UNIVERSITÉ DE LORRAINE

Ecole doctorale IAEM Lorraine

Improving the productivity of Collaborative Robots in the context of manufacturing companies: identification and control improvement

Amélioration de la productivité des Robots Collaboratifs dans le cadre des entreprises manufacturières : identification et amélioration de la commande

Thesis

This dissertation is submitted in partial fulfillment of
the requirements for the degree

DOCTOR OF PHILOSOPHY
Specialization: Automatic control, Signal and Image Processing, Computer Engineering

Thesis defended in Metz, France,
May 9, 2022

by Carlos Wilfrido PONCE QUIROGA

Members of the Jury

Jacques GANGLOFF, Professor, Université de Strasbourg	President
Philippe FRAISSE, Professor, Université de Montpellier	Reviewer
Nazih MECHBAL, Professor, ENSAM Paris	Reviewer
Nafissa LAKBAKBI EL YAAQOUBI, Lecturer, Université de Lorraine	Examiner
Yannick AOUSTIN, Professor, Université de Nantes	Examiner
Philippe GARREC, Researcher, CEA-LIST	Co-supervisor
Thibaut RAHARIJAONA, Professor, Université de Lorraine	Co-director
Gabriel ABBA, Professor, Université de Lorraine	Director

"The good news about
computers is that they do
what you tell them to do.
The bad news is that they do
what you tell them to do."

Ted Nelson

A mis padres
y hermanas,

Acknowledgments

This work was carried out in the *Laboratoire de conception fabrication commande* (LCFC) and it was funded by the French Atomic Energy and Alternative Energy Commission (CEA, *Commissariat à l'énergie atomique et aux énergies alternatives*) and the Grand-Est Region in conjunction with the *Université de Lorraine*.

First I would like to thank God for everything he has done in my life, for the countless blessings and opportunities that allowed me to get to this moment. Thank you God, we successfully accomplish this major project.

I would like to thank my thesis director Gabriel Abba, professor at the Université de Lorraine, for accepting the supervision of this thesis. Thank you for all your support during these past few years, for all the messages, suggestions, reminders, and the many proofreadings. I will always be thankful for your noumerous advices and for you trust.

Similarly, I would also like to thank Philippe Garrec, research engineer at the CEA, for accepting the co-supervision of this thesis. Thank you for all the reunions, lessons, proofreadings and all the knowledge shared. Thank you for your enthusiasm about everything, your support and all the moments we shared.

Also, I would like to thank Thibaut Raharijaona, professor at the *Université de Lorraine*, for accepting the co-supervision of this thesis. Thank you for all the help, the meetings, suggestions, and proofreadings.

I express my gratitude to Philippe Fraisse, professor at the *Université de Montpelier*, et Nazih Mechbal, professor at the *ENSAM Paris*, for honoring me by reviewing this work. My gratitude to Jacques Gangloff, professor at the *Université de Strasbourg*, who agreed to chair the defense jury. As well, my gratitude to Nafissa Lakbakbi El Yaaqoubi, lecturer at the *Université de Lorraine* and Yannick Aoustin, professor at the *Université de Nantes*, for having agreed to be part of the evaluation of this work.

I would like to thank Jean-François Antoine for his support and orientation at some stages of this work, and also thanks to Adrien Koessler for his advices and initial insights about setting

up the UR10e robot with ROS. Thanks to all my colleagues and friends from the laboratory LCFC, and to the UL, ENSAM and CEA staff.

A special aknowledgment to the *Instituto de Innovación y Transferencia de Tecnología* (MX) and *Consejo Nacional de Ciencia y Tecnología* (MX) for the support provided.

Finally, I would like to thank my parents and sisters. Thank you for all your support, your prayers, comprehension, and love. And thank you Denisse for all your suggestions. This is possible thanks to all of you!

Abstract

The combination of human capabilities and dexterity with the machine efficiency has been a dream for decades. Technology plays a vital role in helping humans work more efficiently. Thus, collaborative robots have attracted much interest in manufacturing companies.

The first part of the thesis is dedicated on determining the influence of bending moment on friction and torque transfer in transmissions used in common collaborative robots: Harmonic Drive gearboxes. Due to their great capabilities, they are widely applied in the field of collaborative robots. However, despite all the great advantages they offer, they present several problematic behaviors that affect their overall performance. Therefore, two experimental setups are designed and developed to study three types of Harmonic Drive transmissions. Friction laws are obtained using force transfer diagrams, which are a parametric representation of the equilibrium of a transmission subjected to input and output torque. The results present new parabolic friction laws for this type of transmissions.

The essence of the second part is the improvement of the collaboration to reduce the danger of the robot to the human working nearby by avoiding obstacles at maximum operating speed. This improvement can be achieved by proper obstacle avoidance and collision-free path planning, focusing on path continuity, final motion times, and average end-effector speed. We propose an offline path planning approach, considering the obstacle model, given an initial linear path from pose A to pose B. The obstacle model allows to examine in detail the performance of the trajectory in terms of time and traveled distance.

The trajectory planning aims to avoid the obstacle using a series of arcs. A variation of Dubins curves defines the path planning. A set of low-computation equations (using radius, sagittas, tangents, etc.) meets the following requirements: (i) C^1 continuity at the meeting points of the different segments (equal direction and magnitude of the tangents), (ii) minimum orthogonal and longitudinal distance of the new path, relative to the original path and the obstacle (main arrow and chord). From the general study in 2D, we develop the corresponding equations in 3D.

The path planning equations are computed and validated in MATLAB. Then, the libraries and modules are transferred to the Python language, language supported by the robotics simulation framework: ROS and MoveIt, with application to the UR10e cobot.

The simulation results validate the feasibility of the obstacle avoidance planner. These are then adapted to the real robot and improved to address certain issues when using the simulation methodology. The control of the end-effector speed is achieved, and trajectories are evaluated in terms of total distance traveled by the end effector, total task time, average linear velocity of the movement, and resulting joint positions and velocities. The evaluation allows a quantitative comparison to rank the best trajectories.

Résumé

La combinaison des capacités et de la dextérité humaines avec l'efficacité des machines est un rêve depuis plusieurs décennies. La technologie joue un rôle essentiel en aidant les humains à travailler plus efficacement. Ainsi, les robots collaboratifs ont suscité beaucoup d'intérêt dans les entreprises de production manufacturière.

La première partie de la thèse est dédié à déterminer l'influence du moment de flexion sur la friction et le transfert du couple dans les transmissions utilisées dans les robots collaboratifs courants : les réducteurs Harmonic Drive. En raison de leurs grandes capacités, elles sont largement appliquées dans le domaine de robots collaboratifs. Cependant, malgré tous les grands avantages qu'ils offrent, ils présentent plusieurs comportements problématiques qui affectent leurs performances globales. Par conséquent, deux montages expérimentaux sont conçus et développés pour étudier trois types de boîtes de vitesses Harmonic Drive. Les lois de frottement sont obtenues en utilisant des diagrammes de transfert de force, qui sont une représentation paramétrique de l'équilibre d'une transmission soumise à un couple d'entrée et de sortie. Les résultats présentent de nouvelles lois de frottement paraboliques pour ce type de transmissions.

L'essence de la deuxième partie porte sur l'amélioration de la collaboration pour réduire le danger du robot par rapport à l'homme travaillant à proximité en évitant les obstacles à vitesse maximale de fonctionnement. Cette amélioration peut être obtenue par une planification correcte de l'évitement des obstacles et des chemins sans collision, en se concentrant sur la continuité du chemin, les temps de mouvement finaux et la vitesse moyenne de l'effecteur final. Nous proposons une approche de planification de trajectoire hors ligne, en considérant le modèle d'obstacle, étant donné une trajectoire linéaire initiale de la pose A à la pose B. Le modèle d'obstacle permet d'examiner en détail la performance de la trajectoire en termes de temps et de distance parcourue.

La planification de la trajectoire vise à éviter l'obstacle en utilisant une série d'arcs. Une variation des courbes de Dubins définit la planification de la trajectoire. Un ensemble d'équations à faible calcul (utilisant le rayon, les flèches, les tangentes, etc.) répond aux exigences suivantes : (i) continuité C^1 aux points de rencontre des différents segments (direction et amplitude des tangentes égales), (ii) distance orthogonale et longitudinale minimale du nouveau chemin, par

rapport au chemin original et à l'obstacle (flèche principale et corde). A partir de l'étude générale en 2D, nous développons les équations correspondantes en 3D.

Les équations de planification de trajectoire sont calculées et validées dans MATLAB. Ensuite, les bibliothèques et les modules sont transférés dans le langage Python, car il s'agit d'un langage supporté par le cadre de simulation robotique : ROS et MoveIt, avec application au cobot UR10e.

Les résultats des simulations valident la faisabilité du planificateur d'évitement d'obstacles. Ceux-ci sont ensuite adaptés au robot réel, puis améliorés pour répondre à certaines problématiques lors de l'utilisation de la méthodologie de simulation. Les trajectoires sont évaluées en termes de distance totale parcourue par l'effecteur, du temps total de la tâche, de la vitesse moyenne linéaire du mouvement et des positions et vitesses articulaires résultantes. L'évaluation permet une comparaison quantitative pour classer les meilleures trajectoires.

Résumé étendu

Actuellement la technologie joue un rôle essentiel en aidant les humains à travailler plus efficacement. Ainsi, les robots collaboratifs ont suscité beaucoup d'intérêt dans les entreprises de production manufacturière. La combinaison des capacités et de la dextérité humaines avec l'efficacité des machines est un rêve depuis plusieurs décennies. Cette collaboration homme-robot est la raison pour laquelle de nombreux chercheurs tentent de surmonter les obstacles que ce rêve représente.

La robotisation industrielle permet de réduire considérablement le temps nécessaire à la fabrication et à la finition des pièces industrielles et, en même temps, elle réduit le nombre de tâches répétitives confiées aux opérateurs. Cependant, lorsque les robots sont amenés à collaborer avec les humains, ils doivent répondre à une série de restrictions pour assurer la sécurité et le bien-être des opérateurs.

Deux points importants des robots collaboratifs peuvent être soulignés, les problèmes de normalisation et la vitesse des mouvements. Les problèmes de normalisation dépendent des pays et de la législation du travail. L'évolution des normes est lente car il est nécessaire de garantir la sécurité des opérateurs qui travaillent à proximité des robots collaboratifs ou directement en interaction avec eux. En ce qui concerne leur vitesse, comme les robots collaboratifs sont montés dans des environnements sans barrières pour faciliter l'accès au poste de travail, ils sont en moyenne quatre fois plus lents que les robots conventionnels, ce qui réduit leur taux de production.

Cette thèse a deux propositions principales pour résoudre certains obstacles à l'introduction de robots collaboratifs sur les lignes de production. La première consiste en déterminer l'influence du moment de flexion sur le frottement et le transfert du couple dans les transmissions utilisées dans les robots collaboratifs courants comme KUKA ou UR. Et la seconde : trouver une solution pour réduire le danger du robot par rapport à l'homme travaillant à proximité sans restreindre sa vitesse maximale de fonctionnement.

La première partie porte plus précisément sur l'étude de la caractérisation du frottement en fonction de la charge des Harmonic Drives. Ces transmissions, également appelées engrenages à ondes de déformation, utilisent les propriétés de l'élastodynamique pour leur

fonctionnement de base. En raison de leurs grandes capacités, elles ont été largement utilisées dans le domaine de la robotique, et en particulier dans les cobots tels que KUKA et UR. Cependant, malgré tous les grands avantages qu'ils offrent, ils présentent plusieurs comportements problématiques qui affectent leurs performances globales. Le problème principal provient du frottement et de la transmission du couple, et pour cette raison une méthodologie a été introduite pour caractériser et déduire un modèle empirique.

Le contexte de notre étude est la linéarisation de la loi de transfert de couple dans les boîtes de vitesse Harmonic Drive pour permettre améliorer leur contrôle de couple. Dans les robots collaboratifs, la précision du contrôle du couple est particulièrement importante à basse vitesse pour améliorer le contrôle des forces de contact.

