applies to the retreat stage where the values for joint velocity and acceleration at the end must match the user-defined values at the beginning of the manipulator's movement. In the final planning stage, the compound movement is defined and the parameterization of the time for each step is performed once the movements to be performed during the proposed task are defined.

In previous research, the described planning technique has been applied only to anthropomorphic robotic platforms such as ARoS (Bicho et al. (2011); Gulletta et al. (2015); Silva (2011) Gulletta et al. (2021)). In the present work, the procedures for the selection of the final and bounce postures were adopted from the dissertation by de Sá (2018), in which the author has already worked on the generalization of the procedures for the selection of the final and bounce postures for the collaborative robot Sawyer. The trajectory to be executed is effectively selected for both manipulators using the approach developed in Silva (2011). The modeling of the robot and the objects in the workspace (Section 5.4), as well as nonlinear optimization problems with constraints, are explained in the following sections.

[4]Flowchart adapted from de Sá (2018)

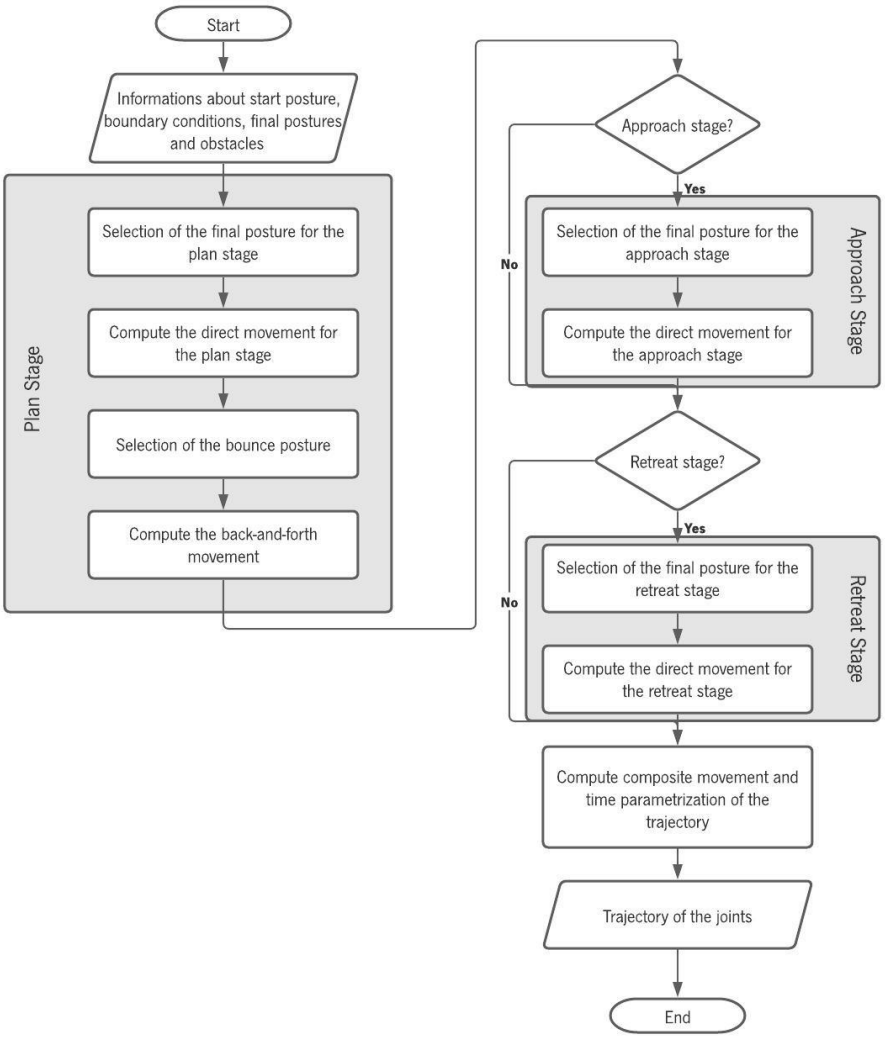


Figure 5.4: Flowchart of the HUMP method.[4]

5.4 Modelling of the Scenario

When selecting the final and bounce postures, certain limits are taken into account to avoid collisions of the manipulator with the robot's body and with obstacles in the workspace. To avoid collisions, the motion planner must be told how to understand the context in which a task is to be performed. The models used to represent the robot and obstacles in the workspace are shown in the next sections.

5.4.1 Model of Body of the Robot

The body of the collaborative Sawyer robot is modelled by a superellipsoid due to its body consisting only of a base. This square surface is described by the equation (5.17) in which $a, b, c \in \Re^+$ correspond to the dimensions of the body in the x, y, and z axes. The parameters $\varepsilon_1, \varepsilon_2 \in \Re^+$ determine the shape

of the superellipsoid in the parallel and perpendicular intersections with the xy plane.

$$\left(\left(\frac{x}{a}^{\frac{2}{\varepsilon_2}}\right)+\left(\frac{y}{b}^{\frac{2}{\varepsilon_2}}\right)\right)^{\frac{\varepsilon_2}{\varepsilon_1}}+\left(\frac{z}{c}\right)^{\frac{2}{\varepsilon_1}}\leq 1 \tag{5.17}$$

To represent the body of the robot by a superellipsoid, it is first necessary to define the parameter values of $\varepsilon_1, \varepsilon_2$. Considering the different types of manipulators that exist in the market and the work developed by Silva (2011), in this dissertation we will consider that $\varepsilon_1 = 0.1$ and $\varepsilon_2 = 1$. As can be seen in figure (5.5), the body of the Sawyer is further represented by an elliptic cylinder with dimension of the z-axis.

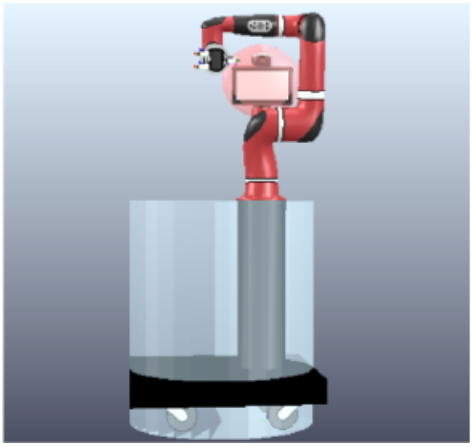


Figure 5.5: The body of the robot is modelled by a superellipsoid (blue).[5]

However, the representation of the body does not include the head of the robot, therefore the head will be considered an object to be avoided in the trajectory generation.

5.4.2 Model of the Arm of the Robot

The robotic arm is represented by a series of spheres with different diameters in the posture selection problems. These spheres are determined by direct kinematics and correspond to the main points of the manipulator that are more prone to collisions. The number of spheres required to model a robotic manipulator depends on the type and dimensions of the segments. A robotic manipulator with offsets in the shoulder, elbow, and wrist joints (θ_2, θ_4, and θ_6) can be represented by a maximum of 14 spheres, while a robotic manipulator without offsets in any of its joints can be represented by a maximum of 8 spheres, depending on the dimensions of its segments. These spheres can be calculated using the equations of

[5]Image adapted from de Sá (2018).

Table 5.1. The most collision-prone locations are determined using an algorithm based on the d_i of the Denavit-Hartenberg convention to generalize the proposed model for all types of manipulators with 7DOFs.

Table 5.1: Points of the arm defined for a manipulator with offsets on the shoulder (θ_2), elbow (θ_4) and wrist (θ_6) joints, modeled by a maximum of 14 spheres.

Sphere	**Arm Point**	**Description**
P_1	$\frac{P_{Base}+P_{S_{off}}}{2}$	Midpoint between the base and shoulder offset
P_2	$P_{S_{off}}$	Shoulder offset
P_3	$\frac{P_{S_{off}}+P_S}{2}$	Midpoint between the shoulder offset and the shoulder
P_4	P_S	Shoulder
P_5	$\frac{P_S+P_{E_{off}}}{2}$	Midpoint between the shoulder and the elbow offset
P_6	$P_{E_{off}}$	Elbow offset
P_7	$\frac{P_{E_{off}}+P_E}{2}$	Midpoint between the elbow offset and the elbow
P_8	P_E	Elbow
P_9	$\frac{P_E+P_{W_{off}}}{2}$	Midpoint between the elbow and the wrist offset
P_{10}	$P_{W_{off}}$	Wrist offset
P_{11}	$\frac{P_{W_{off}}+P_W}{2}$	Midpoint between the wrist offset and the wrist
P_{12}	P_W	Wrist
P_{13}	$P_W + 0,45 \times (P_H - P_W)$	Point between the wrist and the palm of the hand
P_{14}	$P_W + 0,75 \times (P_H - P_W)$	Point between the wrist and the palm of the hand

To determine the arm model, it is necessary to capture the size of the spheres, which is determined a priori by the user based on the real dimensions of the robot and the type of motion planned. This approach describes the spheres that must be used to model the arm and their placement. The arm is modelled in three steps based on the given information, with the shoulder, elbow, and wrist represented on the robotic platform. Depending on the d_i values, P_1 and P_3 spheres can be represented for the shoulder, P_5 and P_7 for the elbow, and P_9 and P_11 for the wrist.

Considering this method, the typology and dimensions of the collaborative robot Sawyer, it can be represented by 10 spheres. As can be seen in the figure 5.6, only the extremities of the manipulator, which are more prone to collisions, are modelled, but the geometry is not accurately represented. It can

be seen that P_5 and P_9 were modelled based on the size of the the segment going from the shoulder to the elbow offset and the segment going from the elbow to the wrist offset.

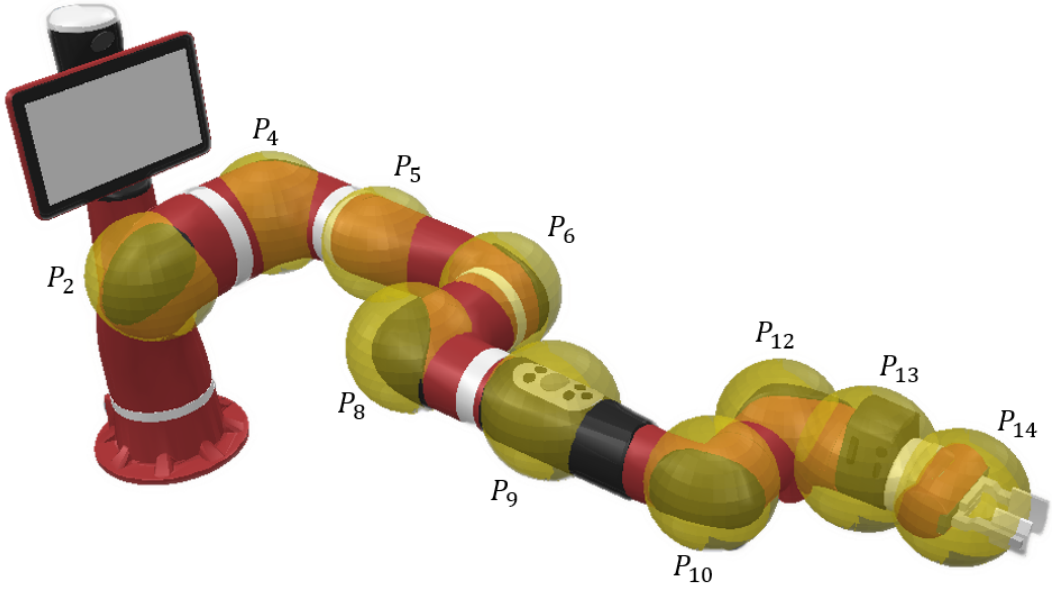


Figure 5.6: Model of the arm of the robotic manipulator Sawyer.[6]

5.4.3 Model of the Objects

In the final posture and bounce posture selection problems, the existing objects in the workspace are represented by ellipsoids, as shown in figure 5.7. These square surfaces are obtained by the equation (5.18) in which $a, b, c \in \Re$ correspond to the dimensions of the objects in the x, y and z axes multiplied by an additional factor of 30%. By using this method, we can obtain a more accurate representation of the objects, which ensures collision-free trajectory planning.

$$\left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 + \left(\frac{z}{c}\right)^2 \leq 1 \tag{5.18}$$

Only the objects that are not used in a particular task are subject to this modeling process. In optimization problems, the object to be transported is represented by a series of spheres (Figure 5.8), as it is attached to the end-effector of the robot during the manipulation phase. A procedure based on the size of the object axis determines the number of spheres required to model the object.

[6]Image adapted from de Sá (2018).

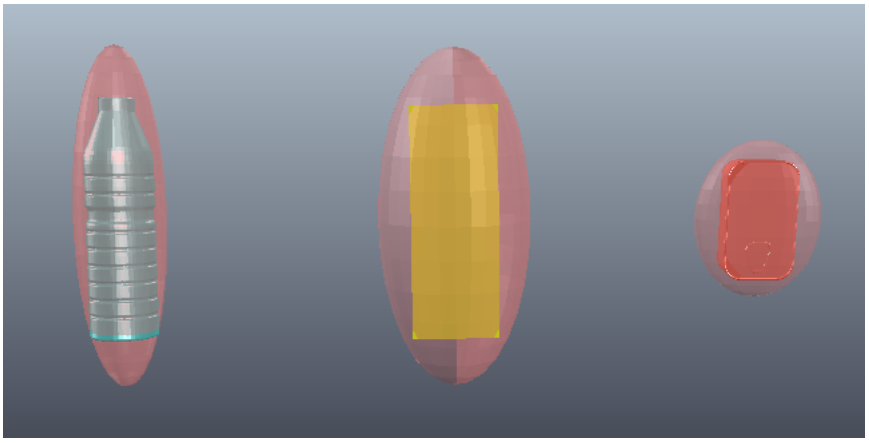


Figure 5.7: The objects in the workspace are represented by a red ellipsoid.

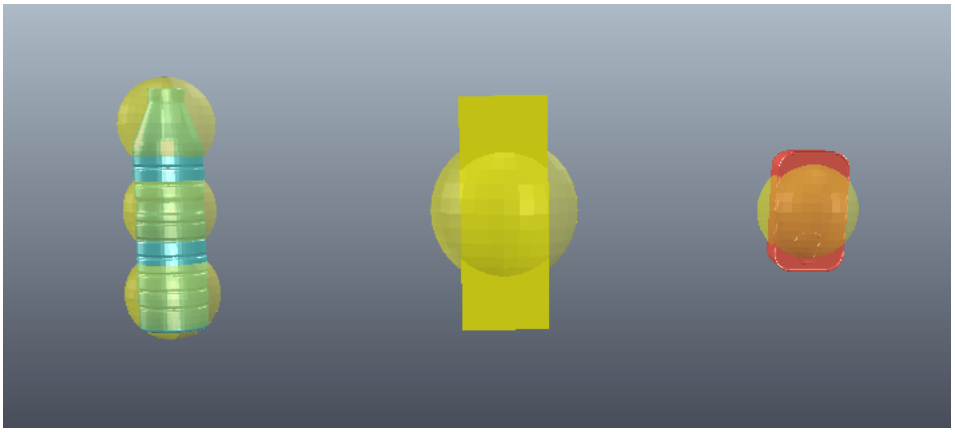


Figure 5.8: During the manipulation phase, the object to be transported is represented by a series of yellow spheres.

5.5 Optimization Software

5.5.1 A Modelling Programming Language (AMPL)

AMPL is a programming language for modeling and solving linear and nonlinear optimization problems. Fourer et al. (1989) implemented this language, which supports a variety of solvers such as IPOPT, KNITRO, Gurobi, CPLEX, and MINOS. The solver to be used to find the best solution is chosen by the user, who can also choose the parameters for its performance.