Par conséquent, deux montages expérimentaux ont été conçus et développés pour étudier trois types de boîtes de vitesses Harmonic Drive. Les lois de frottement sont obtenues en utilisant des diagrammes de transfert d'effort, qui sont une représentation paramétrique de l'équilibre d'une transmission soumise à un couple d'entrée et de sortie. Sur une transmission mécanique donnée, le frottement agit contre l'effort moteur, qu'il soit appliqué à l'entrée ou à la sortie. Cela signifie qu'il existe potentiellement deux lois de couple de frottement distinctes : DIRECT (motorisation) et INDIRECT (régénération), impliquant des couples de frottement correspondants. Les résultats dans les trois types de boites de vitesse Harmonic Drive présentent de nouvelles lois de frottement paraboliques, ce qui constitue une nouveauté si on les compare avec autres types de transmissions. Les mesures sont effectuées dans des conditions quasi-statiques produisant des résultats hautement reproductibles qui peuvent être considérés comme fiables.

En ce qui concerne l'introduction de la robotique industrielle dans les usines, l'objectif était d'améliorer le contrôle des robots afin qu'ils puissent exécuter des tâches plus complexes avec moins d'intervention humaine. Dans cette optique, les chercheurs ont orienté leurs études vers le problème de la planification des trajectoires et son importance pour l'évitement des collisions : les machines devaient se déplacer d'un endroit à un autre sans collision.

Les techniques de fabrication rigides traditionnelles, en revanche, se sont révélées incapables de répondre aux demandes industrielles actuelles, telles que la production de petits lots et la personnalisation. En conséquence, la fabrication industrielle évolue vers des processus

plus flexibles et plus intelligents : l'Industrie 4.0. La coexistence et la collaboration entre les humains et les robots sont des attributs majeurs de la nouvelle génération de robots. Ils associent l'intelligence humaine en matière de réflexion, d'apprentissage et de prise de décision à la précision, la vitesse et la répétabilité des robots pour s'adapter rapidement aux nouvelles exigences de travail et aux situations imprévues.

L'essence de la deuxième partie de la thèse porte sur l'amélioration de la collaboration entre les robots collaboratifs face à des obstacles. Cette amélioration peut être obtenue par une planification correcte de l'évitement des obstacles et des chemins sans collision, en se concentrant sur la continuité du chemin, les temps de mouvement finaux et la vitesse moyenne de l'effecteur final.

Pour aborder cette problématique, nous proposons une approche de planification de trajectoire hors ligne, en considérant le modèle d'obstacle de particule (pour simplifier les calculs), étant donné une trajectoire linéaire initiale de la pose A à la pose B. Le modèle d'obstacle de particule permet d'examiner en détail la performance de la trajectoire en termes de temps et de distance parcourue.

La planification des chemins sans collision proposé vise à éviter l'obstacle en utilisant une série d'arcs. L'importance d'une bonne planification de chemin repose sur la sélection de la meilleure façon de se rendre d'un point A à un point B en fonction de certains indicateurs de performance, tels que la distance plus courte à parcourir, le coût de fonctionnement le plus bas, le temps de la tâche ou la vitesse moyenne.

La planification de chemin dans ce travail est basée sur une variation des courbes de Dubins. Un ensemble d'équations à faible calcul (utilisant des rayons, des flèches, des tangentes, etc.) répond aux exigences suivantes pour la génération de chemin: (i) continuité C^1 aux points de rencontre des différents segments (direction et amplitude des tangentes égales), (ii) distance orthogonale et longitudinale minimale du nouveau chemin, par rapport au chemin original et à l'obstacle (flèche principale et corde). A partir de l'étude générale en 2D, nous développons les équations correspondantes en 3D.

La planification de chemin pour l'évitement des obstacles n'est qu'un aspect de la commande du manipulateur. L'étape suivante consiste à vérifier la faisabilité cinématique de

cette trajectoire dans le manipulateur, en termes de capacités de l'espace de travail. Une fois que l'exécution d'une trajectoire appropriée est confirmée, une trajectoire en termes de variables articulaires doit être calculée.

Les équations de planification de chemin sont calculées et validées dans MATLAB. Ensuite, les bibliothèques et les modules sont transférés en Python, car il s'agit d'un langage supporté par le cadre de simulation robotique : Robot Operating System (ROS) et MoveIt!, avec application au cobot UR10e. Grâce à cette synergie, nous avons conçu trois trajectoires acceptables d'évitement d'obstacles en simulation. Le paramètre flèche est le principal paramètre réglable de l'algorithme de planification de la trajectoire des obstacles et il peut être facilement adapté aux besoins de l'utilisateur. Le paramètre flèche représente la distance orthogonale la plus courte à laquelle l'effecteur final évite un obstacle particulaire donné (par rapport à la trajectoire linéaire originale).

Le cadre de simulation est préparé en conséquence pour être facilement transférée à un scénario de cas réel. Dans ce sens, l'utilisation d'outils déjà disponibles dans le domaine de la robotique s'avère être un grand atout. MoveIt! avec ROS et l'UR10e, permettent de mettre en place tous les outils nécessaires pour travailler dans des environnements de simulation et la mise en œuvre réelle.

Les simulations valident ainsi la viabilité du planificateur, et conclusions importantes sont extraites, en termes de performances attendues dans le scénario du cas réel. Ils montrent que le problème d'évitement des obstacles diminue en fait les performances temporelles, mais nous constatons qu'en ajustant un paramètre (le flèche) du planificateur de chemin, nous pouvons améliorer ces performances.

Bien que la limitation du contrôle de la vitesse soit le principal inconvénient du calcul des premiers trajectoires, celles-ci possèdent néanmoins deux attributs importantes : i) les différentes configurations articulaires pour chacun des points de cheminement cartésiens sont calculées, nécessaires pour les envoyer et commander le robot, et ii) ces points de cheminement de la trajectoire articulaire ont été évalués par MoveIt!, assurant la viabilité de l'exécution d'une trajectoire donnée, compte tenu de l'espace de travail prédéfini.

En fonction du niveau d'engagement de la collaboration, certaines exigences de sécurité doivent être respectées. Selon la spécification technique ISO/TS 15066:2016, la fonctionnalité collaborative est un état dans lequel un opérateur et un système robotique travaillent dans un espace collaboratif. Ceci précise qu'une limitation à 0,25m/s du point central de l'outil (TCP ou *end-effector*) serait adéquate pour permettre l'interaction homme-robot. Cette vitesse est destinée à laisser aux opérateurs suffisamment de temps pour se retirer ou pour arrêter les robots manuellement.

Les trajectoires générées et les différents modules ROS sont ensuite adaptés au robot réel pour comparer leurs performances entre simulations et cas réels. Les premiers résultats confirment les trajectoires obtenues par les simulations, avec de meilleures performances globales. Cependant, ces trajectoires ne répondent pas aux exigences des limites de vitesse de sécurité.

Pour améliorer et commander en vitesse les trajectoires du robot, une méthodologie en utilisant polynômes d'interpolations est donc introduit et proposé. L'objectif est de pouvoir modifier l'échelle de temps à partir des points de passage des trajectoires calculées précédemment en introduisant un paramètre de vitesse linéaire maximale, afin de lisser le mouvement de l'effecteur final et de contrôler sa vitesse linéaire maximale, assurant ainsi le respect des normes de sécurité.

Les trajectoires sont évaluées en termes de distance totale parcourue par l'effecteur, du temps total de la tâche, de la vitesse moyenne linéaire du mouvement et des positions et vitesses articulaires résultantes. L'évaluation permet une comparaison quantitative pour classer les meilleures trajectoires.

Des conclusions générales sur le travail de recherche sont réalisées dans le cadre de cette thèse de doctorat. Cette thèse a permis de franchir les premières étapes vers l'amélioration du modèle de frottement des Harmonic Drives et d'obtenir des résultats initiaux satisfaisants vers l'amélioration du contrôle de la vitesse linéaire de l'effecteur final pour l'évitement d'obstacles. Enfin, des sujets de recherches futures sont discutés, afin d'améliorer encore la collaboration entre les robots et les humains.

Contents

List of Figures

List of Tables

Chapter 1

1. Introduction

1.1. Context of the work

Technology plays a vital role in helping humans to work more efficiently. Therefore, collaborative robots have received a lot of interest by manufacturing production companies. Their main advantages are: i) low cost compared to other machines or technological solutions for the automation of production processes, ii) robotic capabilities are developing rapidly and covering more and more fields of application, iii) collaboration between humans and robots reduces absenteeism: it helps to increase the pace of the human being, iv) robots have smaller, more versatile moving parts that help them perform tasks more precisely, v) robots are significantly stronger and faster than humans, vi) size of robot can be adapted to the size and the work environment, and vii) robots can work relentlessly, efficiently and under severe environmental conditions that are not suitable for humans.

However, collaborative robots also have disadvantages: i) robots can certainly manage their prescribed tasks, but they cannot usually handle unexpected situations, ii) return on investment is not obvious if robots are not used enough or if they are assigned too few tasks, iii) training employees on how to work with collaborative robots is an added cost that must be taken into account at the production lines design, iv) robots need large number of sensors to manage the different situations in their environment which complicates their use and programming and v) humans are much more effective than robots when it comes to making decisions and managing difficult situations which requires efficient and fast communication.

1.1.1. Motivations and objectives

Industrial robotization considerably reduces the time required for manufacturing and finishing industrial parts, and at the same time, it reduces the number of repetitive tasks assigned to operators. However, when the robots are set to collaborate with humans they must meet a series of restrictions to assure the safety and wellbeing of operators.

Two important points from collaborative robots can be highlighted, the problems of standardization and the speed of movements. The problems of standardization depend on

countries and labor legislation. The evolution of the standards is slow because it is necessary to guarantee the safety of the operators who work close to collaborative robots or directly in interaction with them. Regarding their speed, as collaborative robots are mounted in free-fence environments to facilitate access to the workstation, they are on average four times slower than conventional robots, reducing their production rate.

This thesis proposes to remedy some obstacles for the introduction of collaborative robots on the production lines, towards the goal to improve their overall performances. The first major motivation is to determine the influence of the bending moment on the friction and the transfer of the torque in the transmissions used in common collaborative robots. And the second one, to find a solution to reduce the danger of the robot in relation to the human working nearby without restricting its maximum operating speed.

1.1.2. Contributions

The main research punctual contributions in this manuscript are the following:

1) The friction laws characterization of Harmonic Drives gearboxes
    - Analysis of three different Harmonic Drive models
    - Use of empirical method to obtain the friction laws based on the Force Transfer Diagram
    - Two workbench/setups developed and adapted for testing gearboxes
    - Experimental validation and reproducibility of the friction models

2) Obstacle avoidance in collaborative robots
    - Offline approach towards obstacle avoidance
    - Collision-free path planning focusing on terms of path continuity
    - Trajectory generation and simulation framework using ROS (Robot Operating System), MoveIt! and the UR10e collaborative robot
    - The development of ROS modules (nodes) for obstacle avoidance trajectories generation
    - Improvement of precomputed trajectories to control the end-effector speed

- Experimental testing and validation of new speed-controlled trajectories, evaluating their overall performances

1.2. From tools to collaborative robots

Technology has always pushed forward the development of societies through history. The origin of what today is considered modern technology comes from a long path of ideas and inventions. Humans are characterized for being creative and curious, and since the early Stone Age humans have discovered many things by the process of trial and error. They have tested all sort of things to fulfill their must essential tasks, pushing them to use their creativity and curiosity for the creation of their first tools. These inventions were getting complex as knowledge started to get richer, by taking advantage of previous technologies for the gestation of the next one. In general manner, the purpose of every new technology aims to facilitate the workload of the user, to reduce the duration of the work, to obtain more reliable results, or simply to produce more with less.

If one wants to understand how collaborative robots work and their purposes, sometimes we have to take one step back and look into some key parts in the history, and some of the theory behind the technologies developed.

1.2.1. Technical origins of telemanipulation

Since ancient times, tools allowed to extend hand gestures beyond its reach in order to achieve a certain dexterity (Vertut & Coiffet, 1984). They are meant to ease a given task, and in some cases to prevent user from potential dangers. Let us take for example the pliers used by blacksmiths. The main advantage of this tool is that they can assure the safety of the user when handling incandescent objects, the following advantage is that the user is able to manipulate incandescent objects in the 3D space at certain distance. As shown in Figure 1.1, the blacksmith pliers provide limited handling dexterity within a distance of a few centimeters, but they allow three degrees of freedom (DOF) of position and three degrees of freedom of orientation.

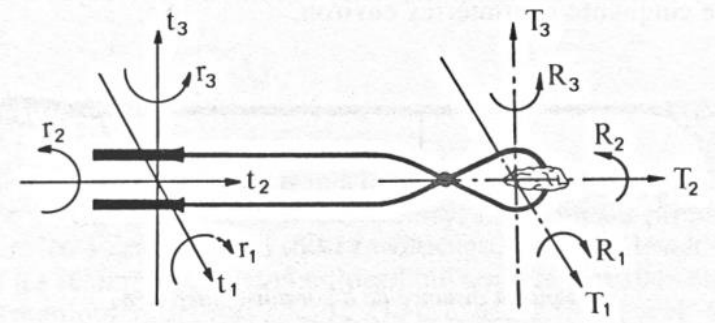


Figure 1.1 Blacksmith pliers (Vertut and Coiffet 1984)

A little further into the future, the first mechanical telemanipulation system was conceived by Raymond Goertz in 1948 and it can be considered as the ancestor of the family of master-slave manipulators. The purpose of this system was to respond to the needs in nuclear environments, as it allowed operators (in the master side) to handle radioactive material (in the slave side) without exposing to radioactive danger. As shown in Figure 1.2, this system permitted operators to transmit motion naturally, providing haptic feedback from the remote zone. However, these mechanical devices were limited by the length of connected links. This led to the development of a new version that was electrically coupled (Goertz & Bevilacqua, 1952), and with it, the foundations of modern telerobotics field was originated.

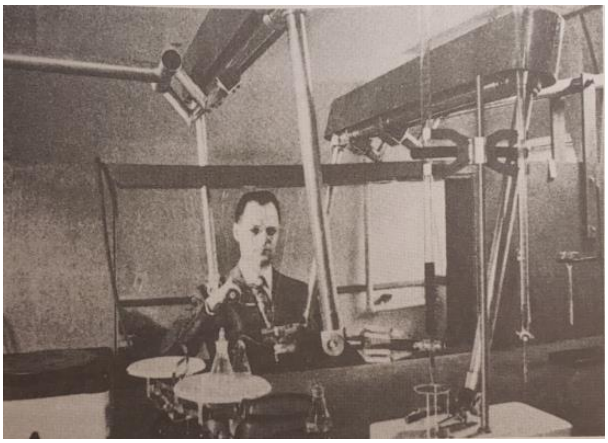


Figure 1.2 First mechanical telemanipulation system

1.2.2. Robots: service robots, industrial robots and collaborative robots

The term robot appeared for the first time in the play *Rossum's Universal Robots* or *R.U.R.*, written by the Czech author Karel Čapek in 1920 (Markel, 2011), which originally was suggested by his brother Josef Čapek. According to Markel, the word *robot* is drawn from an old

Church Slavonic word, “robota”, which means “servitude” or “forced labor”. Since then, the concept started to gain recognition, but it was until the years 40’s and 50’s that gained even more, thanks again to cultural literacy and to science fiction author Isaac Asimov, with his futuristic stories.

In the real world, thanks the invention of transistors and early development of computers, robots came to exist as we nowadays know them. They are not exactly as imagined in fiction but is pertinent to say that many of the concepts and ideas have influenced their development. It is as such, that many researchers have been working hard in the robotic field, trying to overcome the scientific barriers that limit the use of robots to accomplish innovative tasks at the service of men. There are many applications in the field of robotics and we can categorize them given their given purpose.

The International Federation of Robotics (IFR) divides robots into two main categories (IFR, 2019):

— So-called "service" robots, whose purpose is to perform tasks that are useful for humans (with the exception of industrial tasks). Service robots are divided into two sub-categories: professional service robots that are controlled by users trained as part of their job, and service robots as domestic staff used by people without technical knowledge in robotics, in their everyday life.

— Industrial robots, which aim to automate tasks and improve production. They are mainly used in production lines (for example, as assembly robots). Industrial robots capable of learning a gesture or a task by observing a human began to appear (like the robot Baxter).

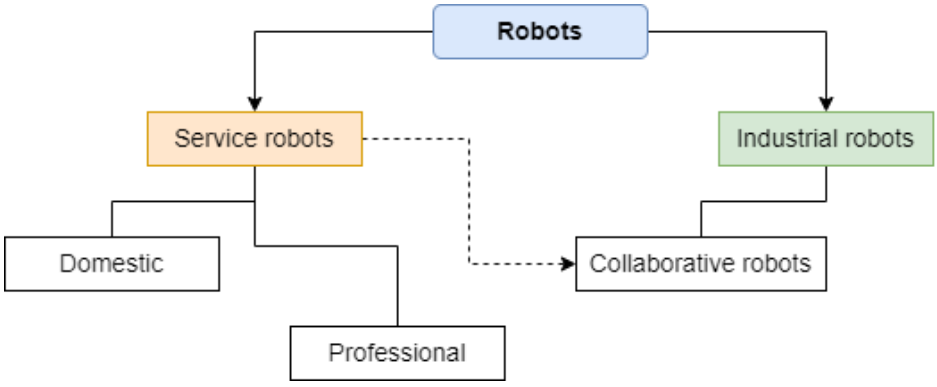


Figure 1.3 Robots’ classification

An industrial robot can be defined as well as a general purpose programmable manipulator, that can be programmed to perform a useful task for the industry. This definition and functionality originated, from the integration of previously mentioned telemanipulators and the Numerical Controlled machines that were developed around the same time. By this nature, in 1961 the first industrial robot, called *Unimate,* made its first appearance in the industry at a General Motors plant (Glagowski et al., 1992).

Fast forward some years into the future, the so called collaborative robots were derived, which according to the IFR they are a direct subcategory from the industrial robots, but in a way, they also respond to the purposes of service robots. We can say they belong in between the two classifications, with major inclination towards the industry (see Figure 1.3).

1.3. Collaborative robots in the industry

In the industry, most of the robots have 6 DOF or more. They are catalogized as serial manipulators and they have a number of rigid links connected with joints (Khalil & Dombre, 1999). In general, serial robots are used in the industry due to the following advantages (Chaoui, 2020):

- Their kinematic models are relatively simple
- Human operators can easily identify themselves with an open-loop kinematic chain (compared to a human arm, see Figure 1.4)
- Each actuator benefits from total independence
- Inverse and forward kinematic models are well known, and the dynamics have been thoroughly analyzed for several cases

However, the use of serial robots has certain disadvantages:

- Serial manipulators require an actuator for each joint, thus creating unwanted inertia.
- The speed reducer and the drive shafts may deflect when torque or force is transmitted, causing unwanted behaviors.
- Singularities can occur, for which redundant robots may be better suited to overcome

Figure 1.4 depicts a comparison between anatomical and kinematical capabilities of the human arm, with those of a generic industrial robotic arm (ω_i, and $Joint_i$ correspond to the DOF).

This type of robot is used to perform many tasks: packaging, assembly, inspections in sensitive areas, part control, laser cutting; and above all, they make it possible to carry out tasks that would be very difficult and risky for humans. As a rule, these are application-specific, and they exclude humans from the field of work and have their own workspace.

Now, these industrial robots could become more "collaborative" with the operators when their workspace is fitted with sensors that detect when human motion cross certain barrier. The purpose of the sensors is to ensure the safety of the operator, by, either reducing their speed, or stopping completely the robot's motion. In the latter case the robot can restart operation only when the worker leaves the designated workspace; while if the speed is only reduced, obstacle avoidance or collision-free trajectories should be executed by the robot.

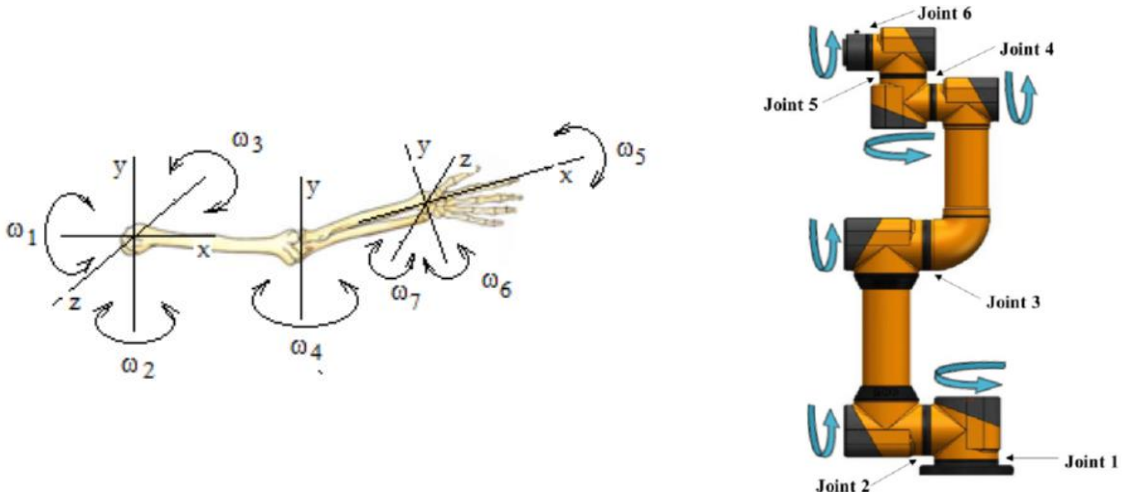


Figure 1.4 Human arm with 7 DOF vs serial robot of 6 DOF..

As stated before, collaborative robotics (also known as cobotics) is halfway between industrial robotics and service robotics. The field combines the ability of humans to cope with uncertainties of the workspace and the task, with the precision, strength, and stability of robots (Blanchet, 2021).

It should be noted that in the concept of collaboration we find as well interaction and shared intention (Grosz, 1996). This means that all the parts intend to work together towards a

common goal or objective. In this regard, a new generation of robots were designed and developed, to meet and explore new capabilities of interaction between industrial robots and humans. Figure 1.5 shows different types of Human-robot collaboration depending on the task in hand, depicting by semicircles their respective workspaces.

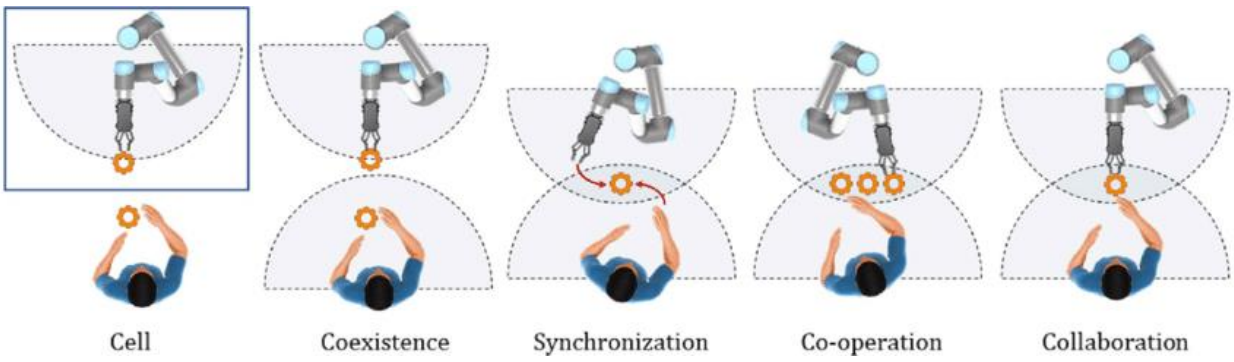


Figure 1.5 Engagement level between human and collaborative robot (Malik & Bilberg, 2019)

Additionally, collaborative robots are easily programmable thanks to their modern interfaces (El Zaatari et al., 2019) and lighter than conventional industrial robots. They are suitable to constantly change environments or workspaces, whenever required. They are developed to occupy the same workspace as humans and interact with them, hence they must comply with ISO/TS 15066:2016 safety recommendations, by limiting their speed or incorporating various safety measures.