Due to its simple syntax, the AMPL language is often used to solve optimization problems. Since the mathematical models are written in a compact and understandable algebraic notation, even users with little programming experience can easily solve their problems with this language.

5.5.2 Interior-Point Optimizer (IPOPT)

The IPOPT solver is a free software package that can be used to solve optimization problems with a large number of variables and constraints. This type of solver, implemented in C++ by Wächter and Biegler (2006), allows determining local solutions to optimization problems using a search method known as Interior-Point Filter Line.

As mentioned in the previous section, there are now many types of solvers for solving linear and nonlinear optimization problems. However, based on Silva (2011) work on the ARoS anthropomorphic robot, the proposed motion planning approach exclusively uses the IPOPT optimization tool. Compared to existing solvers such as KNITRO, SNOPT and LOQO, this tool has a lower computational cost according to the authors' experiments.

Chapter 6: Application of the Motion Planner to Sorting Tasks

In this chapter, we examine the performance of the collaborative robot Sawyer in the execution of the motion planning task for the sorting task. First, we address how the motion planning method was validated in a simulated environment. Then, the evaluation criteria for analyzing the human likeness of the motions are presented. In addition, the task of the collaborative robot is explained as well as the sequence of planned motions is described. Finally, a summary of the results is given.

6.1 Overview

The proposed motion planning method is validated in a simulated environment to ensure the integrity of the robotic platform and workspace. The figure 6.1 shows the architecture for the execution of motor actions in the robot, which consists of three main modules:

(i) V-REP (Virtual Robot Experimental Platform) simulator, also known as Coppelia Sim;

(ii) Motion Manager;

(iii) Motion Planner.

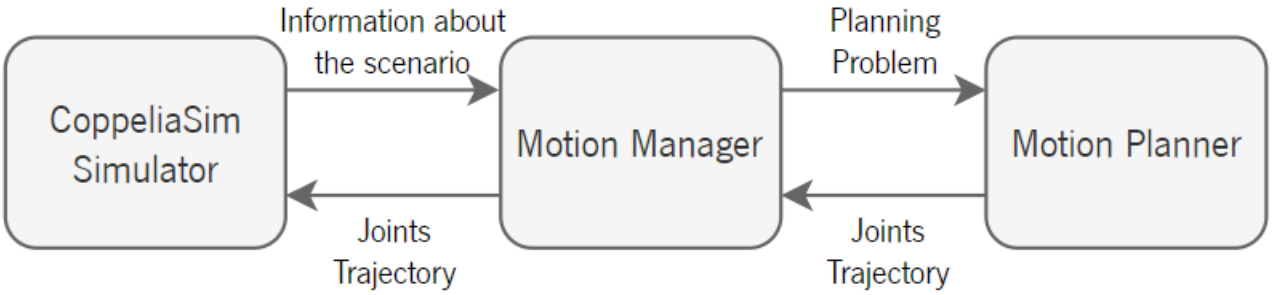


Figure 6.1: Validation of the Motion Planning Method is processed through the modules: V-REP simulator; Motion Manager; Motion Planner(HUMP).[1]

[1]Image adapted from de Sá (2018).

The open-source ROS (Robot Operating System) framework by Quigley et al. (2009) serves as an intermediary for communication between the various components. The Motion Manager module is used to collect data about the scenario, including the robotic platform, obstacles in the workspace, and the target object. This module allows the user to specify certain motion criteria, such as the distances to be maintained from obstacles, the distance to be covered during the approach and retreat phases, and the initial velocity and acceleration of each joint of the robotic platform.

The motion planning problem is then planned and given to the Motion Planner module (HUMP) using the previously specified information. This C++ module generates a collision-free trajectory similar to that of a human.

Then, the joint trajectory is transmitted to the robotic simulator V-REP where the scenario is projected. This communication is handled by the ROS framework, which, as mentioned earlier, allows individual control of each joint of the robotic platform.

6.2 Evaluation Criteria

The primary purpose of this dissertation is to deploy a motion planning approach that allows Sawyer, a collaborative robot, to reach, grasp, sort, and move objects in the same way that humans do. The proposed method will be compared to a set of quantitative measures typically used in psychology and neurology when studying human arm movements (Gulletta et al. (2021)).

Although other factors such as speed, force, and control strategy have been considered in various experimental studies of humans with gate disorders, smoothness of the movement is used to evaluate human-robot interaction scenarios (Chang et al. (2005)). According to Aboelnasr et al. (2017), the smoothness of a movement can be quantified using two kinematic metrics: the change in acceleration per time (Normalized Jerk Score - NJS) and the number of movement units (Number of Movement Units - NMU).

A low NJS value indicates a smoother and more efficient grasping movement. According to Chang et al. (2007), this value can be calculated using the equation 6.1, where the parameters t represents the total time of movement and s represents the distance traveled by the end-effector, x represents the position of the end-effector on the $x-axis$, y represents the position of the end-effector on the $y-axis$, and z represents the position of the end-effector on the $z-axis$.

$$NJS = \sqrt{\frac{1}{2}\left(\frac{t^5}{s^2}\right)\int\left(\left(\frac{d^3x}{dt^3}\right)^2+\left(\frac{d^3y}{dt^3}\right)^2+\left(\frac{d^3z}{dt^3}\right)^2\right)dt} \tag{6.1}$$

A kinematic metric called NMU shows a similar trend to that described above, with a larger NMU indicating less uniform motion. A single unit of motion is commonly used to denote grasping movements (Chang et al. (2005)). This author believes that the NMU can be determined using the following procedures:

(i) Searching the velocity profile for local minima and maxima;

(ii) An increase in velocity between the adjacent minimum and maximum that exceeds a threshold value signifies the occurrence of a unit of motion.

6.3 Sorting Task

The proposed motion planning method is validated in a context similar to automated production lines in the industrial domain. In this scenario, we want the collaborative robot to sort the different products on the shelf.

The different items to be sorted are scattered on a table in front of the collaborative robot Sawyer, as shown in 6.2. Three types of items are present:

(i) Water bottle - blue;

(ii) Cookie box - yellow;

(iii) Tuna Can - red;

In this scenario, there is also a shelf divided into four sections where the items are to be sorted (Figure 6.3). The water bottle is sorted into the top shelf, the tuna can is sorted into the middle shelf, and the cookie boxes are sorted into the bottom shelf.

The task to be performed by the robotic platform is explained in the following sections, where the motion sequence to be executed is analysed and planned using the proposed motion planning method.

6.3.1 Task Description

The first validation task is realized in the previous scenario with the Sawyer robotic platform sorting the water bottle into the shelf.

As can see in the figure 6.4, the execution of this task requires the planning of three motor actions in which the robot must perform the following movements:

(i) reaching and grasping the water bottle (Figure 6.4 (b));

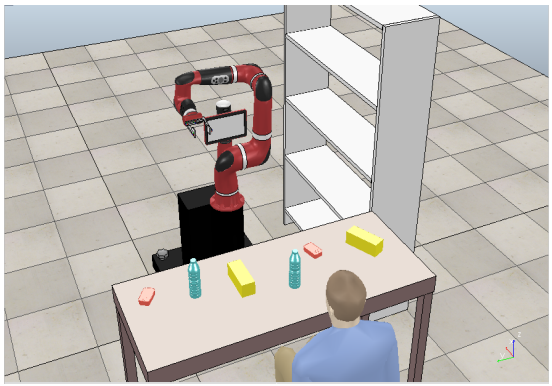


Figure 6.2: Scenario of the sorting task.

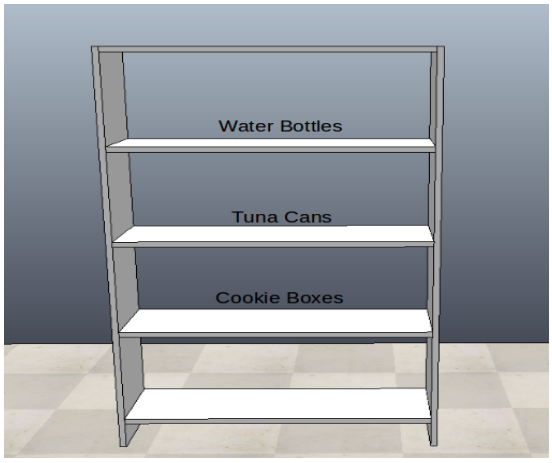


Figure 6.3: Shelf where the items will be sorted.

(ii) transporting and placing the water bottle in the corresponding shelf (Figure 6.4 (c));

(iii) going back to the initial posture (Figure 6.4 (d)).

(a) Initial Posture

(b) Reach and grasp the water bottle

(c) Transport and place the water bottle

(c) Go back to the initial posture

Figure 6.4: Sequence of movements in order to sort the water bottle in the simulation scenario.

The movements described must be fluid and avoid collisions with the other objects in the workspace, with the shelf and with the robotic platform itself.

Another important consideration is the initial position of the collaborating robot in the scenario. The joint values in the table 6.1 were chosen so that the robotic platform occupies only a small portion of the workspace and the end-effector is not on the table.

Table 6.1: Initial posture of the collaborative robot Sawyer in the scenario.

Joint	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	θ_9	θ_{10}	θ_{11}
Value (degrees)	0	-80	-90	90	-90	-90	90	0	70	70	70

6.3.2 Movement Sequence and Obtained Results

In the next sections, we present the motor actions performed by the robotic platform during the sorting task. As mentioned before, this sequence of motions is represented by the motion planning method proposed in this dissertation as well as the evaluation of the obtained results, namely the position and velocity profile of the end-effector and the performance evaluation metrics mentioned in the previous section. The results presented were obtained using an Intel®Core™ i7-7700HQ CPU @ 2.80GHz processor running the 64-bit operating system Ubuntu 16.04 LTS, with an Intel®HD Graphics 630 (Kaby Lake GT2) graphics card and 8Gb of RAM.

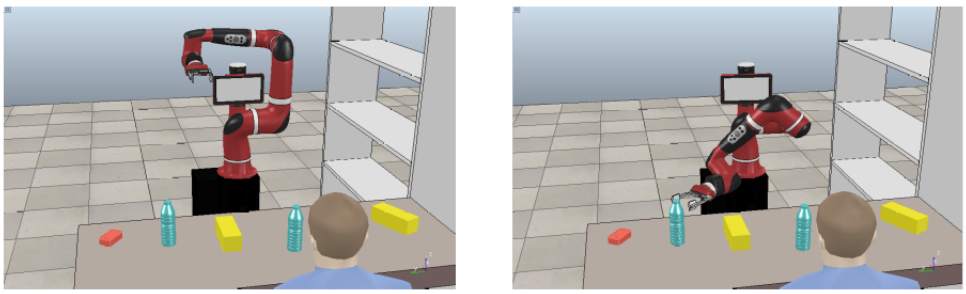
As mentioned earlier, pick-and-place movements are composed of the plan, approach, and retreat stages. The plan stage includes the problems of final posture and bounce posture, so this is the only stage in which collisions must be considered, leaving the approach and retreat stages with only the problem of final posture without the need to consider collisions. The application of this methodology allows the robotic platform to perform smooth and precise movements, which are fundamental for the correct execution of the task. Next, the complete motion results for each of the scenario objects are presented.

6.3.2.1 Water Bottle 1

Pick Movement In the figure 6.5 we can see the stages of the movement performed by the end-effector from its initial position to the water bottle 1 in the table.

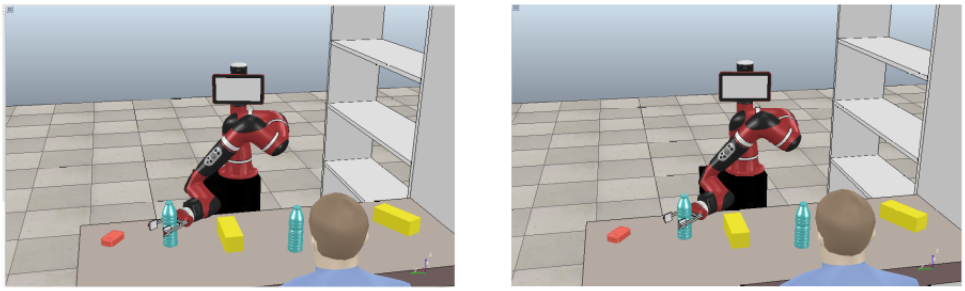
In this scenario, the target position of the plan stage is defined at $100mm$ distance from the water bottle, and the position to be reached by the retreat stage is defined at $30mm$ distance from the water bottle in the $z - axis$ direction from the target position of the approach stage.

As can be seen in figure 6.6, the grasping motion of the water bottle has features similar to those of human upper limbs. The grasping motion is characterized by a bell-shaped profile of the end-effector velocity (Figure 6.6(b)), with a slightly curved trajectory (Figure 6.6(a)). The method of ensuring that the velocity and acceleration values at the beginning of a stage match those checked at the end of the previous stage, allows for smooth and fluid execution of the movements. In addition to the aspects already mentioned, it is important to note that the collaborative robot does not collide with any of the existing



(a) Initial Posture

(b) Final Posture of the Plan stage

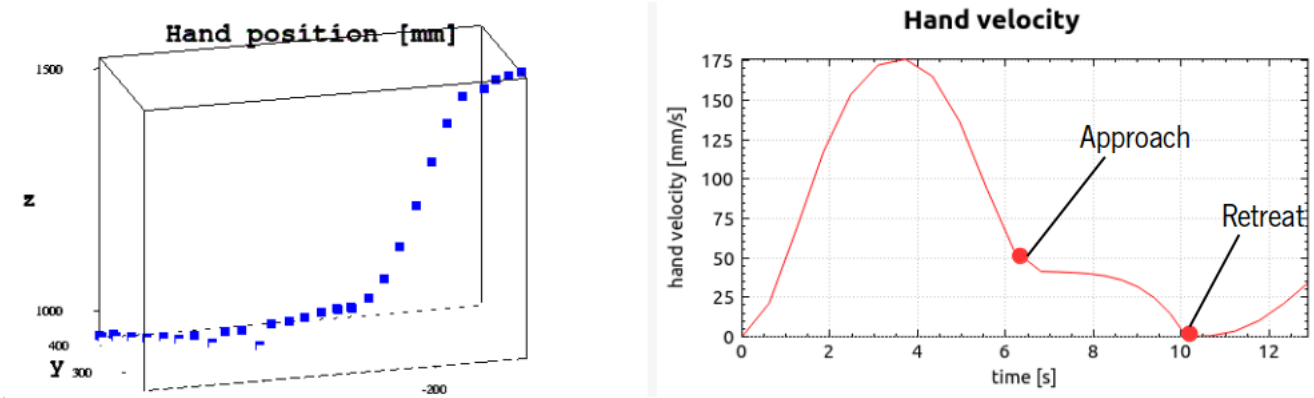


(c) Final Posture of the Approach stage

(d) Final Posture of the Retreat stage

Figure 6.5: Planned movement sequence for the grasping movement of the water bottle 1.

obstacles in the workspace during the task.



(a) Position of the end-effector

(b) Velocity of the end-effector

Figure 6.6: Kinematic characteristics for the pick movement of the water bottle 1.

Relatively to the evaluation metrics (table 6.2), the NMU corresponds to what has been demonstrated for grasping movements performed by human upper limbs. The NJS has a slightly higher value compared to the position of the end-effector, implying that the motion performed by the robotic platform could be smoother and more precise. The resolution time required by the motion manager may vary depending on the processor used, the complexity of the motion, the number of spheres modelling the arm and the end-effector of the manipulator, the arrangement of the workspace, and finally the initial posture of the

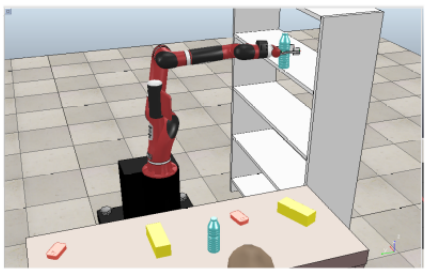
robot.

Table 6.2: Planning results of the pick movement for the water bottle 1.

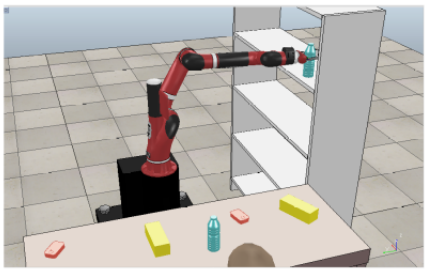
NJS	MNU	Resolution time (ms)
2182.87	1	4902

Place Movement In the figure 6.13, we see the stages of the movement performed by the end-effector from the table while picking the water bottle 1 to the top shelf.

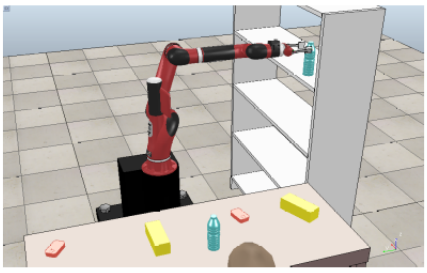
(a) Initial Posture



(b) Final Posture of the Plan stage



(c) Final Posture of the Approach stage

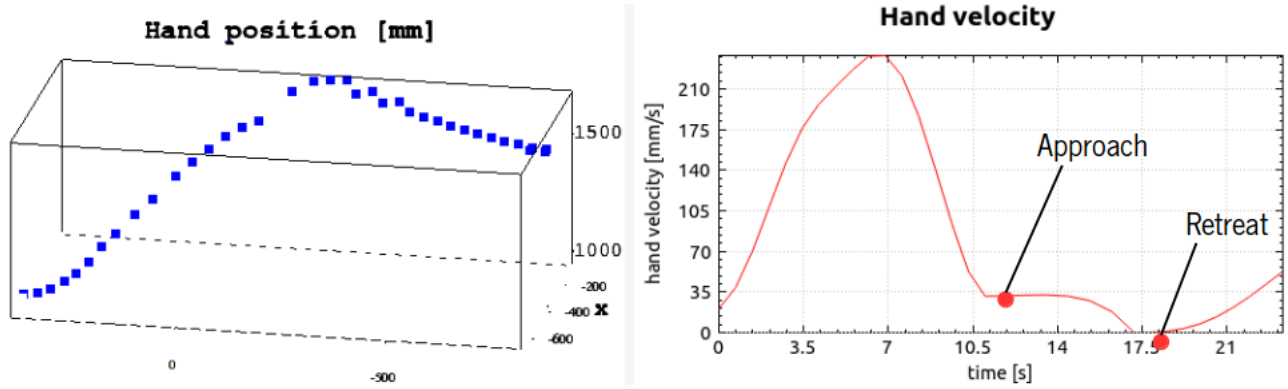


(d) Final Posture of the Retreat stage

Figure 6.7: Planned movement sequence for the place movement of the water bottle 1.

In this scenario, the target position of the plan stage is defined at $65mm$ from the top shelf and the position to be reached by the retreat stage is defined at $45mm$ from the top shelf in the $z - axis$ from the target position of the approach stage.

As can be seen in figure 6.8, the place movement of the water bottle 1 has characteristics similar to those of human upper limbs. According to Flash and Hogan (1985), during the approach stage, it can be seen that the end-effector executes a straight-line trajectory until the target is reached. The place motion is characterized by a bell-shaped profile of the end-effector velocity (Figure 6.8(b)), with a slightly curved trajectory (Figure 6.8(a)). The collaborative robot does not collide with any of the existing obstacles in the workspace during the task.



(a) Position of the end-effector (b) Velocity of the end-effector

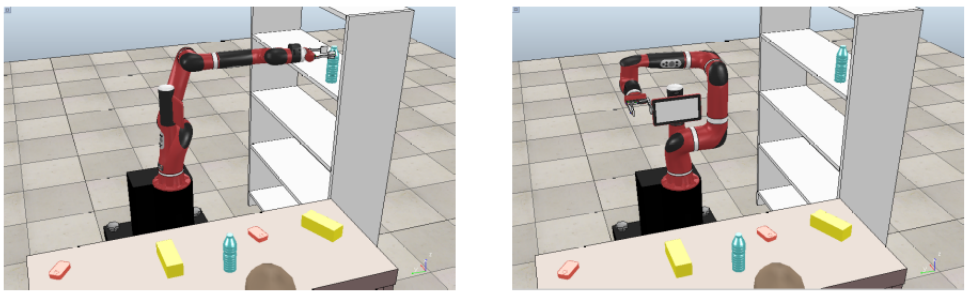
Figure 6.8: Kinematic characteristics for the place movement of the water bottle 1.

With respect to the evaluation metrics (table 6.3), the NMU corresponds to what is checked in the movements performed by human upper limbs. The NJS has a slightly higher value compared to the previous movement, which may be due to the complexity of the movement, the proximity of obstacles in the workspace, the initial posture of the manipulator, and the distance and duration of the approach and retreat stages.

Table 6.3: Planning results of the place movement for the water bottle 1.

NJS	MNU	Resolution time (ms)
3880.97	1	13534

Go-Park Movement The go-park movement is the return of the robotic platform to its initial position (Figure 6.9), which is thus formed only by the move movement and the plan stage.

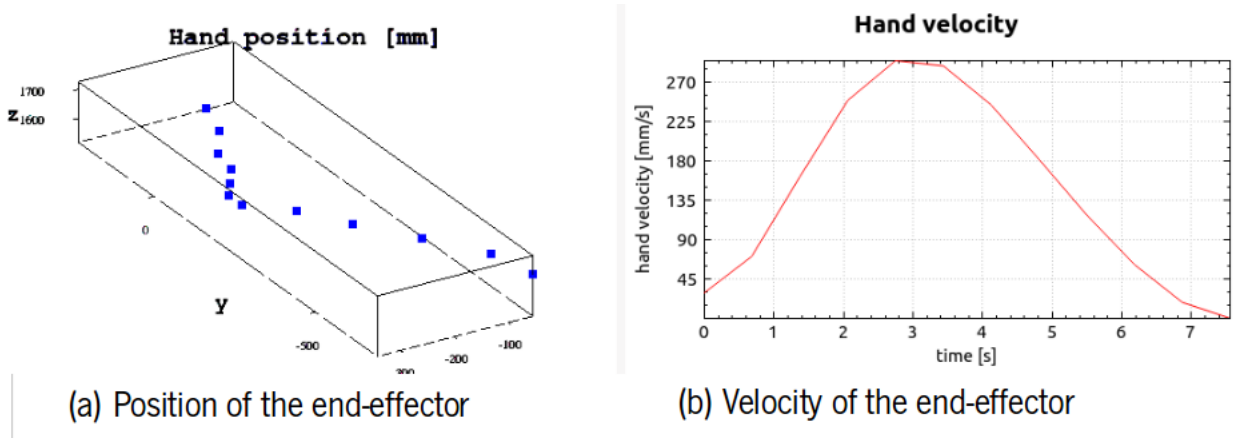


(a) Initial Posture (b) Final Posture

Figure 6.9: Sequence of the go-park movement of the water bottle 1.

As can be seen in figure 6.10, the motion to be executed has a bell-shaped velocity profile (6.10(b)). Again, the velocity and acceleration values match those determined at the end of the place movement, so

the trajectory of the end-effector is smooth and flowing (6.10(a)). The manipulator does not collide with any obstacle in the workspace.



(a) Position of the end-effector (b) Velocity of the end-effector

Figure 6.10: Kinematic characteristics for the go-park movement of the water bottle 1.

Relatively to the evaluation metrics (table 6.4), the NMU corresponds to what has been demonstrated for movements performed by human upper limbs. The NJS has low values compared to the other movements, which is due to the absence of the problems in the final posture selection problems in the approach and retreat stages, so the values for the go-park movement are significantly lower.

Table 6.4: Planning results of the go-park movement for the water bottle 1.

NJS	MNU	Resolution time (ms)
105.486	1	4205

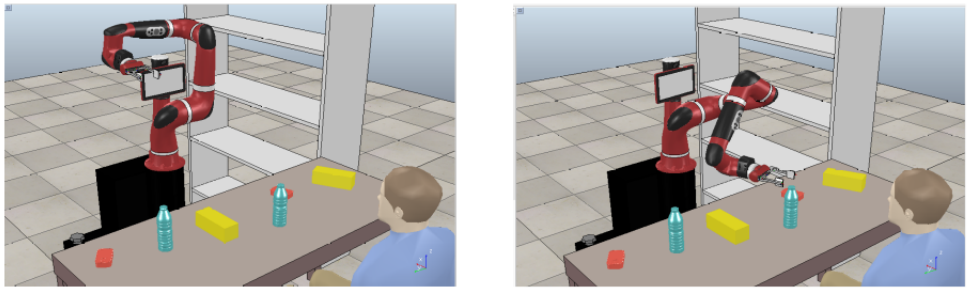
6.3.2.2 Water Bottle 2

Pick Movement In the figure 6.11, we see the stages of the movement that takes the water bottle 2 from the table to the top shelf.

In this scenario, the target position of the plan stage is defined at $100mm$ distance from the water bottle, and the position to be reached by the retreat stage is defined at $30mm$ distance from the water bottle in $z - axis$ direction from the target position of the approach stage.

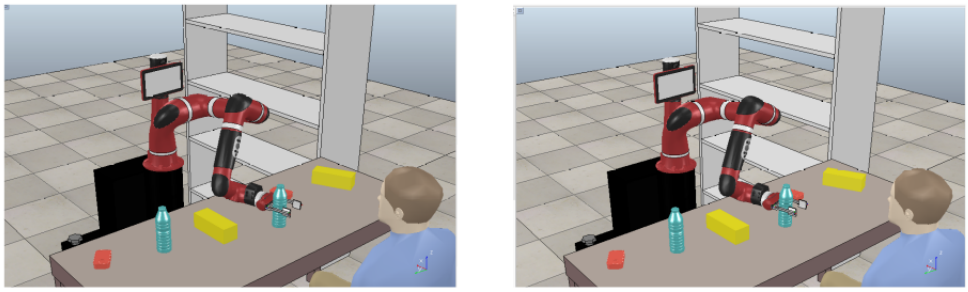
As can be seen in figure 6.12, the grasping motion of the water bottle has characteristics similar to those of the human upper limbs. The grasping motion is characterized by a bell-shaped profile of the end-effector velocity (Figure 6.12(b)), with a slightly curved trajectory (Figure 6.12(a)). The collaborative robot does not collide with any of the existing obstacles in the workspace during the task.

Relatively to the evaluation metrics (table 6.5), the NMU corresponds to what has been demonstrated



(a) Initial Posture

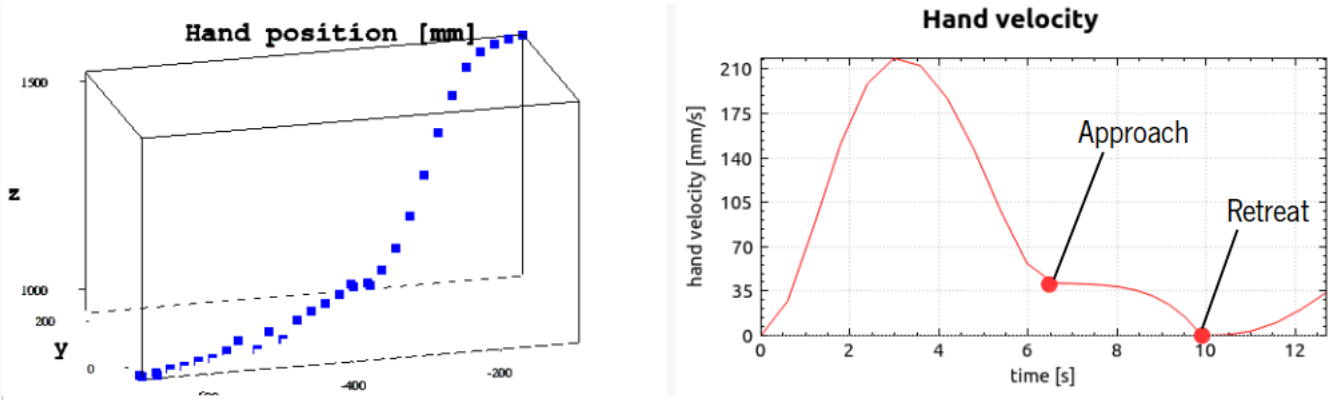
(b) Final Posture of the Plan stage



(c) Final Posture of the Approach stage

(d) Final Posture of the Retreat stage

Figure 6.11: Planned movement sequence for the grasping movement of the water bottle 2.



(a) Position of the end-effector

(b) Velocity of the end-effector

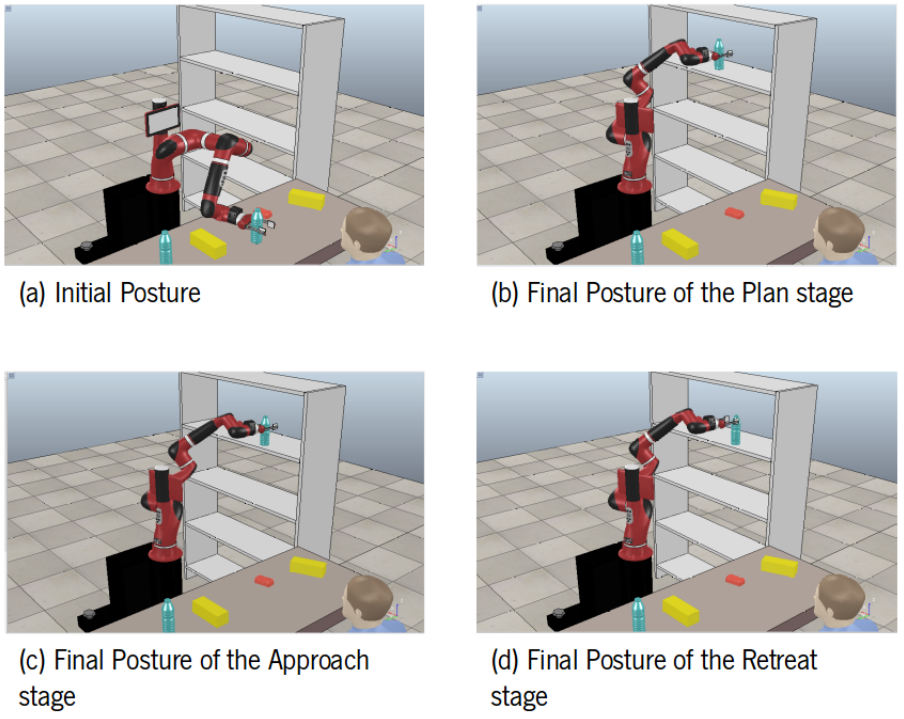
Figure 6.12: Kinematic characteristics for the pick movement of the water bottle 2.

for the movements performed by human upper limbs. The NJS has a slightly high value when compared to the position of the end-effector, where we can see that some points deviate slightly from the trajectory, implying that the motion performed by the robotic platform could be smoother and more precise.