1.3.1. Safety requierments and risk assesments

Depending on the engagement level of collaboration, some safety requirements should be met. According to the technical specification ISO/TS 15066:2016, the collaborative functionality is a state in which an operator and a robotic system work within a collaborative space.

The collaboration scale can range from no direct human-robot contact to a robot that reacts to operator motion in real time and adjusts its own motion accordingly (see Figure 1.6).

Depending on the complexity of interaction, more intrinsic safety features are needed. This means that control laws and external sensors shall comply foreseen tasks, considering risk assessment in order to assure the wellbeing of operators.

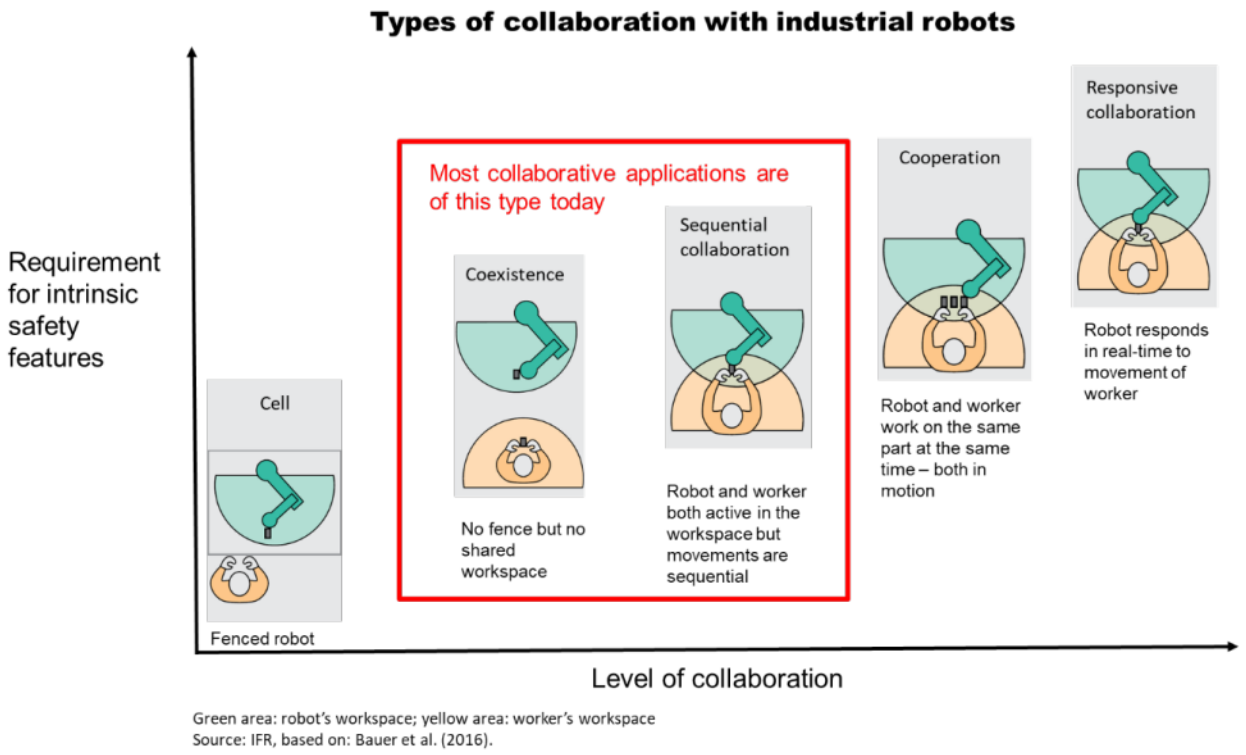


Figure 1.6 Types of Human-robot collaboration and safety requirements (adapted from (IFR, 2019))

More detailed information (with slightly different classification) can be found in the technical specification ISO/TS 15066:2016, as it is shown in Table 1.1. This specification provides general guidelines that shall be met in order to reduce the risk of potential accidents when robots operate in either of the four types of collaborative operations: 1) Safety-rated monitored stop, 2) Hand guiding, 3) Speed and separation monitoring, and 4) Power and force limiting.

A combination of operations can also be possible, in that case, the overall security requirements associated shall comply. Risk assessments will indicate safe values for robot's operating speed, separation distance, power or force, depending on the collaborative operating mode used.

Generally speaking, the robot needs to stop motion, in order to guarantee the safety of the operator when certain conditions are no longer respected, for example: when the speed determined by the risk assessment is no longer respected or when the robot moves in situations where it must be stationary.

Table 1.1 Type of collaborative operations according to ISO/TS 15066:2016

Type of collaborative operation	1) Safety-rated monitored stop	2) Hand guiding	3) Speed and separation monitoring	4) Power and force limiting
Main means of risk reduction	• No robot motion when operator is in collaborative space	• Robot motion only through direct input of operator	• Robot motion only when separation distance above minimum separation distance	• In contact events, robot can only impart limited static and dynamic forces
Typical measures	• Supervised standstill • Stop in case of fault • Robot resumes after the operator had left workspace	• Controls close to end-effector • Input means for motion commands • Emergency stop	• Supervision of distance and speed • Protective stop if minimum separation is violated • Speed is lowered at least to the limit speed (safety-rated) • Consider braking distance	• Low inertia, suitable geometry and material, control functions to limit speeds, torques, sensory input
Common applications	• Loading/unloading end-effector	• Lift assist • Ergonomic adaptation • Load positioning	• Working in common area on separate tasks	• Applications in mixed environments • When frequent operator presence is required
Incidental physical contact	• -	• -	• -	• Transient (<50ms) • Quasi-static • (See Figure 1.7)

As shown in Figure 1.7 and the previous table, the two cases of incidental physical contact are to take into consideration when the operator and the robot require to share their workspace repeatedly. The limit values critierias depend on the peak forces, pressures, and stresses. The viability for each case scenario (free or constrained) depends on the foreseen design or control, by effectively considering the relative speed, contact area, effective mass of the robot, payload, joint forces, and duration of contact.

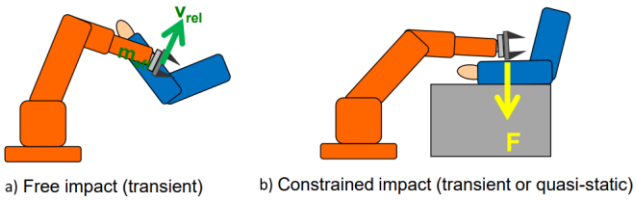


Figure 1.7 Types of incidental physical contacts (adapted from (Matthias, 2015))

Figure 1.8 shows results of free transient contacts reported in (ISO/TS, 2016). Different body regions were tested for pain levels, by converting peak pressure limits to effective robot mass. As it can be seen, as the effective robot mass increases, the acceptable robot speed limit decreases, until they arrive to apparent asymptotes. This can give us an idea of the given

operational maximum speed at which the robot could acceptably enter in transient contact (<50ms) with the human operator.

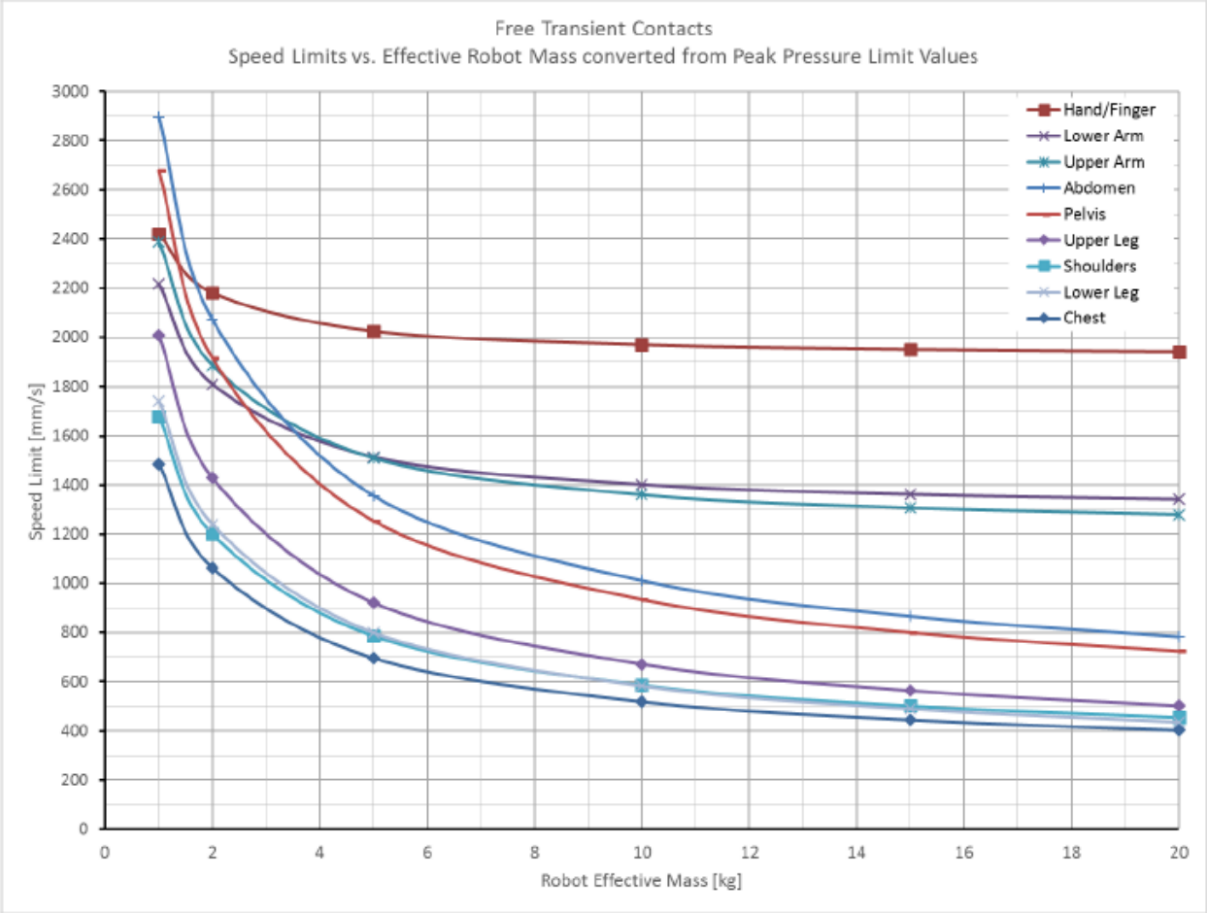


Figure 1.8 Speed limits vs. effective robot mass converted from peak pressure limit values (ISO/TS, 2016)

1.3.2. Collaborative robots' market

According to (IFR, 2020), 2.7 millions of industrial robots are operating in factories around the world. And in recent years, factories have been increasing their adoption of collaborative robots (see Figure 1.9). The IFR states that cobots installations grew by 11% in 2019, as a result of increasing collaborative solutions and range of applications offered by the new trends. They indicate that even though the market is growing rapidly, it is still in beginning stages (cobots had 4.8% market share of the 373,000 industrial robots installed in 2019), meaning that these are relatively new grounds and there is still too much to explore and adapt.

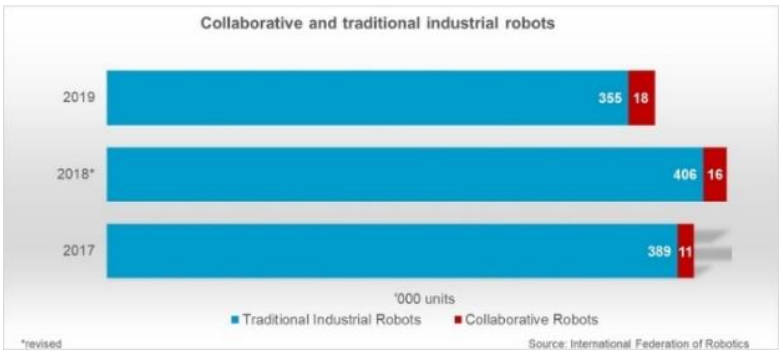


Figure 1.9 Collaborative and traditional industrial robots (IFR, 2020)

Even though there has been an impact on this growth due to the COVID-19 pandemic, collaborative robots sales forecast is displayed in Figure 1.10 based on the market size from 2020 and 2021. In 2020 it was estimated in 590.5 million USD and it is predicted to reach a value of almost 2,000 million USD by the year 2030.

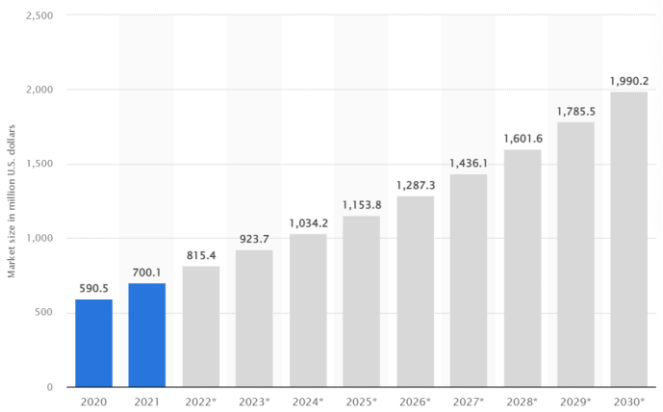
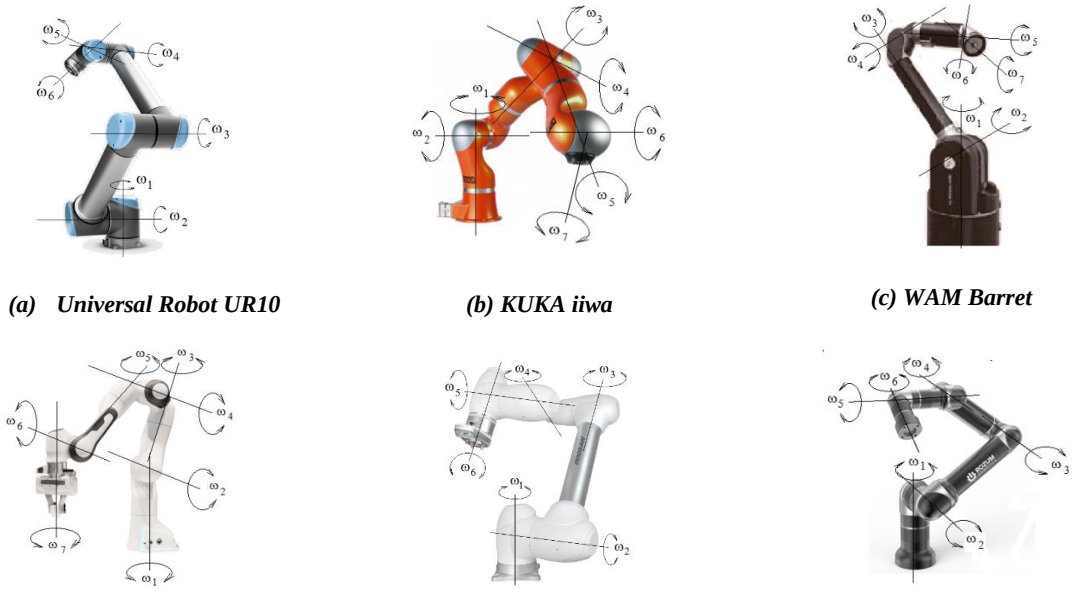


Figure 1.10 Global collaborative robots market size in 2020-2030, *Forecast reported in (Statista, 2022)

Figure 1.11 depicts some of the collaborative robots currently on the market. Some of these robots have more DOF than others, but they all possess the ability to work with humans without the need for additional protective barriers. In fact, most of them are designed to achieve a collaborative mode of operation by limiting their speed, their power and their force by design or control.



(a) Universal Robot UR10 *(b) KUKA iiwa* *(c) WAM Barret*

(d) **Panda** *(e) Doosan M 1013* *(f) ROZUM*

Figure 1.11 Examples of collaborative robots and their degrees of freedom (modified from (Staretu, 2021)).

1.4. Universal Robots

A first prototype model of industrial collaborative robot was developed in 1998, designed to help employees to load and unload car doors in the automotive industry (Wannasuphoprasit et al., 1998). However, the development and adoption of cobots was not well spread around the world. It was until 2008 when the Danish robotic enterprise Universal Robots pushed the cobots adoption by introducing the first anthropomorphic collaborative robot arm to the market: the first lightweight, flexible UR5 robot (UR, n.d.-a),

Later, in 2012, they launched the UR10 series, targeting manufacturers with larger tasks with lifting ability of 10kg and 130cm of reach. And in 2018, the co-founder Esben Østergaard is awarded with the *Engelberger Robotics Award*, because of the significant technological brakethrough that the company apported to the robotics community. 2018 is also the year when the new e-series is launched, improving the past robotic models to meet the new necessities in the market (UR10e is shown in Figure 1.12, with an increased payload of 12.5kg).

Nowadays, Universal Robots stands as one of the most important robotic companies that specialize in collaborative robots' solutions and robotic equipment.

Figure 1.12 Universal robot UR10e

1.4.1. Programming

URControl is the low-level controller that comes with the UR robot models' Control Box. PolyScope or Graphical User Interface, is the software that connects as a client using a local TCP/IP connection. One can program directly through they graphic interface and give a series of sequential commands or by sending URscripts to the TCP/IP connection directly. But some may argue that is not the most practical way to develop and test complex tasks.

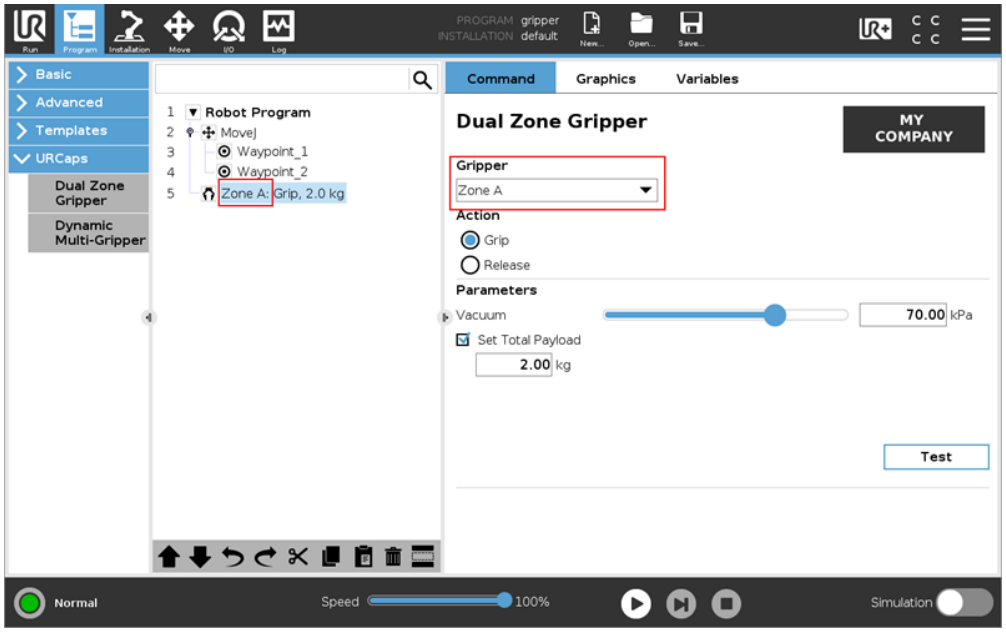


Figure 1.13 PolyScope software from Universal Robots (ROS, 2021)

There exist some other options for programming robots such as the UR10e, by taking advantage of this connection TCP/IP. Some examples of offline programming suits are ArtiMinds RPS and RoboDK, which are paid solutions.

Although these solutions are viable candidates to program robots such as the UR10e, in this thesis we opted for the Robotic Operating System (ROS). ROS is an open-source set of software frameworks for robots that over the past decade has gained much acceptance in the academic and research community. It has several advanced features that enable robotic researchers to work more efficiently towards the fulfillment of their respective goals. The modules developed in ROS can be reused between different robotics applications, for instance one path planning algorithm could be easily adapted from what is developed in the UR10e to any other robotic arm, just by changing some minor configuration parameters. Figure 1.14 shows the GitHub repository with the required libraries to synchronize ROS with the UR10e robot, and Figure 1.12, the language supported to develop user ROS modules: C++ and Python.

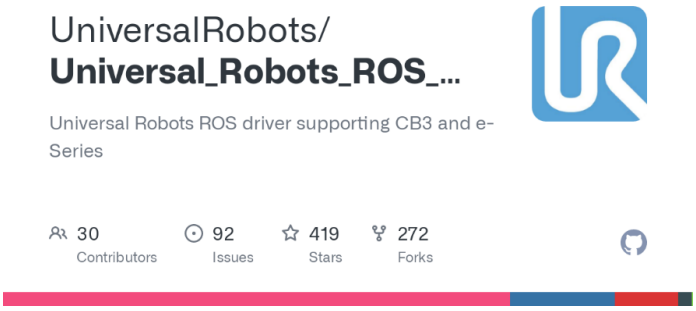


Figure 1.14 Universal Robots ROS Driver [github]

Figure 1.15 Languages used in ROS: C++ and Python

1.5. Thesis organization

The rest of the thesis can be divided in 3 parts:

Part I. Harmonic Drive study

Chapter 2. It presents the problematic behaviors of Harmonic Drive gearboxes, which are used in many robotic applications, such as in joints of collaborative robots. The methodology to obtain load-dependent friction laws is presented. And the results from tests executed in three different Harmonic Drive models.

Part II. Obstacle avoidance study

Chapter 3. This chapter presents a state of the art of the obstacle avoidance trajectory problem in robotics. The methodology used with the Dubins curve variation is introduced for the obtention of arc based path planning. The resulting set of equations are presented. As well the simulation framework with ROS and MoveIt! is introduced, testing and validating the resulting trajectories, evaluated in terms of final times, total distance traveled and average speeds.

Chapter 4. In here the trajectories obtained with the simulation are executed and obtaining the initial experimental results. The issues encountered in the simulation framework are then addressed, which leads to the control of the end-effector linear speed, so it complies to the security requirements at the optimum maximal speed. The results, from three different sets of trajectories, are presented and compared to the original ones.

Part III. Conclusion

Chapter 5. General conclusions on the research work are carried out in the framework of this doctoral thesis. This thesis achieved the first steps towards the improvement of the friction model of Harmonic Drives and satisfactory initial results towards the improvement of controlling the end-effector linear speed towards obstacle avoidance, with the methodology here proposed. Finally, topics for future researches are discussed, to further improve the collaboration between robots and humans.

Chapter 2

2. Load-dependent friction characterization of strain wave gears

2.1. Introduction

Strain wave gears (or Harmonic Drives) are special types of speed reducer mechanisms exploiting the elastic deformation of a metallic clutch as a basic functional principle. They achieve a high reduction ratio in a single stage with few parts and possess specific characteristics such as a remarkable compacity, high torque mass (volume) ratio, a virtually null backlash, input-output coaxiality and hollow shaft, which allow a wide variety of applications. However, several inherent properties are known to generate specific defaults such as a complex friction torque, kinematics irregularities and vibrations, and a somewhat complex stiffness response. Moreover, the simplicity of construction hides subtle mechanical effects which complicate its modeling.

This chapter recalls the principle of the Harmonic Drive speed reducer as well as its main advantages and focuses on the problem of friction torque in the context of the control of collaborative robots.

2.1.1. Harmonic Drive in collaborative robots

The unique set of characteristics prior listed explain its success in the field of robotics, extensively used for decades since a high reduction ratio paired to a high compacity and low mass allow designing lightweight robotic arms.