Table 6.5: Planning results of the pick movement for the water bottle 2.

NJS	MNU	Resolution time (ms)
1950.53	1	6837

Place Movement In the figure 6.13 we can see the stages of the movement performed by the end-effector from the table while grasping the water bottle 2 to the top shelf.



(a) Initial Posture

(b) Final Posture of the Plan stage

(c) Final Posture of the Approach stage

(d) Final Posture of the Retreat stage

Figure 6.13: Planned motion sequence for putting down the water bottle 2.

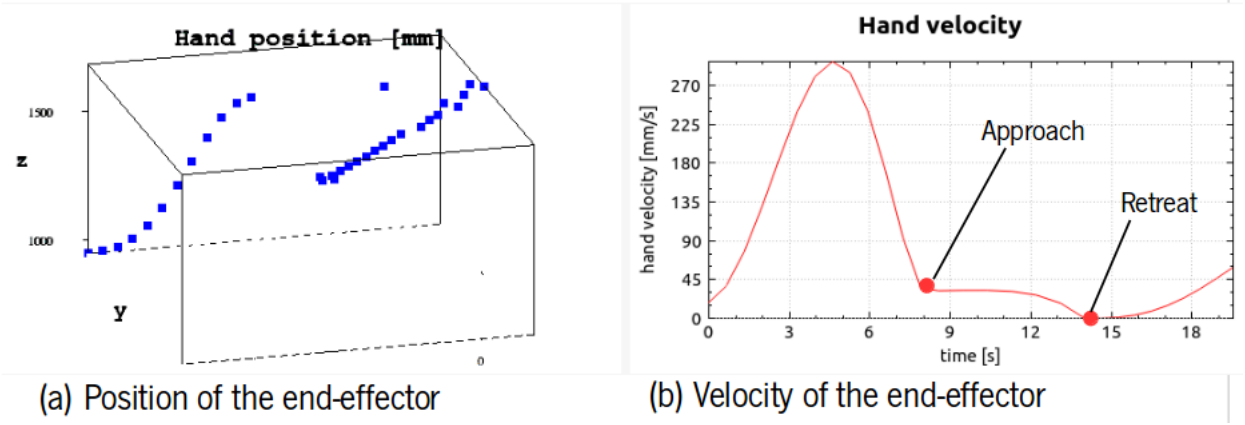
In this scenario, the target position of the plan stage is defined at $65mm$ from the top shelf, and the position to be reached by the retreat stage is defined at $45mm$ from the top shelf in the direction of $z - axis$ from the target position of the approach stage.

As can be seen in figure 6.14, the place movement of the water bottle 1 presents characteristcs similar to those of human upper limbs. The place movement is characterized by a bell-shaped profile of the end-effector velocity (Figure 6.14(b)), with a curved trajectory (Figure 6.14(a)). The collaborative robot does not collide with any of the existing obstacles in the workspace during the task.

Relatively to the evaluation metrics (table 6.6), the NMU corresponds to what is checked for the movements performed by human upper limbs. The NJS has a higher value compared to the previous movement because it is a more complex movement, as can be seen in the figure (Figure 6.14(a)).

Table 6.6: Planning results of the place movement for the water bottle 1.

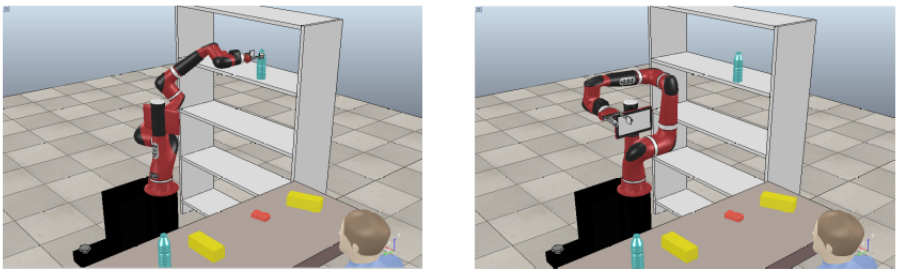
NJS	MNU	Resolution time (ms)
2975.86	1	9861



(a) Position of the end-effector (b) Velocity of the end-effector

Figure 6.14: Kinematic characteristics for the place movement of the water bottle 2.

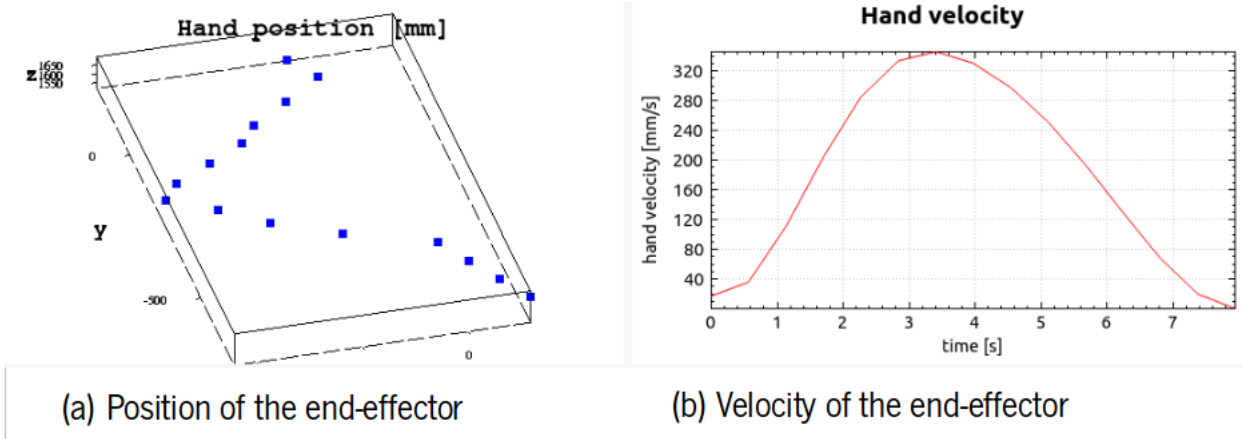
Go-Park Movement The go-park movement is the return of the robotic platform to its initial position (Figure 6.15), which is thus formed only by the move movement and the plan stage.



(a) Initial Posture (b) Final Posture

Figure 6.15: Sequence of the go-park movement of the water bottle 2.

As can be seen in figure 6.16, the motion to be executed has a bell-shaped velocity profile (6.16(b)). Again, the velocity and acceleration values match those determined at the end of the place movement, so the trajectory of the end-effector is smooth and flowing (6.16(a)). The manipulator does not collide with any obstacle in the workspace.



(a) Position of the end-effector (b) Velocity of the end-effector

Figure 6.16: Kinematic characteristics for the go-park movement of the water bottle 2.

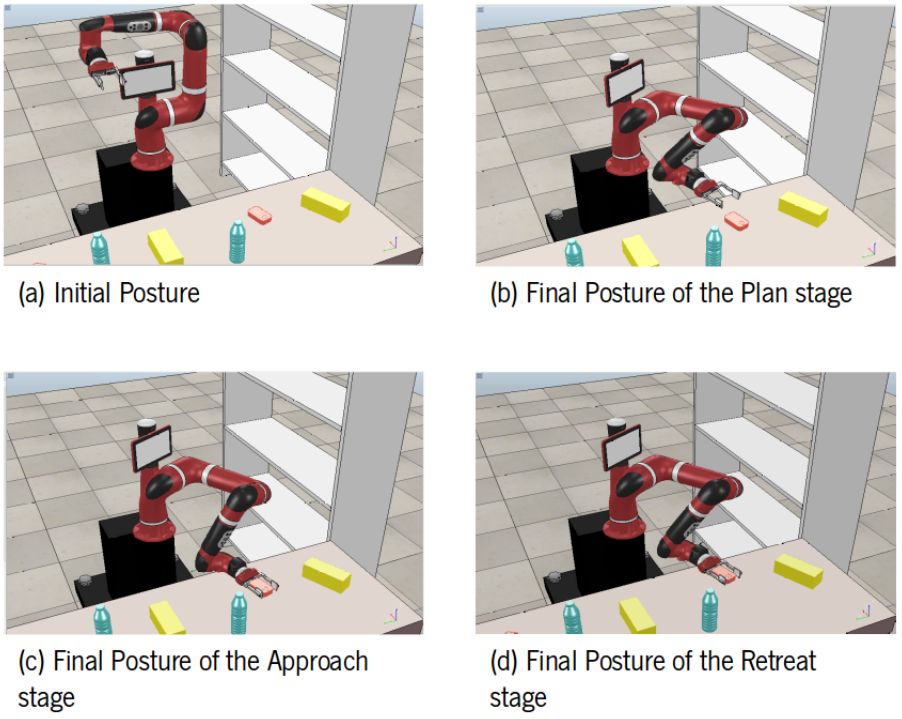
Relatively to the evaluation metrics (table 6.7), the NMU corresponds to what is checked in the movements performed by human upper limbs. The NJS has a very low value.

Table 6.7: Planning results of the go-park movement for the water bottle 2.

NJS	MNU	Resolution time (ms)
58.1073	1	3314

6.3.2.3 Tuna Can 1

Pick Movement In the figure 6.17 we can see the stages of the movement that tuna can 1 makes from the table to the middle shelf.



(a) Initial Posture

(b) Final Posture of the Plan stage

(c) Final Posture of the Approach stage

(d) Final Posture of the Retreat stage

Figure 6.17: Planned movement sequence for the pick motion of tuna can 1.

In this scenario, the target position of the plan stage is set at a distance of $60mm$ from the tuna can and the position to be reached by the retreat stage is set at a distance of $30mm$ from the tuna can in the direction of the $z - axis$ from the target position of the approach stage.

As can be seen in figure 6.18, the grasping motion of the tuna can has characteristics similar to those of the human upper limbs. The grasping motion is characterized by a bell-shaped profile of the end-effector velocity (Figure 6.18(b)), with a highly curved trajectory (Figure 6.18(a)). The collaborative robot does not collide with any of the existing obstacles in the workspace during the task.

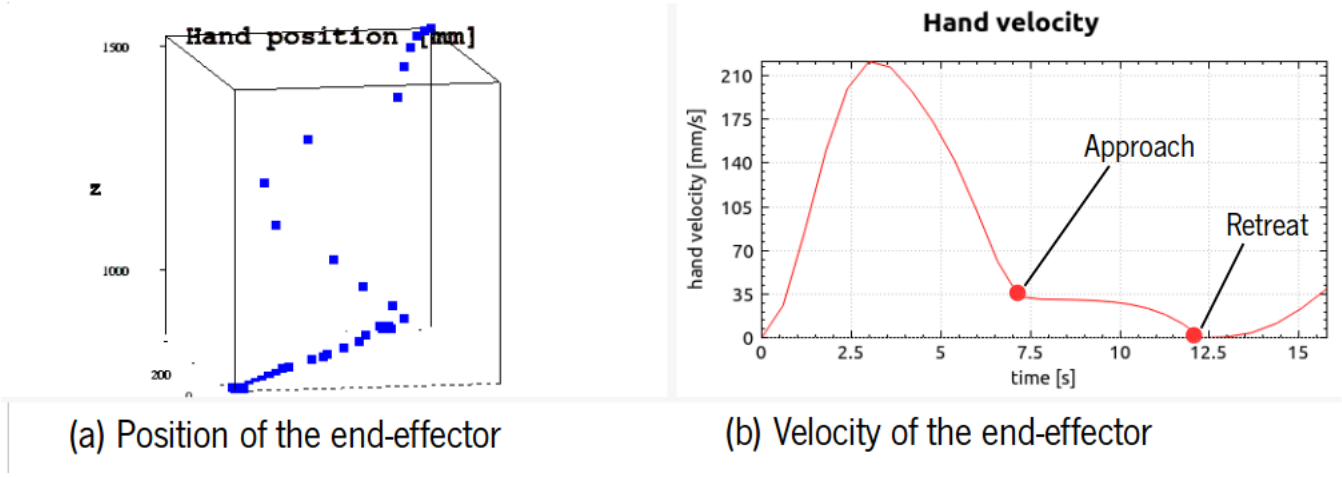


Figure 6.18: Kinematic characteristics for the pick movement of the tuna can 1

With respect to the evaluation metrics (table 6.11), the NMU corresponds to what is checked in the movements performed by human upper limbs. The NJS has a slightly higher value compared to the water bottles, which may be due to the complexity of the trajectory due to its small size compared to the other objects.

Table 6.8: Planning results of the pick movement for the tuna can 1.

NJS	MNU	Resolution time (ms)
2365.8	1	7904

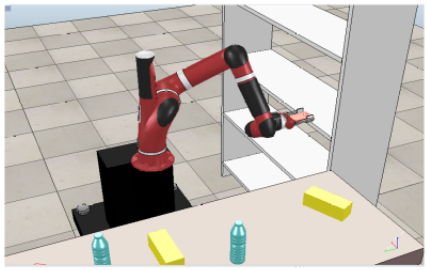
Place Movement In the figure 6.19 we can see the stages of the movement performed by the end-effector from the table while grasping the tuna can 1 on to the middle shelf.

In this scenario, the target position of the plan stage is defined at $60mm$ from the middle shelf, and the position to be reached by the retreat stage is defined at $30mm$ from the middle shelf in the $z - axis$ direction from the target position of the approach stage.

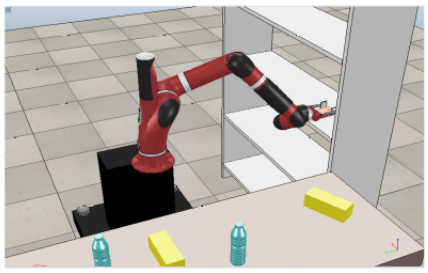
As can be seen in figure 6.20, the place movement of tuna can 1 has similar characteristics to those observed in human upper limbs. The place motion is characterized by a bell-shaped velocity profile of the end-effector (Figure 6.20(b)), with a slightly curved trajectory (Figure 6.20(a)). The collaborative robot does not collide with any of the existing obstacles in the workspace during the task.

Relatively to the evaluation metrics (table 6.9), the NMU corresponds to what is checked in the movements performed by human upper limbs. The NJS is slightly higher compared to the previous movement, but looking at the figure 6.20(a), we can see that this movement could be more fluid and precise.

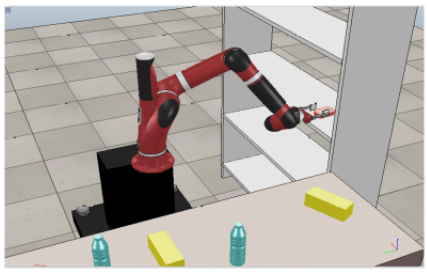
(a) Initial Posture



(b) Final Posture of the Plan stage

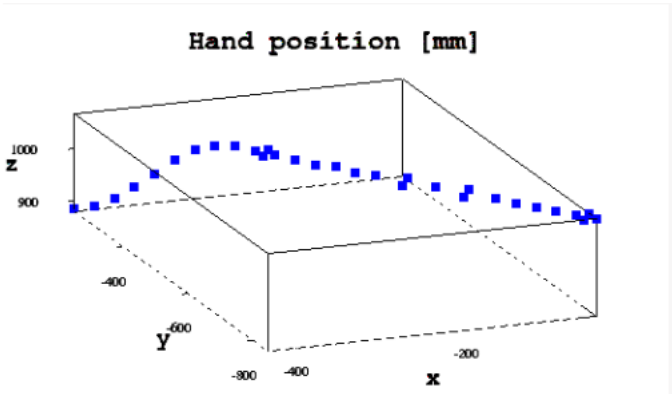


(c) Final Posture of the Approach stage

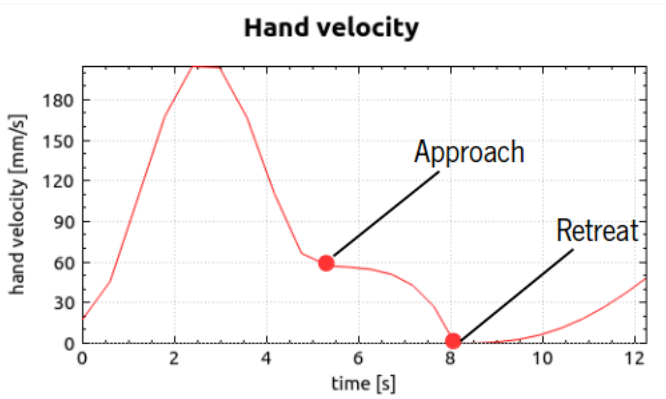


(d) Final Posture of the Retreat stage

Figure 6.19: Planned motion sequence for the place movement of the tuna can 1.