Figure 2.1 UR10e and KUKA LBR IIWA 7 R800 collaborative robots, using Harmonic Drive gearboxes

Some undesired behaviors have been reported in the literature and this chapter address the influence of the friction torque transmission at low speed, primarily for the purpose of applications in collaborative and light weight robotic manipulators where Harmonic Drives are used, such as in the Universal Robots and KUKA brands (see Figure 2.1).

The context of the study is the linearization of the torque transfer law in Harmonic Drive gearboxes to allow improving their torque control and potentially the position/speed control loop. In collaborative robots the accuracy of torque control is especially important at low speed to improve the control of contact forces. Thus, we will focus only on the dependence of friction torque at low speed and its compensation.

The main elements of this chapter were presented in an international conference in July 2021 (Ponce Quiroga et al., 2021).

2.2. The Harmonic Drive gearbox

The Harmonic Drive was invented in the 1950s by Walton Müsser for aerospace purposes and it belongs to the family of strain wave gears. Its design, construction and operation is unique: it has teeth in full contact at two diametrically opposite ends at any moment, and the arrangement of their elements grant several interesting properties such as compact and lightweight structure, high gear ratio, hollow shaft possibility, coaxial input-output shaft, and a negligible backlash, reasons for which they are still widely applied in aerospace and robotics applications (Kircanski & Goldenberg, 1997). However, they also present some problematic behaviors.

To further understand the Harmonic Drive, we first must look at its internal structure, how it works, and then discuss on these problematic behaviors.

2.2.1. Structure and operation of Harmonic Drives

There are three main structural elements that constitute the Harmonic Drive: the wave generator, the flexspline and the circular spline (Figure 2.2 depicts the 3 main components and Figure 2.3, the internal configuration of the Harmonic Drive gearbox)

Figure 2.2 Harmonic Drive gearbox main components: 1) Wave generator, 2) flexspline and 3) circular spline [2].

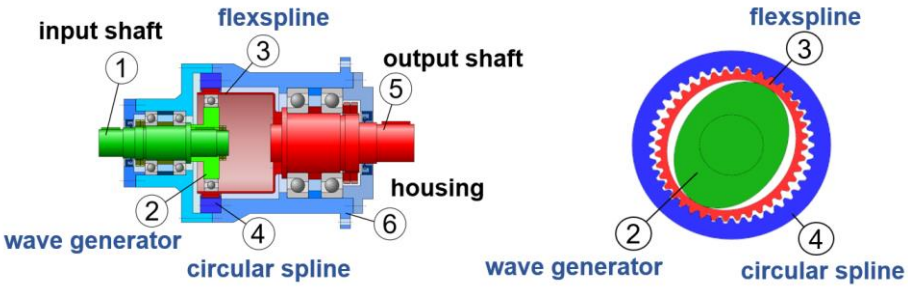


Figure 2.3 Harmonic Drive gearbox structure

The wave generator consists of an elliptical ball bearing assembly with a small eccentricity that is inserted into the bore of the externally toothed flexspline. The flexspline (or flexible spline) is a thin-walled hollow cup made of alloy steel and takes advantage of the elastic properties of the material to enclose the elliptical shape of the wave generator. The outer teeth of the flexspline engage with the inner teeth of the crown (or circular spline) along the major axis of the ellipse while they are disengaged along the minor axis.

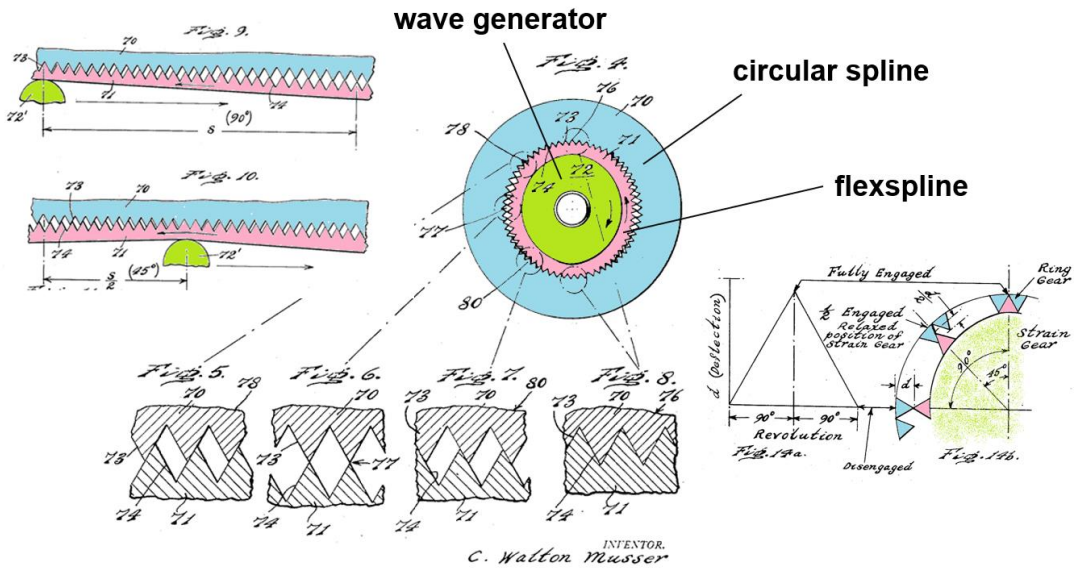


Figure 2.4 Harmonic Drive patent drawings depicting teeth engagement

Figure 2.4 shows patent drawings from Walton Musser that were originally submitted in 1955 (Müsser, 1959); here we can appreciate more clearly how at two diametrically opposed positions the teeth are fully engaged, from these positions, at 45° degree, teeth are half-engaged, and at 90°, fully disengaged. Generally speaking, this means that there are many more engaged teeth than in a traditional gear.

In a standard Harmonic Drive configuration, the wave generator induces the rotation of the flexspline in an opposite direction relative to the circular spline due to the difference in the number of teeth, which leads to the definition of its input/output velocity ratio (reduction ratio) formula.

2.2.2. Problematics and drawbacks of Harmonic Drives

Despite all the advantages that this gearbox structure may provide, its performance is not perfect due to this unique configuration. Several issues in the Harmonic Drives have been identified and studied, such as kinematics, elastic deformation, contact pressure, friction, fatigue, initially by its inventor; later, researchers study the kinematic error (Gandhi et al., 2002; Ghorbel et al., 2001), hysteresis (Gandhi et al., 2002; Yamamoto et al., 2009), inertia, and a somewhat complex friction behavior (Hauschild et al., 2004).

The friction and kinematic error have been identified as the main attributes responsible for the transmission performance degradation, which is the reason why they have been studied extensively in previous decades (Tuttle & Seering, 1996; Zou et al., 2017).

The motivation of this research is then to characterize the torque losses they have a direct impact on the input/output torque transmission.

2.3. Friction torque modeling and identification

The friction torque generated in a speed reducer is essentially the consequence of the contact friction on the particular geometries/kinematics of its moving parts. Since the contact load increases the resulting friction increases, it is inferred that a variable load necessarily leads to a variable frictional torque.

Friction torque can be an asset (bolts and nuts), but regarding transmissions they are generally disruptive, which is the reason many studies try to identify friction models to later

compensate these disruptions by the means of new mechanical designs, the use of special lubricants or with control compensation.

2.3.1. Contact friction laws – Tribology

The contact friction law involved in mechanical models is mostly the Coulomb law (dry contact) or the Stribeck law (lubricated contact). In mechanical design the Coulomb law is used either in static (adherent/static friction coefficient) of for sliding parts (kinematic friction coefficient) (Spenlé & Gourhant, R., 2007). In tribology, the Stribeck law expresses the kinematic friction coefficient as a function of the Sommerfeld/Hershey parameter which combines the viscosity of the lubricant, the contact pressure, and the sliding speed. Often in robotics, authors use a simplified Stribeck law where only the speed influences the friction.

2.3.2. Friction torque model of gearboxes

In his initial patent, inventor Walton Müsser (Müsser 1959) proposed a first torque transfer model from which a linear friction torque model can be deduced.

For this, he used two inclined planes in series, to represent the effect of the elliptical ball bearing cam (wave generator) deforming the flexspline and a pair of engaged teeth. He then assigned each contact a coefficient of kinematic friction using Coulomb's law for the pair of teeth and even for the rolling contact of the elliptical bearing. By multiplying the force transfer ratios of the two inclined planes, he determined an expression of the output/input ratio. By forming the difference between the reduction slope and the torque transfer slopes, one can deduce a linear expression of the friction torque of the Harmonic Drive reducer. Actually, when using inclined (or wedges), there are two distinct expressions of the torque transfer ratio depending on the energy transfer direction, each being associated to a distinct torque transfer coefficient (DIRECT or INDIRECT). This leads to two distinct friction torque laws depending on the sense of the speed.

However, in must publications the influence of the load is neglected. The variation of friction torque as a function of contact load is rarely correctly modeled, which could be somewhat problematic when complex systems such as robots have large load variations.

Several studies have been done on the friction torque model of Harmonic Drive gearboxes considering friction as constant (Han et al., 2016; Liao et al., 2016; Ma et al., 2018;

Shi et al., 2017; Tadayoni et al., 2011; Zou et al., 2017). Regarding outside of the Harmonic Drive scope, some other friction models consider the friction as dependent of the load and as well the sense of motion (Abba & Sardain, 2005; Hamon et al., 2010; Hamon, P., 2011; Kammerer & Garrec, 2013).

2.3.3. Force Transfer Diagram identification

The Force Transfer Diagram is a parametric representation of the equilibrium of a gearbox (or an actuator) submitted to an input and output torque, for various conditions of movement (speed and acceleration). Considered in its broader reach, this diagram is used to characterize the dynamic model of a gearbox or actuator.

In quasi-static, its interest relies in revealing the influence of contact friction, which correspond to our first objective. The diagram is directly accurately identified for any type of mechanism by plotting a set of equilibrium points on a two-dimensional graph, using counterweights, levers and pulleys (see Figure 2.5)

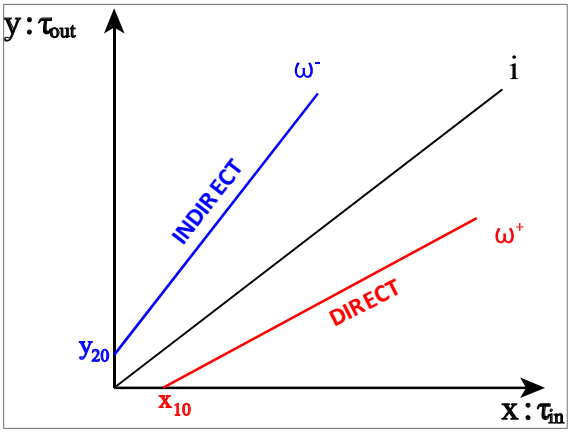


Figure 2.5 Force Transfer Diagram notional shape

τ_{in} represents the input torque applied to the gearbox.

τ_{out} represents the output torque of the gearbox.

The black line i (THEORETICAL) corresponds the reduction ratio of the gearbox i.e. the torque transfer law in the absence of friction.

The red line (at the right of i) represents the DIRECT direction (transmission of energy from the input to the output).

The blue line (at the left of i) represents the INDIRECT direction (transmission of energy from the output to the input (Garrec, 2011)).

$\omega^+$and ω^-represent the direction of rotation of the gearbox referring to the DIRECT and INDIRECT slopes.

$x_{10}$and y_{20}represent the DIRECT and INDIRECT thresholds.

Experience shows that the DIRECT and INDIRECT transmission lines are not parallel to the reduction ratio (THEORETICAL) and that there are offsets from the origin (thresholds).

The ratio (DIRECT slope/reduction slope) gives the DIRECT force transfer coefficient. The ratio (reduction slope/ INDIRECT slope) gives the INDIRECT force transfer coefficient. When the INDIRECT coefficient is positive, the system is reversible and irreversible when it is negative. When both coefficients are equal, the mechanism is qualified as symmetrical, and asymmetrical when they differ. A screw is potentially asymmetrical, excepted for a lead angle of 45°. With a spur gear, the coefficient of transfer deduced from the theoretical sliding formula of meshing teeth and the coefficient of sliding friction is equal for both sense meaning that a spur gear is symmetrical.

The efficiency can be defined as the ratio of output torque with friction/theoretical output torque only if it is positive and assuming an exact constant reduction ratio. Besides the existence of thresholds, force transfer coefficients should not be confused with efficiencies.

On any given mechanical transmission, the friction is acting against the motoring effort whether it is applied at the input or output. This means there are potentially two distinct friction torque laws DIRECT (motoring) and INDIRECT (regenerating) implying corresponding friction torques. Previous works (Garrec, 2011; Garrec et al., 2004) have established that these laws can be deduced from the Force Transfer Diagram of the gearbox. As depicted on Figure 2.5, friction torque laws are obtained as the differences between DIRECT and INDIRECT characteristics and the reduction ratio slope.

2.4. Harmonic Drive FTD identification setups

Two experimental setups are designed and adapted to obtain the corresponding Force Transfer Diagrams of three different types of Harmonic Drive. Table 2.1 presents the main specifications for each one of the tested Harmonic Drive, including the ideal velocity ratio, rated torque, repeatable and momentary peak torque, and Figure 2.6 shows the schematics of these three Harmonic Drives.

In Setup A two types of Harmonic Drives are tested and in the Setup B, the third type.

Table 2.1 Harmonic Drives specifications

Setup	Harmonic Drive	Size	Vel. ratio i	Rated torque T_N [Nm]	Repeatable peak torque T_R [Nm]	Momentary peak torque T_M [Nm]
A	HFUC-20-2UH-100 (Cup type)	20	100	40	82	147
A	HFUS-20-2SO-100 (Mushroom type)	20	100	40	82	147
B	HDUR-32-IH-120 (Differential)	32	120	137	353	686

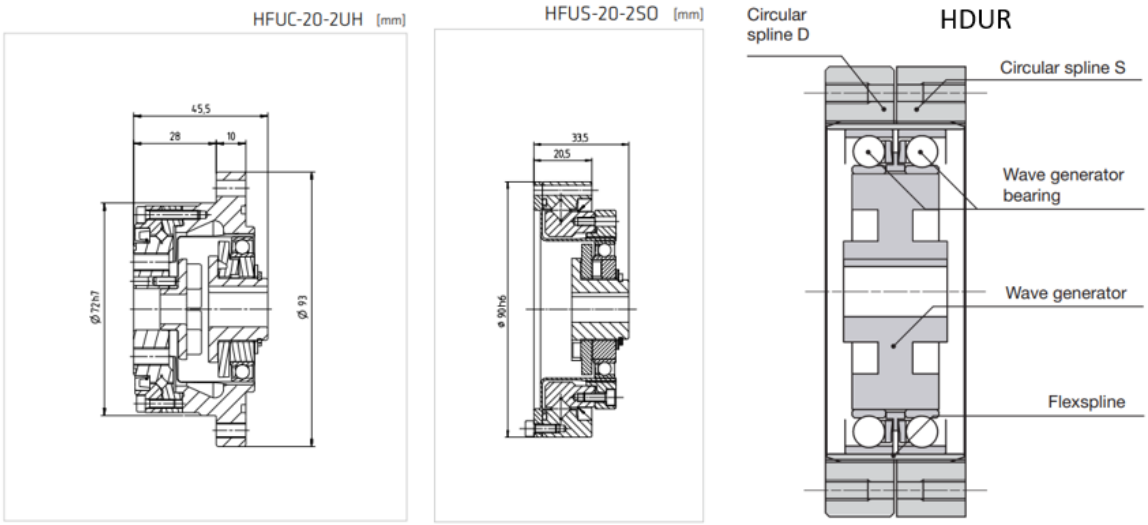


Figure 2.6 Harmonic Drive schematics for HFUC, HGUS and HDUR types

2.4.1. Setup A: Workbench

The transfer diagram is drawn from data experimentally acquired in quasi-static conditions. Setup A is a conceived the workbench to measure the torque values at the input and output for two different Harmonic Drive gearboxes: HFUC-20-2UH-100 and HFUS-20-2SO-100. This workbench (see Figure 2.7 and Figure 2.8) allows to obtain the corresponding input and output torques by adding weight at the input pulley and output pulleys of the gearbox: more specifically, it reproduces the effect of the torques (refer to section 2.4.3 for the identification procedures).

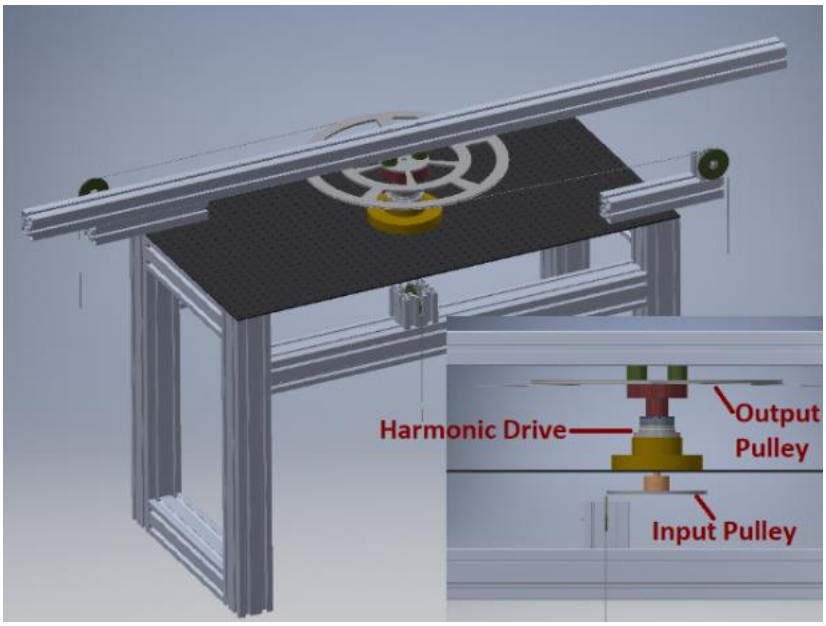


Figure 2.7 Setup A: Workbench 3D CAD model

Figure 2.8 Setup A: Workbench designed for testing

In Figure 2.7, we introduced one pulley with $r_{in} = 33\,mm$ below the table to induce a torque at the input. We used another pulley with $r_{out} = 250\,mm$ above the Harmonic Drive in the output. On the latter, we attached cables at both extremities. The cables go in opposite directions to exert equal forces in both sides: they generate "pure torques" at the output of the gearbox.

2.4.2. Setup B: Differential Harmonic Drive joint-link

A second test bench is developed to test a differential Harmonic Drive gearbox (HDUR-32-IH-120) included in a special type of joint developed earlier at CEA. The HDUR principle is already described in the original patent of Walton Müsser dated from 1955 as a variant of the cup

model. In this special joint, two units are symmetrically mounted in the branches of the fork articulation, and both are driven by a unique shaft. We neglect here a potential slight preload in the mounting and its effect on the friction on each unit. Similarly, to Setup A, we draw the transfer diagram in quasi-static conditions to determine the load-dependent friction performance, assuming it is representative of a virtual gearbox that would have twice the capacity of each.

Joint schematics including two units of the differential HDUR gearbox is shown in Figure 2.9 (the Harmonic Drives appear in red).

Figure 2.10 presents the Setup B. With this configuration we can add weight at the input (directly from a pulley in the input axe joint) and at the output (at a given distance on the bar) to generate the input and output torques.

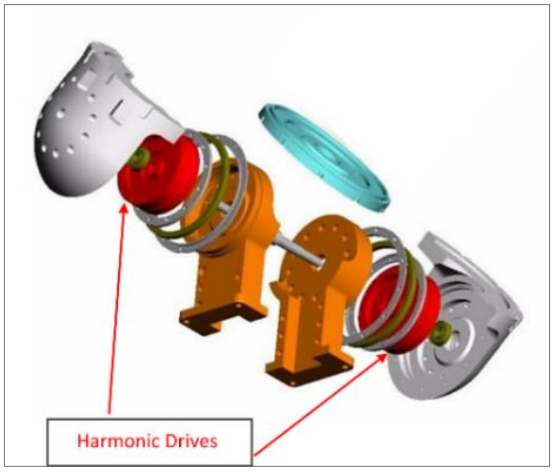


Figure 2.9 Fork joint arrangement using a pair of HDUR gearboxes working in parallel.

Figure 2.10 Setup B: fork joint arrangement adapted for testing

2.4.3. Identification of the Force Transfer Diagram

2.4.3.1. Measurement method

As stated in section 2.4.1, these setups allow performing measurements by adding weight at the input and output, and afterwards, these weights are used to compute the torque values. The material used for the measurements: calibrated weights and datasheets for computations.

There are three different procedures to be performed:

a) INDIRECT threshold obtention:

Adding known weights at the output without weight at the input until the system starts moving continuously.

b) DIRECT threshold obtention:

Adding known weights at the input without weight at the output until the system starts moving continuously.

c) DIRECT and INDIRECT weight values obtention at the input, at a given output weight:
   1. Adding a known weight at the output.
   2. For the known weight, weight is added at the input until the system starts moving continuously. The value is registered (DIRECT input weight value).
   3. The weight is to decrease at the input until the output starts moving in the other direction. This new value is registered (INDIRECT input weight value).
   4. Repeat steps 1-3 by changing the weight at the output.

Equations (2.1) and (2.2) are used to calculate the torque values for the Setup A.

$$\tau_{in}{}^{\{D,I\}} = m_{in}{}^{\{D,I\}}\, r_{in}\, g \tag{2.1}$$

$$\tau_{out} = m_{out}\, r_{out}\, g \tag{2.2}$$

Where:

$\boldsymbol{m_{in}}$= Input mass

$\boldsymbol{m_{out}}$= Output mass

$\boldsymbol{r_{in}}$ = Radius of the Input pulley

$\boldsymbol{r_{out}}$ = Radius of the Output pulley

$\boldsymbol{g}$ = 9.81 m/s^2

Equation's (2.1) superscript $\{D, I\}$ specifies whether if it is a value coming from a DIRECT measurement or an INDIRECT measurement, respectively. ${\tau_{in}}^{D}$ is the DIRECT input torque, ${\tau_{in}}^{I}$ is the INDIRECT input torque and τ_{out} is the common DIRECT and INDIRECT output torque for the same output mass.

Equation (2.3) is used instead of (2.1) when performing the procedure for Setup B.

$$\tau_{out} = m_{out}\, l_{out}\, g \tag{2.3}$$

Where:

$\boldsymbol{l_{out}}$ = Distance to the center of mass

Figure 2.11, depicts the friction torque derived laws present at a determined output load τ_{LOAD}. Here P_1 and P_2 represent the DIRECT and INDIRECT measurements with coordinates (x_1 , τ_{LOAD}) and (x_2 , τ_{LOAD}) obtained with (2.1) and (2.2) or (2.3); x_{th} is the ideal desired torque, obtained from the reduction ratio i at the given τ_{LOAD}.

The DIRECT friction torque (DFT) and INDIRECT friction torque (IFT) represent the frictions values existing in the gearbox depending on the direction of movement.

When the gearbox is motoring (2.4) shows that DFT is acting in the opposite direction of x_{th} which is why x_1 compensates this friction with respect to the ideal ratio i.