(a) Position of the end-effector



(b) Velocity of the end-effector

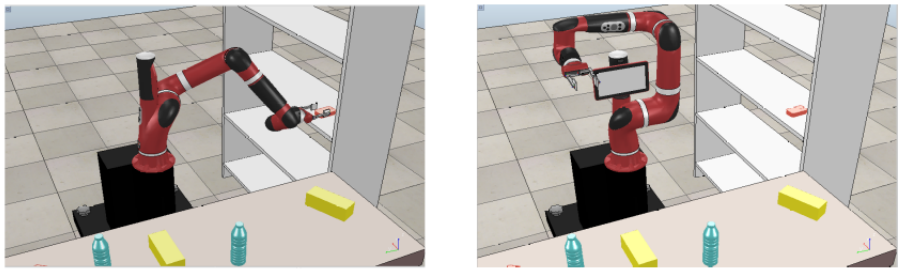
Figure 6.20: Kinematic characteristics for the place movement of the tuna can 1.

Table 6.9: Planning results of the place movement for the tuna can 1.

NJS	MNU	Resolution time (ms)
2988.05	1	7293

Go-Park Movement The go-park movement is the return of the robotic platform to its initial position (Figure 6.21), which is thus formed only by the move movement and the plan stage.

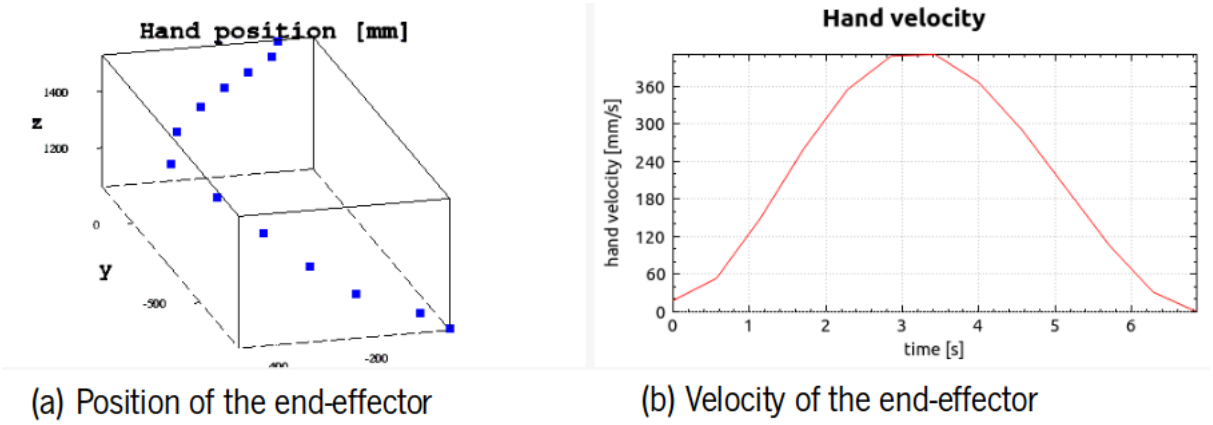
As can be seen in figure 6.22, the motion to be executed has a bell-shaped velocity profile (6.22(b)).



(a) Initial Posture (b) Final Posture

Figure 6.21: Sequence of the go-park movement of the tuna can 1.

Again, the velocity and acceleration values match those determined at the end of the place movement, so the trajectory of the end-effector is smooth and flowing (6.22(a)). The manipulator does not collide with any obstacle in the workspace.



(a) Position of the end-effector (b) Velocity of the end-effector

Figure 6.22: Kinematic characteristics for the go-park movement of the tuna can 1.

Relatively to the evaluation metrics (table 6.10), the NMU corresponds to what is checked in the movements performed by human upper limbs. The NJS has a low value.

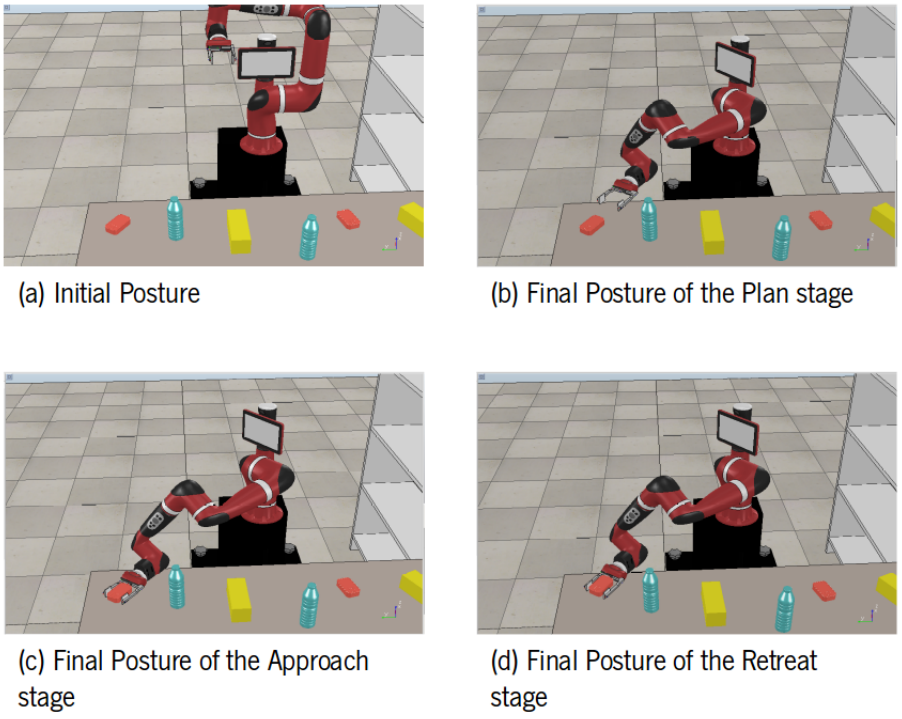
Table 6.10: Planning results of the go-park movement for the tuna can 1.

NJS	MNU	Resolution time (ms)
54.26	1	4166

6.3.2.4 Tuna Can 2

Pick Movement In the figure 6.23 we see the stages of the movement that the tuna can 2 makes from the table to the middle shelf.

In this scenario, the target position of the plan stage is set at a distance of $60mm$ from the tuna can and the position to be reached by the retreat stage is set at a distance of $30mm$ from the tuna can in the



(a) Initial Posture

(b) Final Posture of the Plan stage

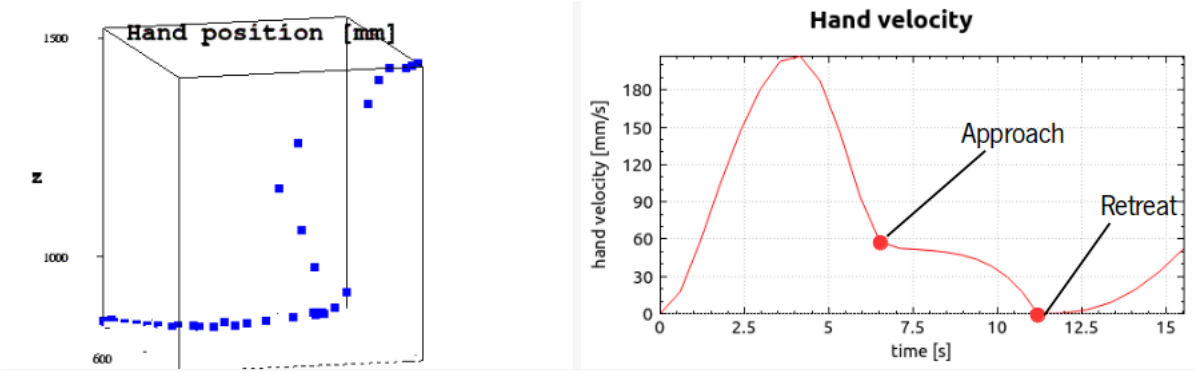
(c) Final Posture of the Approach stage

(d) Final Posture of the Retreat stage

Figure 6.23: Planned movement sequence for the pick motion of the tuna can 2.

direction of the $z - axis$ from the target position of the approach stage.

As can be seen in figure 6.24, the grasping motion of the tuna can has characteristics similar to those of the human upper limbs. The grasping motion is characterized by a bell-shaped profile of the end-effector velocity (Figure 6.24(b)), with a highly curved trajectory (Figure 6.24(a)). The collaborative robot does not collide with any of the existing obstacles in the workspace during the task.



(a) Position of the end-effector

(b) Velocity of the end-effector

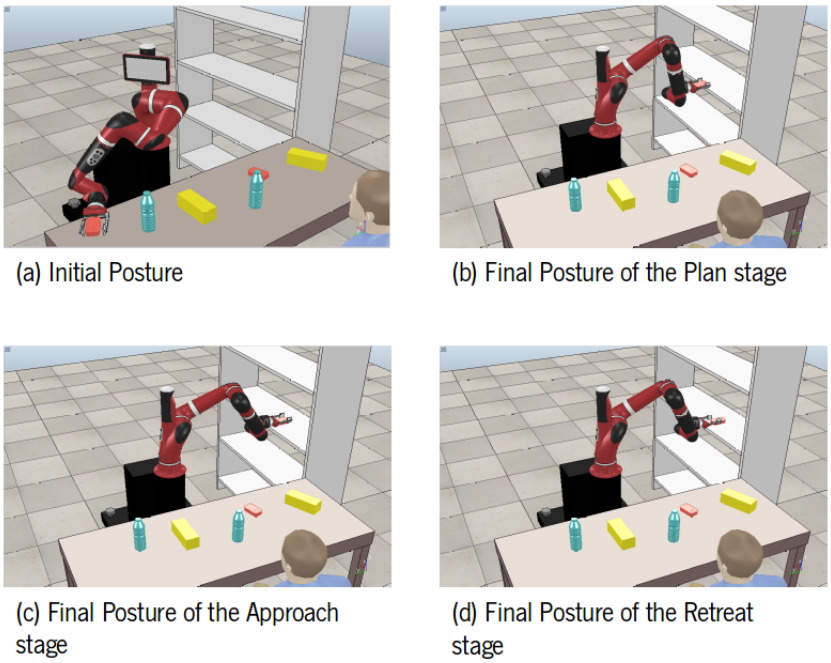
Figure 6.24: Kinematic characteristics for the pick movement of the tuna can 2.

With respect to the evaluation metrics (table 6.11), the NMU corresponds to what is checked in the movements performed by human upper limbs. The NJS has a slightly higher value compared meaning that the trajectory could be more fluid.

Table 6.11: Planning results of the pick movement for the tuna can 2.

NJS	MNU	Resolution time (ms)
2842.86	1	12246

Place Movement In the figure 6.25 we can see the stages of the movement performed by the end-effector from the table while grabbing the tuna can 2 on to the middle shelf.



(a) Initial Posture

(b) Final Posture of the Plan stage

(c) Final Posture of the Approach stage

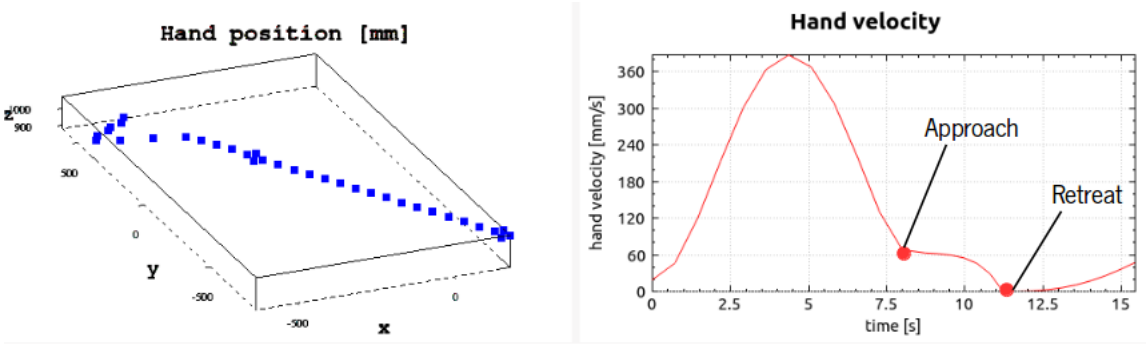
(d) Final Posture of the Retreat stage

Figure 6.25: Planned movement sequence for the place movement of the tuna can 2.

In this scenario, the target position of the plan stage is defined at $60mm$ from the middle shelf, and the position to be reached by the retreat stage is defined at $30mm$ from the middle shelf in the $z-axis$ direction from the target position of the approach stage.

As can be seen in figure 6.26, the place motion of the tuna can 2 has characteristics similar to those of human upper limbs. The place motion is characterized by a bell-shaped end-effector velocity profile (Figure 6.26(b)), with a slightly curved trajectory (Figure 6.26(a)). The collaborative robot does not collide with any of the existing obstacles in the workspace during the task.

Relatively to the evaluation metrics (table 6.12), the NMU corresponds to what is checked in the movements performed by human upper limbs. The value for NJS is slightly increased compared to the previous movement, but the figure 6.26(a) shows that this movement could be more precise and fluid.



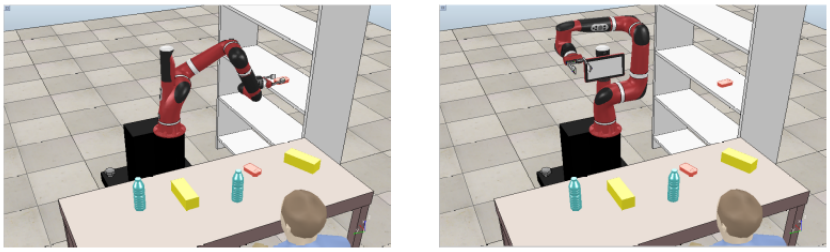
(a) Position of the end-effector (b) Velocity of the end-effector

Figure 6.26: Kinematic characteristics for the place movement of the tuna can 2.

Table 6.12: Planning results of the place movement for the tuna can 2.

NJS	MNU	Resolution time (ms)
2953.63	1	16952

Go-Park Movement The go-park movement is the return of the robotic platform to its initial position (Figure 6.27).

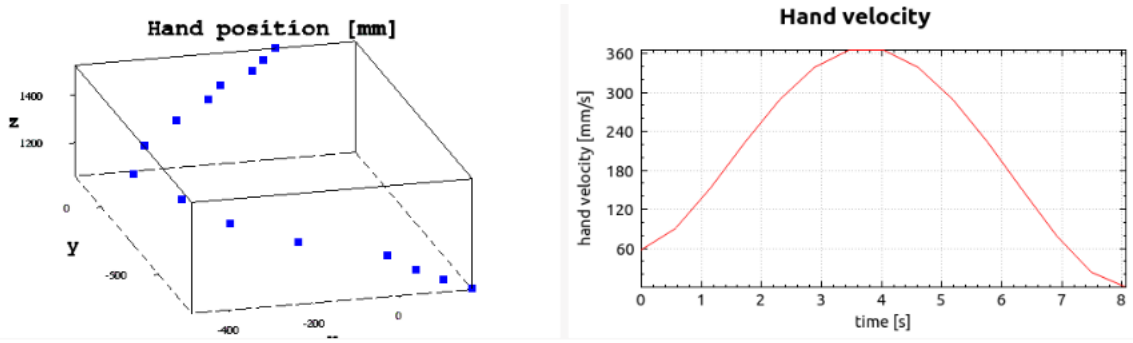


(a) Initial Posture (b) Final Posture

Figure 6.27: Sequence of the go-park movement of the tuna can 2.

As can be seen in figure 6.28, the motion to be executed has a bell-shaped velocity profile (6.28(b)). Again, the velocity and acceleration values match those determined at the end of the place motion, so the trajectory of the end-effector is smooth and flowing (6.28(a)). The manipulator does not collide with any obstacle in the workspace.

Relatively to the evaluation metrics (table 6.13), the NMU corresponds to what is checked in the movements performed by the human upper limbs. The NJS has a low value.



(a) Position of the end-effector (b) Velocity of the end-effector

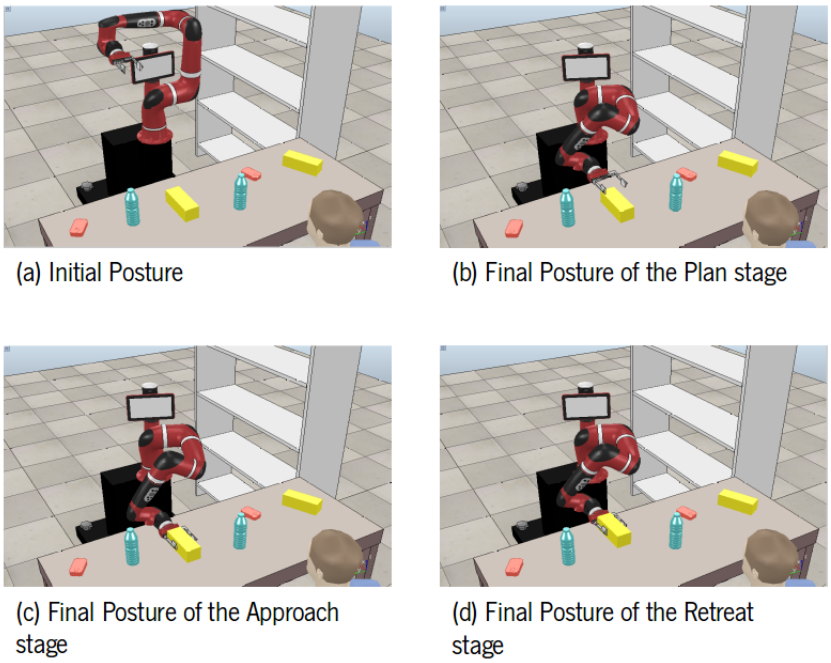
Figure 6.28: Kinematic characteristics for the go-park movement of the tuna can 2.

Table 6.13: Planning results of the go-park movement for the tuna can 2.

NJS	MNU	Resolution time (ms)
54.5437	1	636

6.3.2.5 Cookie Box 1

Pick Movement In the figure 6.29 we see the stages of the movement that the cookie box makes from the table to the lower shelf.



(a) Initial Posture (b) Final Posture of the Plan stage

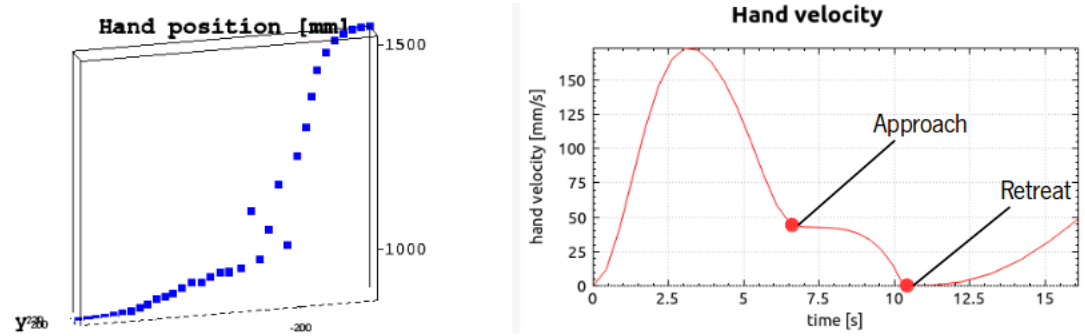
(c) Final Posture of the Approach stage (d) Final Posture of the Retreat stage

Figure 6.29: Planned movement sequence for the pick movement of the cookie box 1.

In this scenario, the target position of the plan stage is defined at a distance of $100mm$ from the

cookie box, and the position to be reached by the retreat stage is defined at a distance of $60mm$ from the cookie box in the direction of the $z - axis$ from the target position of the approach stage.

As can be seen in figure 6.30, the reach and grasp motion of the cookie box has characteristics similar to those of human upper limbs. The grasping motion is characterized by a bell-shaped profile of the end-effector velocity (Figure 6.30(b)), with a curved path approximately in the middle of the motion (Figure 6.30(a)). The collaborating robot does not collide with any of the existing obstacles in the workspace during the task.



(a) Position of the end-effector (b) Velocity of the end-effector

Figure 6.30: Kinematic characteristics for the pick movement of the cookie box 1.

Relatively to the evaluation metrics (table 6.14), the NMU corresponds to what is checked in the movements performed by human upper limbs. The NJS has a slightly higher value compared to the position of the end-effector, showing that the movements performed by the robotic platform could indeed be smoother and more precise.

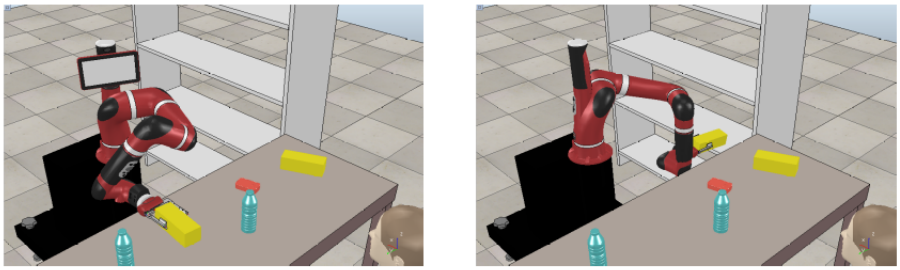
Table 6.14: Planning results of the pick movement for the cookie box 1.

NJS	MNU	Resolution time (ms)
3028.43	1	8967

Place Movement In the figure 6.31 we can see the stages of the movement performed by the end-effector from the table while grasping the cookie box 1 on to the lower shelf.

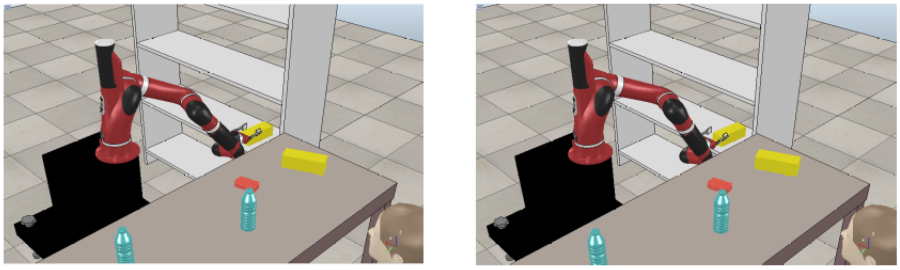
In this scenario, the target position of the approach stage is defined at $60mm$ from the lower shelf and the position to be reached by the retreat stage is defined at $30mm$ from the lower shelf in the direction of $z - axis$ from the target position of the approach stage.

As can be seen in figure 6.32, the place motion of the cookie box 1 has characteristics similar to those of human upper limbs. The place motion is characterized by a bell-shaped end-effector velocity



(a) Initial Posture

(b) Final Posture of the Plan stage

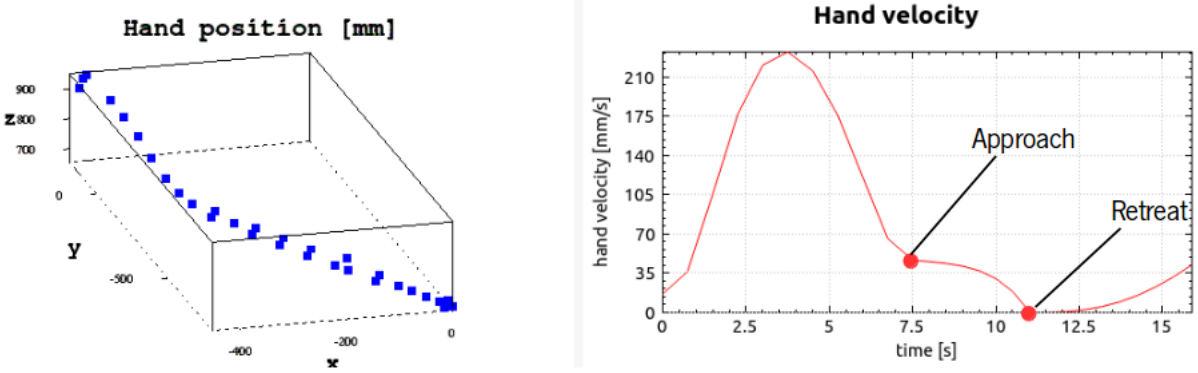


(c) Final Posture of the Approach stage

(d) Final Posture of the Retreat stage

Figure 6.31: Planned sequence of movements for placing the cookie box 1.

profile (Figure 6.32(b)), with a slightly curved trajectory (Figure 6.32(a)). The collaborative robot does not collide with any of the existing obstacles in the workspace during the task.



(a) Position of the end-effector

(b) Velocity of the end-effector

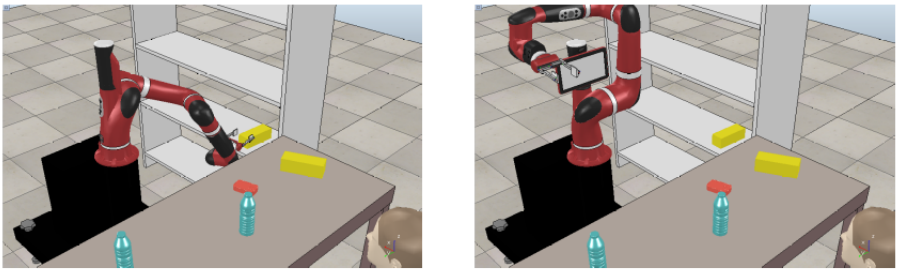
Figure 6.32: Kinematic characteristics for the place movement of the cookie box 1.

Relatively to the evaluation metrics (table 6.15), the NMU corresponds to what is checked in the movements performed by human upper limbs. The value for NJS is slightly higher compared to the previous movement, but looking at 6.32(a), we can see that the movement could be more fluid and precise.

Table 6.15: Planning results of the place movement for the cookie box 1.

NJS	MNU	Resolution time (ms)
3140.93	1	14145

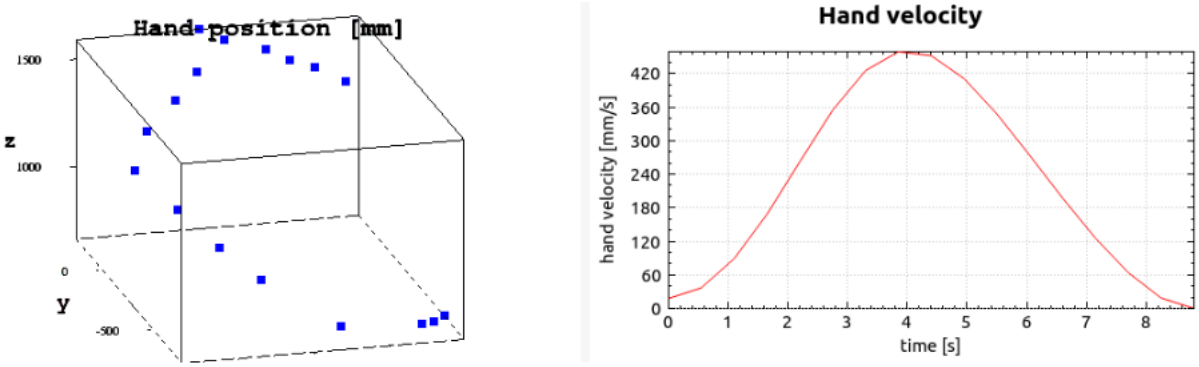
Go-Park Movement The go-park movement is the return of the robotic platform to its initial posture (Figure 6.33).



(a) Initial Posture (b) Final Posture

Figure 6.33: Sequence of the go-park movement of the cookie box 1.

As can be seen in figure 6.34, the motion to be executed has a bell-shaped velocity profile (6.34(b)). Again, the velocity and acceleration values match those determined at the end of the place motion, so the trajectory of the end-effector is smooth and flowing (6.34(a)). The manipulator does not collide with any obstacle in the workspace.



(a) Position of the end-effector (b) Velocity of the end-effector

Figure 6.34: Kinematic characteristics for the go-park movement of the cookie box 1.

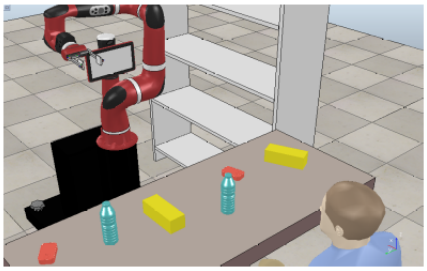
With respect to the evaluation metrics (table 6.16), the NMU corresponds to what is checked in the movements performed by human upper limbs. The NJS has a low value.

Table 6.16: Planning results of the go-park movement for the cookie box 1.

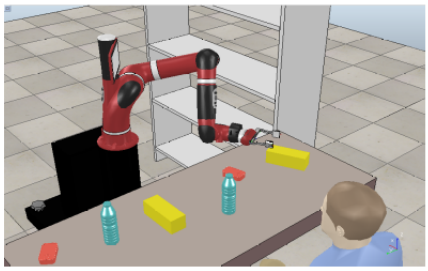
NJS	MNU	Resolution time (ms)
110.994	1	5866

6.3.2.6 Cookie Box 2

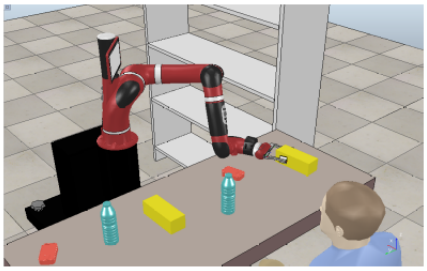
Pick Movement In the figure 6.35 we see the stages of the movement that the cookie box 2 makes from the table to the lower shelf.



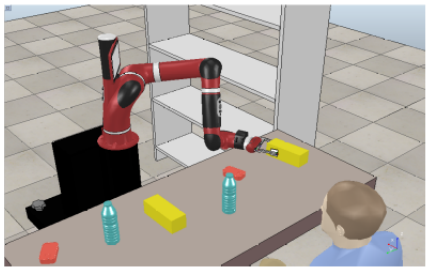
(a) Initial Posture



(b) Final Posture of the Plan stage



(c) Final Posture of the Approach stage



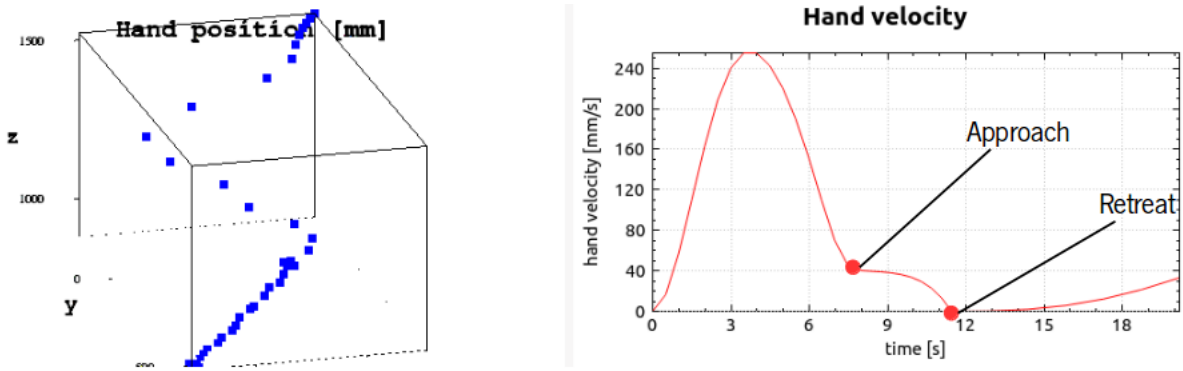
(d) Final Posture of the Retreat stage

Figure 6.35: Planned movement sequence for the pick movement of the cookie box 2.

In this scenario, the target position of the approach stage is defined at $100mm$ distance from the cookie box, and the position to be reached by the retreat stage is defined at $60mm$ distance from the cookie box in the $z - axis$ direction from the target position of the approach stage.

As can be seen in figure 6.36, the grasping motion of the tuna can has characteristics similar to those of the human upper limbs. The grasping motion is characterized by a bell-shaped profile of the end-effector velocity (Figure 6.36(b)), with a highly curved trajectory (Figure 6.36(a)). The collaborative robot does not collide with any of the existing obstacles in the workspace during the task.

With respect to the evaluation metrics (table 6.17), the NMU corresponds to what is checked in the movements performed by human upper limbs. The NJS has a slightly increased value, which means that the movements performed by the robotic platform could be more fluid and precise.



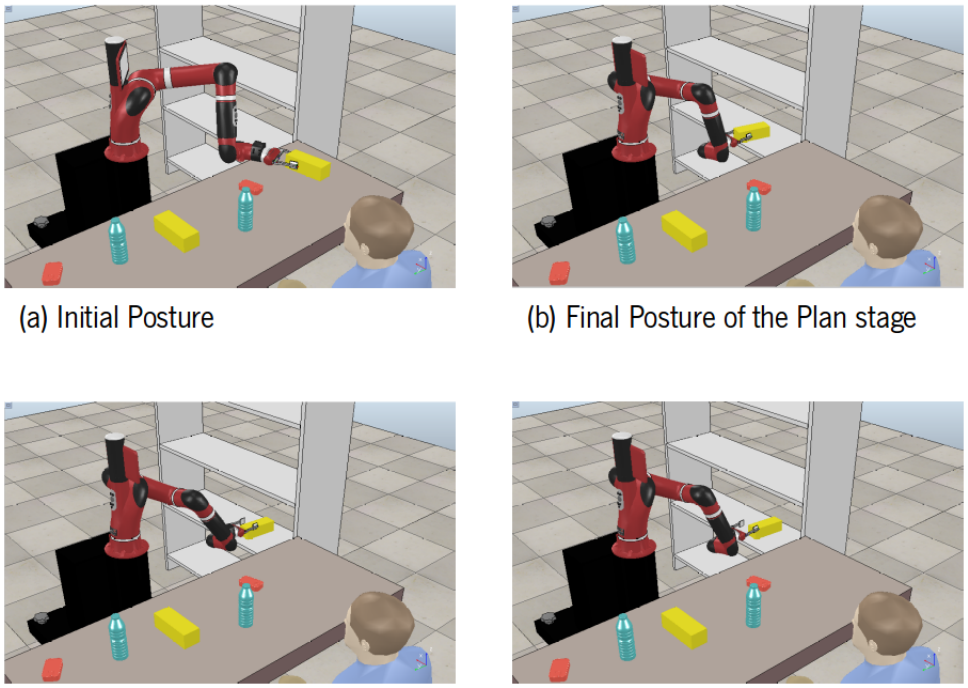
(a) Position of the end-effector (b) Velocity of the end-effector

Figure 6.36: Kinematic characteristics for the pick movement of the cookie box 2.

Table 6.17: Planning results of the pick movement for the cookie box 2.

NJS	MNU	Resolution time (ms)
2881.16	1	18064

Place Movement In the figure 6.37 we can see the stages of the movement performed by the end-effector from the table while grasping the cookie box 2 on to the lower shelf.



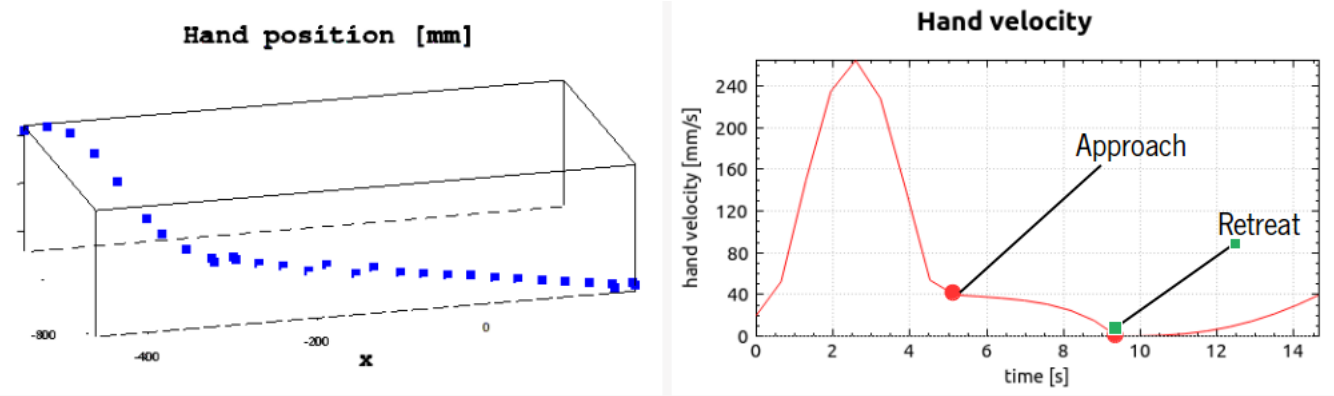
(a) Initial Posture (b) Final Posture of the Plan stage

(c) Final Posture of the Approach stage (d) Final Posture of the Retreat stage

Figure 6.37: Planned sequence of movements for placing the cookie box 2.

In this scenario, the target position of the approach stage is defined at $60mm$ from the lower shelf and the position to be reached by the retreat stage is defined at $30mm$ from the lower shelf in the direction of the $z - axis$ from the target position of the approach stage.

As can be seen in figure 6.38, the place motion of the cookie box 2 has characteristics similar to those of human upper limbs. The place movement is characterized by a bell-shaped end-effector velocity profile (Figure 6.38(b)), with a slightly curved trajectory (Figure 6.38(a)). The collaborative robot does not collide with any of the existing obstacles in the workspace during the task.



(a) Position of the end-effector (b) Velocity of the end-effector

Figure 6.38: Kinematic characteristics for the place movement of the cookie box 2.

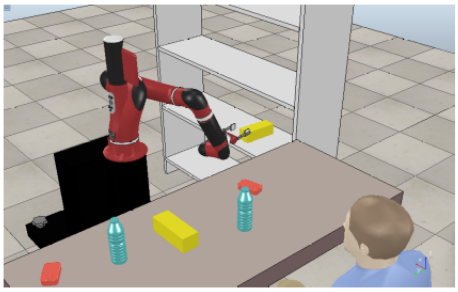
Relatively to the evaluation metrics (table 6.18), the NMU corresponds to what is checked in the movements performed by human upper limbs. The value for NJS is slightly increased compared to the previous movement, and the figure 6.38(a) shows that the movement could be more smooth and precise.

Table 6.18: Planning results of the place movement for the cookie box 1.

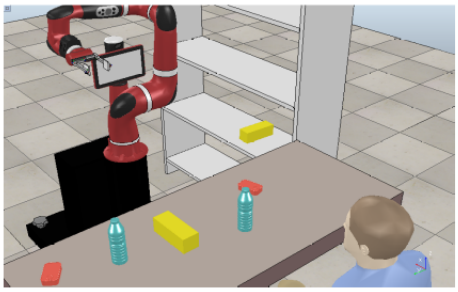
NJS	MNU	Resolution time (ms)
2908.7	1	15988

Go-Park Movement The go-park movement is the return of the robotic platform to its initial posture (Figure 6.39).

As can be seen in figure 6.40, the movement to be executed has a bell-shaped velocity profile (6.40(b)). Again, the velocity and acceleration values match those determined at the end of the place motion, so the trajectory of the end-effector is smooth and flowing (6.40(a)). The manipulator does not collide with any obstacle in the workspace.

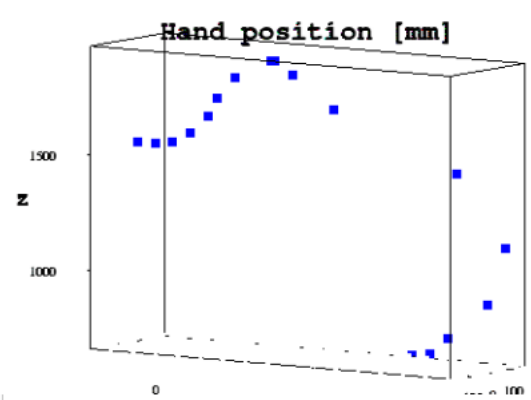


(a) Initial Posture

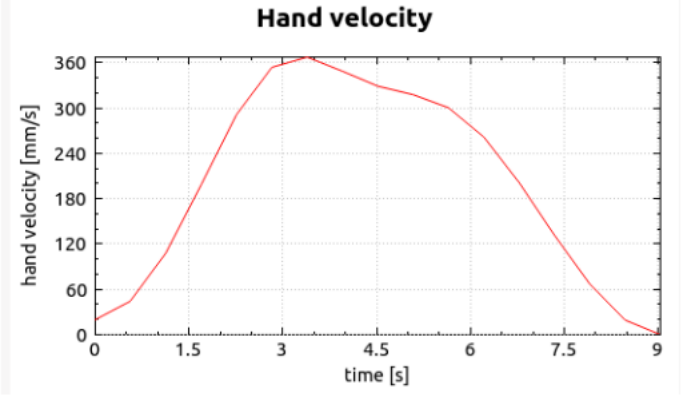


(b) Final Posture

Figure 6.39: Sequence of the go-park movement of the cookie box 2.



(a) Position of the end-effector



(b) Velocity of the end-effector

Figure 6.40: Kinematic characteristics for the go-park movement of the cookie box 2.

Relatively to the evaluation metrics (table 6.19), the NMU corresponds to what is checked in the movements performed by human upper limbs. The NJS has a low value.

Table 6.19: Planning results of the go-park movement for the cookie box 1.

NJS	MNU	Resolution time (ms)
135.21	1	4907

6.4 Discussion

The sorting task, an interactive task that explores the use of the collaborative robot Sawyer in industrial situations, is used to validate the motion planning method.

As mentioned earlier, the proposed motion planning method can provide smooth and fluid movements free of collisions in the above scenario, which are easily understood by human workers. In practice, this method shows effective, stable and adaptive behavior, which is able to plan trajectories for robotic platforms

of different types.

The use of this technology allows the creation of trajectories similar to those observed in experimental tests with human upper limbs. These features relate to:

(i) the bell-shaped profile of the velocity of the end-effector (Flash and Hogan (1985), Rosenbaum et al. (1995));

(ii) the decomposition of the movements of the end-effector into a series of sub-movements (Milner (1992));

(iii) the presence of a single motion unit (NMU) during the reaching and grasping tasks, or of two motion units when there are significant changes in the orientation of the wrist of the robotic platform (Rosenbaum et al. (2001)).

We note that the NJS values in the interactive scenario are quite high, in addition to the previously stated properties. In planning various motor actions, we can conclude that the usefulness of these metrics may change depending on a variety of factors, including:

(i) the complexity of the movement;

(ii) position and orientation of the objects to manipulate;

(iii) the proximity of the obstacles present in the workspace;

(iv) the initial posture and the typology of the robotic platform;

(v) distance and duration of the approach and retreat stages.

Regarding the resolution time spent on the proposed method, the planning process theoretically has a higher computational costs than those required in real interactive jobs. Under normal circumstances, humans do not need 18.06 seconds - the resolution time of the cookie box 2 pick movement - to analyze and respond to the information in their workspace. However, as mentioned earlier, the resolution time can depend on a number of factors, including:

(i) the type of processor used in obtaining the numeric results;

(ii) the disposition of the workspace;

(iii) the complexity of the movement;

(iv) the number of spheres that model the manipulator;

(v) its initial posture;

(vi) the distance between the manipulator and the body, during the task.

In reality, future implementation and validation of the proposed technique in a real environment will require further development of the motion planner by a processor with more complex specifications and validation of its efficiency and robustness in dynamic situations.

Chapter 7: Conclusion

The goal of this dissertation was to provide the collaborative robot Sawyer with the ability to self-program and autonomously generate its movements with an added element of sorting. To do this, the motion planner HUMP (Gulletta (2021)) was used in a new scenario where the goal was to successfully sort the different items into a shelf, taking into account that the motions would have to be as human-like as possible, since there are no complete and consistent human-like motion planning techniques, while avoiding obstacles. Collaborative robotic platforms are becoming more common in industry, so the ability to perform repetitive and human-like tasks autonomously would be an even greater advantage than they currently are.

Taking this into account, a number of features have been evaluated to ensure a more efficient and natural interaction based on human behaviors specifically associated with it:

(i) the generation of comprehensive and simple movements for human operators;

(ii) the avoidance of collisions with obstacles in the workspace, with the human operators and with the robot platform itself;

(iii) the planning of movements in real-time.

The most widely used motion planning methodologies in robotics today exclusively involve solutions to improve efficiency and computation times, making this methodology critical to increasing production and reducing costs associated with various processes. In practice, science does not yet have reliable ways to transfer the knowledge of human motor control to the field of robotics, taking into account the three conditions mentioned above.

A set of features related to human arm behavior and functional models of human upper limbs are included as premises in the proposed motion planning approach. The method is divided into two subproblems following the work of Rosenbaum et al. (2001): The posture-based motion planning model, which divides into:

(i) final posture selection;

(ii) trajectory selection.

The execution of a second motion - back-and-forth movement - which includes the most appropriate selection of the bounce posture for all manipulator typologies with 7DOFs is used to avoid collisions with obstacles present in the workspace, determined by nonlinear optimization problems with constraints. The trajectory selection procedure is effectively independent of the type of robotic platform and is based on the criterion of minimising the acceleration variation (Silva (2011)).

The proposed solution was tested using an interactive simulation called the Sorting Task. Considering the above objectives, the proposed method resulted in smooth and fluid movements, free from collisions, and efficient and flexible for robotic platforms of different types. The proposed trajectories exhibit characteristics similar to those observed in human upper limb behavior, such as a bell-shaped profile of the end-effector velocity, the decomposition of the final movements of the end-effectors into a series of submovements in high-precision tasks, the presence of a single movement unit, and finally, the attainment of slightly high values for NJS.

Regarding the resolution times in motion planning, the processor used to acquire the numerical data does not allow for a verification of the application process in practice. The recorded times for some motor activities are strikingly high, such as 18.06 seconds for picking up the cookie box 2 or 16.95 seconds for putting down the tuna can 2, which requires further experiments with a more advanced processor. Despite this discrepancy, and considering the physical structure of this 7DOF, the proposed technique is preferable to the inverse kinematics dependent methods because its derivation and computation is more complex.

Following the developed project and considering the identified difficulties in the scenario, especially the obstacles, we propose as future work the modification of the modelling of these in the workspace. In this regard, we propose to represent the obstacles in the environment by super quadratic surfaces (as it was used in the modelling of the robots body). In the interaction scenarios, we can observe that the minimum safety distances implemented in the optimization problems are not applicable in this robotic platform.

Unfortunately, given the initial objective of testing this motion planning task on the collaborative Sawyer in a real environment, it was not possible due to the lack of planning and difficulties I had. There were a lot of errors that took a long time to fix, by the time the motion manager was properly working on a simulation environment, the deadline to submit the dissertation was near and there was no time to implement it on a real environment, this was entirely my responsibility since I did not manage my time correctly and I wish I could have done more.

Additionally, as future work we propose:

(i) adaptation of the motion planner to a more recent operating system;

(ii) better modelling of the obstacles in the scenarios;

(iii) comparison of the proposed method with more common algorithms for motion planning in industrial contexts;

(iv) validation of the interactive task in a real environment, even if this process depends on a vision system currently being developed by another colleague.

References

Aboelnasr, E. A., Hegazy, F. A., and Altalway, H. A. (2017). Kinematic characteristics of reaching in children with hemiplegic cerebral palsy: A comparative study. *Brain Injury*, 31:83–89. PMID: 27830945.

Andani, M. E. and Bahrami, F. (2012). Comap: A new computational interpretation of human movement planning level based on coordinated minimum angle jerk policies and six universal movement elements. *Human Movement Science*, 31:1037–1055.

Bicho, E., Erlhagen, W., Louro, L., and e Silva, E. C. (2011). Neuro-cognitive mechanisms of decision making in joint action: A human–robot interaction study. *Human Movement Science*, 30:846–868. EWOMS 2009: The European Workshop on Movement Science.

Bobrow, J. E. (1988). Optimal robot path planning using the minimum-time criterion. *IEEE Journal on Robotics and Automation*, 4:443–450.

Bogue, R. (2017). Robots that interact with humans: a review of safety technologies and standards. *Industrial Robot: An International Journal*, 44:395–400.

Bohlin, R. and Kavraki, L. E. (2000). Path planning using lazy prm.

Campana, M., Lamiraux, F., and Laumond, J.-P. (2016). A gradient-based path optimization method for motion planning. *Advanced Robotics*, 30:1126–1144.

Chang, J.-J., Tung, W.-L., Wu, W.-L., Huang, M.-H., and Su, F.-C. (2007). Effects of robot-aided bilateral force-induced isokinetic arm training combined with conventional rehabilitation on arm motor function in patients with chronic stroke. *Archives of Physical Medicine and Rehabilitation*, 88:1332–1338.

Chang, J.-J., Wu, T.-I., Wu, W.-L., and Su, F.-C. (2005). Kinematical measure for spastic reaching in children with cerebral palsy. *Clinical Biomechanics*, 20:381–388.

Choset, H. M., Lynch, K. M., Hutchinson, S., Kantor, G., Burgard, W., et al. (2005). Trajectory planning. principles of robot motion: theory, algorithms, and implementation. 2007:603.

Craig, J. J. (2005). *Introduction to robotics: mechanics and control*. Pearson Educacion.

de Sá, S. F. M. (2018). Planeamento de movimentos compreensíveis pelo humano para o robô sawyer. *Master Thesis, Universidade do Minho*.

Elbanhawi, M. and Simic, M. (2014). Sampling-based robot motion planning: A review. *IEEE Access*, 2:56–77.

Fiorini, P. and Shiller, Z. (1996). Time optimal trajectory planning in dynamic environments. *Proceedings - IEEE International Conference on Robotics and Automation*, 2:1553–1558.

Flash, T. and Hogan, N. (1985). The coordination of arm movements: an experimentally confirmed mathematical model. *Journal of neuroscience*, 5:1688–1703.

Fong, T., Nourbakhsh, I., and Dautenhahn, K. (2003). A survey of socially interactive robots. *Robotics and Autonomous Systems*, 42:143–166. Socially Interactive Robots.

Fourer, R., Gay, D. M., and Kernighan, B. (1989). Algorithms and model formulations in mathematical programming. *New York, NY, USA: Springer-Verlag New York, Inc*, pages 150–151.

Freivalds, A. (2011). *Biomechanics of the upper limbs: mechanics, modeling and musculoskeletal injuries*. CRC press.

Fukuyama, M. (2018). Society 5.0: Aiming for a new human-centered society. *Japan Spotlight*, 27:47–50.

Garcia, E., Jimenez, M. A., Santos, P. G. D., and Armada, M. (2007). The evolution of robotics research. *IEEE Robotics Automation Magazine*, 14:90–103.

Gasparetto, A. and Zanotto, V. (2010). Optimal trajectory planning for industrial robots. *Advances in Engineering Software*, 41:548–556.

Gladden, M. E. (2019). Who will be the members of society 5.0? towards an anthropology of technologically posthumanized future societies. *Social Sciences*, 8.

Gréa, H., Desmurget, M., and Prablanc, C. (2000). Postural invariance in three-dimensional reaching and grasping movements. *Experimental Brain Research*, 134:155–162.

Gulletta, G. (2021). From human motor control to human-like motion in robotics: planning, learning and controlling manipulation in anthropomorphic robotic systems. *PhD Thesis, Universidade do Minho*.

Gulletta, G., Araújo, S. M., Costa e Silva, E., Costa, M. F., Erlhagen, W., and Bicho, E. (2015). Nonlinear optimization for human-like synchronous movements of a dual arm-hand robotic system. *AIP Conference Proceedings*, 1648(1):140007.

Gulletta, G., Erlhagen, W., and Bicho, E. (2020). Human-like arm motion generation: A review. *Robotics*, 9(4).

Gulletta, G., Silva, E. C. e., Erlhagen, W., Meulenbroek, R., Costa, M. F. P., and Bicho, E. (2021). A human-like upper-limb motion planner: Generating naturalistic movements for humanoid robots. *International Journal of Advanced Robotic Systems*, 18(2):1729881421998585.

Han, S., Wang, L., Wang, Y., and He, H. (2021). An efficient motion planning based on grid map: Predicted trajectory approach with global path guiding. *Ocean Engineering*, 238.

Hauser, K. (2015). Lazy collision checking in asymptotically-optimal motion planning. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 2951–2957. IEEE.

Hwang, Y. K. and Ahuja, N. (1992). Gross motion planning—a survey. *ACM Computing Surveys (CSUR)*, 24:219–291.

Iossifidis, I. and Schöner, G. (2004). Autonomous reaching and obstacle avoidance with the anthropomorphic arm of a robotic assistant using the attractor dynamics approach. *Proceedings - IEEE International Conference on Robotics and Automation*, 2004:4295–4300.

Iossifidis, I. and Schöner, G. (2006). Dynamical systems approach for the autonomous avoidance of obstacles and joint-limits for an redundant robot arm. *IEEE International Conference on Intelligent Robots and Systems*, pages 580–585.

Kalakrishnan, M., Chitta, S., Theodorou, E. A., Pastor, P., and Schaal, S. (2011). Stomp: Stochastic trajectory optimization for motion planning. *2011 IEEE International Conference on Robotics and Automation*, pages 4569–4574.

Kang, S. B. and Ikeuchi, K. (1994). Determination of motion breakpoints in a task sequence from human hand motion. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 551–556. IEEE.

Karaman, S. and Frazzoli, E. (2011). Incremental sampling-based algorithms for optimal motion planning.

Robotics: Science and Systems, 6:267–274.

Kavraki, L. E., Svestka, P., Latombe, J. ., and Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12:566–580.

Kemp, C., Fitzpatrick, P., Hirukawa, H., Yokoi, K., Harada, K., and Matsumoto, Y. (2008). Humanoids.

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. In *Autonomous robot vehicles*, pages 396–404. Springer.

Kim, H., Miller, L. M., Byl, N., Abrams, G. M., and Rosen, J. (2012). Redundancy resolution of the human arm and an upper limb exoskeleton. *IEEE Transactions on Biomedical Engineering*, 59:1770–1779.

LaValle, S. M. (2006). *Planning algorithms*. Cambridge university press.

LaValle, S. M. et al. (1998). Rapidly-exploring random trees: A new tool for path planning.

Lin, C. S., Chang, P. R., and Luh, J. Y. (1983). Formulation and optimization of cubic polynomial joint trajectories for industrial robots. *IEEE Transactions on Automatic Control*, 28:1066–1074.

Liu, M.-Y., Tuzel, O., Veeraraghavan, A., Taguchi, Y., Marks, T. K., and Chellappa, R. (2012). Fast object localization and pose estimation in heavy clutter for robotic bin picking. *The International Journal of Robotics Research*, 31:951 – 973.

Lynch, K. M. and Park, F. C. (2017). *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, 1st edition.

Maciejasz, P., Eschweiler, J., Gerlach-Hahn, K., Jansen-Troy, A., and Leonhardt, S. (2014). A survey on robotic devices for upper limb rehabilitation. *Journal of NeuroEngineering and Rehabilitation*, 11:3.

Mihelj, M., Bajd, T., Ude, A., Lenarčič, J., Stanovnik, A., Munih, M., Rejc, J., and Šlajpah, S. (2019). Collaborative robots. In *Robotics*, pages 173–187. Springer.

Milner, T. E. (1992). A model for the generation of movements requiring endpoint precision. *Neuroscience*, 49:487–496.

Mukadam, M., Dong, J., Yan, X., Dellaert, F., and Boots, B. (2018). Continuous-time gaussian process motion planning via probabilistic inference. *The International Journal of Robotics Research*, 37:1319–1340.

Napier, J. R. (1956). The prehensile movements of the human hand. *The Journal of Bone and Joint Surgery. British volume*, 38-B:902–913. PMID: 13376678.

Nieuwenhuisen, M., Droeschel, D., Holz, D., Stückler, J., Berner, A., Li, J., Klein, R., and Behnke, S. (2013). Mobile bin picking with an anthropomorphic service robot. pages 2327–2334.

Osa, T. (2020). Multimodal trajectory optimization for motion planning. *The International Journal of Robotics Research*, 39:983–1001.

Papazov, C. and Burschka, D. (2014). An efficient ransac for 3d object recognition in noisy and occluded scenes. dynamic augmented reality for sensory substitution in robot-assisted surgical systems view project navigation for autonomous driving view project an efficient ransac for 3d object recognition in noisy and occluded scenes.

Pardo, D., Möller, L., Neunert, M., Winkler, A. W., and Buchli, J. (2016). Evaluating direct transcription and nonlinear optimization methods for robot motion planning. *IEEE Robotics and Automation Letters*, 1:946–953.

Patel, J. G., Choudhary, Y. A., and Bone, G. M. (2017). Fault tolerant robot programming by demonstration of sorting tasks with industrial objects. *2017 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS)*, pages 278–283.

Perry, J. C., Powell, J. M., and Rosen, J. (2009). Isotropy of an upper limb exoskeleton and the kinematics and dynamics of the human arm. *Applied Bionics and Biomechanics*, 6:175–191.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. (2009). Ros: an open-source robot operating system. volume 3, page 5. Kobe, Japan.

Reimann, H., Iossifidis, I., and Schöner, G. (2010). Generating collision free reaching movements for redundant manipulators using dynamical systems. *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, pages 5372–5379.

Rosenbaum, D. A., Loukopoulos, L. D., Meulenbroek, R. G., Vaughan, J., and Engelbrecht, S. E. (1995). Planning reaches by evaluating stored postures. *Psychological review*, 102:28—67.

Rosenbaum, D. A., Meulenbroek, R. J., Vaughan, J., and Jansen, C. (2001). Posture-based motion planning: applications to grasping. *Psychological review*, 108:709.

Santiago, R. M. C., Ocampo, A. L. D., Ubando, A. T., Bandala, A. A., and Dadios, E. P. (2017). Path planning for mobile robots using genetic algorithm and probabilistic roadmap. *HNICEM 2017 - 9th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management*, 2018-Janua:1–5.

Schaal, S. (2007). The new robotics—towards human-centered machines. *HFSP Journal*, 1:115–126. PMID: 19404417.

Schulman, J., Duan, Y., Ho, J., Lee, A., Awwal, I., Bradlow, H., Pan, J., Patil, S., Goldberg, K., and Abbeel, P. (2014). Motion planning with sequential convex optimization and convex collision checking. *The International Journal of Robotics Research*, 33:1251–1270.

Sherwani, F., Asad, M. M., and Ibrahim, B. (2020). Collaborative robots and industrial revolution 4.0 (ir 4.0). pages 1–5.

Shin, K. G. and Mckay, N. D. (1985). Minimum-time control of robotic manipulators with geometric path constraints. *IEEE Transactions on Automatic Control*, 30:531–541.

Short, A., Pan, Z., Larkin, N., and Duin, S. V. (2016). Recent progress on sampling based dynamic motion planning algorithms. *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM*, 2016-Septe:1305–1311.

Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G. (2009). *Robotics: Modelling, planning and control*. Springer-Verlag London, 1 edition.

Silva, E. C. (2011). Reaching, grasping and manipulation in anthropomorphic robot systems. *PhD Thesis, Universidade do Minho*.

Tanzmeister, G., Wollherr, D., and Buss, M. (2016). Grid-based multi-road-course estimation using motion planning. *IEEE Transactions on Vehicular Technology*, 65:1924–1935.

Tauro, R. A., Kaiser, B., and Wörn, H. (2010). Path planning process optimization for a bin picking system.

Vaughan, J., Rosenbaum, D., and Meulenbroek, R. (2001). Planning reaching and grasping movements: The problem of obstacle avoidance. *Motor control*, 5:116–135.

Vaughan, J., Rosenbaum, D., and Meulenbroek, R. (2006). Modeling reaching and manipulating in 2-and

3-d workspaces: The posture-based model. *Proceedings of the Fifth International Conference on Learning and Development*.

Wächter, A. and Biegler, L. T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57.

Wolpert, D. M. (1997). Computational approaches to motor control. *Trends in Cognitive Sciences*, 1:209–216.

Young, R. W. (2003). Evolution of the human hand: the role of throwing and clubbing. *Journal of anatomy*, 202:165–174.

Zhang, Y., Li, L., Ripperger, M., Nicho, J., Veeraraghavan, M., and Fumagalli, A. (2018). Gilbreth: A conveyor-belt based pick-and-sort industrial robotics application. volume 2018-January, pages 17–24. Institute of Electrical and Electronics Engineers Inc.

Zucker, M., Ratliff, N., Dragan, A. D., Pivtoraiko, M., Klingensmith, M., Dellin, C. M., Bagnell, J. A., and Srinivasa, S. S. (2013). Chomp: Covariant hamiltonian optimization for motion planning. *The International Journal of Robotics Research*, 32:1164–1193.