Similarly, IFT is the friction at the gearbox when τ_{LOAD} is actuating in INDIRECT manner (the τ_{LOAD} regenerates towards the gearbox torque). This value can be reflected at the input by the Reflected INDIRECT friction torque (RIFT). In (2.5) it can be seen that RIFT is reducing the input torque when the gearbox is in the regenerating direction.

$$x_1 = x_{th} + DFT \tag{2.4}$$

$$x_{th} - RIFT = x_2 \tag{2.5}$$

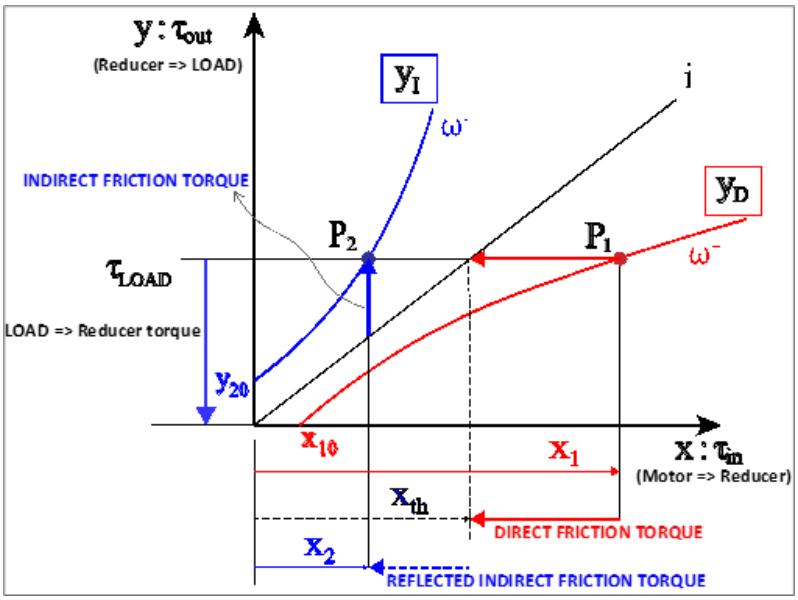


Figure 2.11 Friction torques laws derived from the Force Transfer Diagram of a Harmonic Drive gearbox

2.5. Results and analysis

2.5.1. Load dependent friction of HFUC

Following the established procedure in section 2.4.3, we obtain three data sets with the HFUC Harmonic Drives in order to test the repeatability. Figure 2.12 shows input and output torques. We can observe that three measurements confirm repeatability, as points are almost overlaying each other.

Second order polynomial equations are fitted for the DIRECT (in red) and INDIRECT (in blue) curves. As the load increase, the measurements continue to separate from the theoretical gearbox ratio (black line).

The variation of the three data sets is minimal confirming reproducibility, and polynomial regression coefficient of determination are close to 1.

Table 2.2 presents the experimental data sets. The columns in blue represent the INDIRECT measurements, and in orange, the DIRECT measurements. Each column is numbered corresponding to the experiment data set. First sub-table present the masses and the following the corresponding torques.

Table 2.2 Force transfer measurements of HFUC: raw mass and computed torques

	Indirect **Mass** Measures			Direct **Mass** Measures		
Output	Input 1	Input 2	Input 3	Input 1	Input 2	Input 3
0	-	-	-	0.066 kg	0.068 kg	0.068 kg
0.826 kg	0	0	0	-	-	-
2.452 kg	0.125 kg	0.132 kg	0.135 kg	0.280 kg	0.275 kg	0.276 kg
3.475 kg	0.212 kg	0.211 kg	0.215 kg	0.370 kg	0.367 kg	0.369 kg
7.546 kg	0.514 kg	0.515 kg	0.519 kg	0.732 kg	0.731 kg	0.730 kg
13.658 kg	0.960 kg	0.965 kg	0.970 kg	1.293 kg	1.290 kg	1.286 kg
19.753 kg	1.390 kg	1.400 kg	1.419 kg	1.887 kg	1.885 kg	1.880 kg
25.868 kg	1.812 kg	1.830 kg	1.845 kg	2.515 kg	2.505 kg	2.492 kg
31.974 kg	2.222 kg	2.245 kg	2.265 kg	3.135 kg	3.125 kg	3.117 kg
38.091 kg	2.631 kg	2.650 kg	2.680 kg	3.765 kg	3.755 kg	3.750 kg
44.193 kg	3.030 kg	3.055 kg	3.085 kg	4.419 kg	4.399 kg	4.385 kg

	Indirect **Torque** Measures			Direct **Torque** Measures		
Output	Input 1	Input 2	Input 3	Input 1	Input 2	Input 3
0	-	-	-	0.019 N.m	0.020 N.m	0.020 N.m
2.026 N.m	0	0	0	-	-	-
6.014 N.m	0.037 N.m	0.039 N.m	0.040 N.m	0.082 N.m	0.081 N.m	0.081 N.m
8.522 N.m	0.062 N.m	0.062 N.m	0.063 N.m	0.109 N.m	0.108 N.m	0.109 N.m
18.507 N.m	0.151 N.m	0.152 N.m	0.153 N.m	0.215 N.m	0.215 N.m	0.215 N.m
33.496 N.m	0.283 N.m	0.284 N.m	0.285 N.m	0.381 N.m	0.380 N.m	0.378 N.m
48.444 N.m	0.409 N.m	0.412 N.m	0.418 N.m	0.555 N.m	0.555 N.m	0.553 N.m
63.441 N.m	0.533 N.m	0.539 N.m	0.543 N.m	0.740 N.m	0.737 N.m	0.733 N.m
78.416 N.m	0.654 N.m	0.661 N.m	0.667 N.m	0.923 N.m	0.920 N.m	0.917 N.m
93.418 N.m	0.774 N.m	0.780 N.m	0.789 N.m	1.108 N.m	1.105 N.m	1.104 N.m
108.383 N.m	0.892 N.m	0.899 N.m	0.908 N.m	1.301 N.m	1.295 N.m	1.291 N.m

2.5.2. Load dependent friction of HFUS

Similarly, for the second Harmonic Drive (HFUS), four data sets of measurements are performed. The results are shown in Figure 2.13.

The DIRECT and INDIRECT curves behave in a similar manner as with the HFUC, we can perceive that they are actual curves and no lines with a simple slope.

Polynomial regressions were fitted, and their coefficient of determination are virtually 1, indicating a confident fit.

Table 2.3 Force transfer measurements of HFUS: raw mass and computed torques

	Indirect **Mass** Measures				Direct **Mass** Measures			
Output	Input 1	Input 2	Input 3	Input 4	Input 1	Input 2	Input 3	Input 4
0	-	-	-	-	0.064 kg	0.063 kg	0.065 kg	0.065 kg
0.726 kg	0	0	0	0	-	-	-	-
2.452 kg	0.130 kg	0.136 kg	0.133 kg	0.134 kg	0.277 kg	0.273 kg	0.274 kg	0.273 kg
3.475 kg	0.202 kg	0.210 kg	0.207 kg	0.210 kg	0.363 kg	0.363 kg	0.361 kg	0.364 kg
7.546 kg	0.499 kg	0.515 kg	0.509 kg	0.510 kg	0.733 kg	0.730 kg	0.727 kg	0.725 kg
13.658 kg	0.935 kg	0.952 kg	0.951 kg	0.947 kg	1.305 kg	1.305 kg	1.295 kg	1.293 kg
19.753 kg	1.333 kg	1.360 kg	1.362 kg	1.365 kg	1.945 kg	1.947 kg	1.940 kg	1.937 kg
25.868 kg	1.745 kg	1.747 kg	1.748 kg	1.750 kg	2.615 kg	2.655 kg	2.645 kg	2.655 kg
31.974 kg	2.135 kg	2.125 kg	2.127 kg	2.115 kg	3.295 kg	3.325 kg	3.365 kg	3.345 kg
38.091 kg	2.505 kg	2.515 kg	2.487 kg	2.483 kg	3.995 kg	4.025 kg	4.075 kg	4.065 kg
44.193 kg	2.895 kg	2.905 kg	2.855 kg	2.853 kg	4.695 kg	4.675 kg	4.755 kg	4.765 kg

	Indirect **Torque** Measures				Direct **Torque** Measures			
Output	Input 1	Input 2	Input 3	Input 4	Input 1	Input 2	Input 3	Input 4
0	-	-	-	-	0.019 N.m	0.019 N.m	0.019 N.m	0.019 N.m
1.781 N.m	0	0	0	0	-	-	-	-
6.014 N.m	0.038 N.m	0.040 N.m	0.039 N.m	0.039 N.m	0.082 N.m	0.080 N.m	0.081 N.m	0.080 N.m
8.522 N.m	0.059 N.m	0.062 N.m	0.061 N.m	0.062 N.m	0.107 N.m	0.107 N.m	0.106 N.m	0.107 N.m
18.507 N.m	0.147 N.m	0.152 N.m	0.150 N.m	0.150 N.m	0.216 N.m	0.215 N.m	0.214 N.m	0.213 N.m
33.496 N.m	0.275 N.m	0.280 N.m	0.280 N.m	0.279 N.m	0.384 N.m	0.384 N.m	0.381 N.m	0.381 N.m
48.444 N.m	0.392 N.m	0.400 N.m	0.401 N.m	0.402 N.m	0.572 N.m	0.573 N.m	0.571 N.m	0.570 N.m
63.441 N.m	0.514 N.m	0.514 N.m	0.514 N.m	0.515 N.m	0.770 N.m	0.781 N.m	0.778 N.m	0.781 N.m
78.416 N.m	0.628 N.m	0.625 N.m	0.626 N.m	0.622 N.m	0.970 N.m	0.979 N.m	0.990 N.m	0.984 N.m
93.418 N.m	0.737 N.m	0.740 N.m	0.732 N.m	0.731 N.m	1.176 N.m	1.185 N.m	1.199 N.m	1.196 N.m
108.383 N.m	0.852 N.m	0.855 N.m	0.840 N.m	0.840 N.m	1.382 N.m	1.376 N.m	1.399 N.m	1.402 N.m

If we compare Figure 2.12 and Figure 2.13, we can perceive that the latter has slightly more variability in the last three measurements (rows) in the DIRECT curve, probably due to the difference in models, assembly, constraints, and properties specifications. Similarly, Table 2.3, present the raw masses and computed torques corresponding to each of the experiments.

2.5.3. Load dependent friction of HDUR

Even though the different composition and complexity of the arrangement in the differential HDUR joint, we expected it to perform in a similar curved manner as shown in previous tests with Setup A.

After just the first 3-4 measurements we could deduce that the behavior is similar and even more palpable: the system behavior would diverge in a curved manner, both in the DIRECT and INDIRECT measurements. This can be seen in Figure 2.14.

Analogously, two second order regressions have been obtained to describe the DIRECT and INDIRECT curves with a coefficient of determination close to 1.

Table 2.4 Force transfer measurements of HDUR: raw mass and computed torques

	Indirect Mass Measures	Direct Mass Measures
Output	Input	Input
0	-	0.306 kg
6.316 kg	0	-
13.325 kg	0.544 kg	1.795 kg
18.325 kg	0.845 kg	2.375 kg
23.325 kg	1.125 kg	3.005 kg
28.325 kg	1.375 kg	3.695 kg
33.325 kg	1.585 kg	4.355 kg
38.325 kg	1.815 kg	5.154 kg
43.325 kg	2.025 kg	5.868 kg
48.325 kg	2.255 kg	6.820 kg
52.325 kg	2.425 kg	7.425 kg

	Indirect Torque Measures	Direct Torque Measures
Output	Input	Input
0	-	0.300 N.m
62.515 N.m	0	-
131.899 N.m	0.534 N.m	1.761 N.m
181.390 N.m	0.829 N.m	2.330 N.m
230.882 N.m	1.104 N.m	2.948 N.m
280.373 N.m	1.349 N.m	3.625 N.m
329.865 N.m	1.555 N.m	4.272 N.m
379.356 N.m	1.781 N.m	5.056 N.m
428.847 N.m	1.987 N.m	5.757 N.m
478.339 N.m	2.212 N.m	6.690 N.m
517.932 N.m	2.379 N.m	7.284 N.m

In Figure 2.14, the curvature of transfer function is steeper, compared to Figure 2.12 and Figure 2.13. This may be explained by the difference of construction because the differential principle used, the strain gear is not a single unit. Instead, it is composed of a cylindrical toothed

portion engaging in a ring that has the same number of teeth. During operation, even if the relative angular motion of these two parts is null, there are some torque losses induced by the radial movement of the teeth.

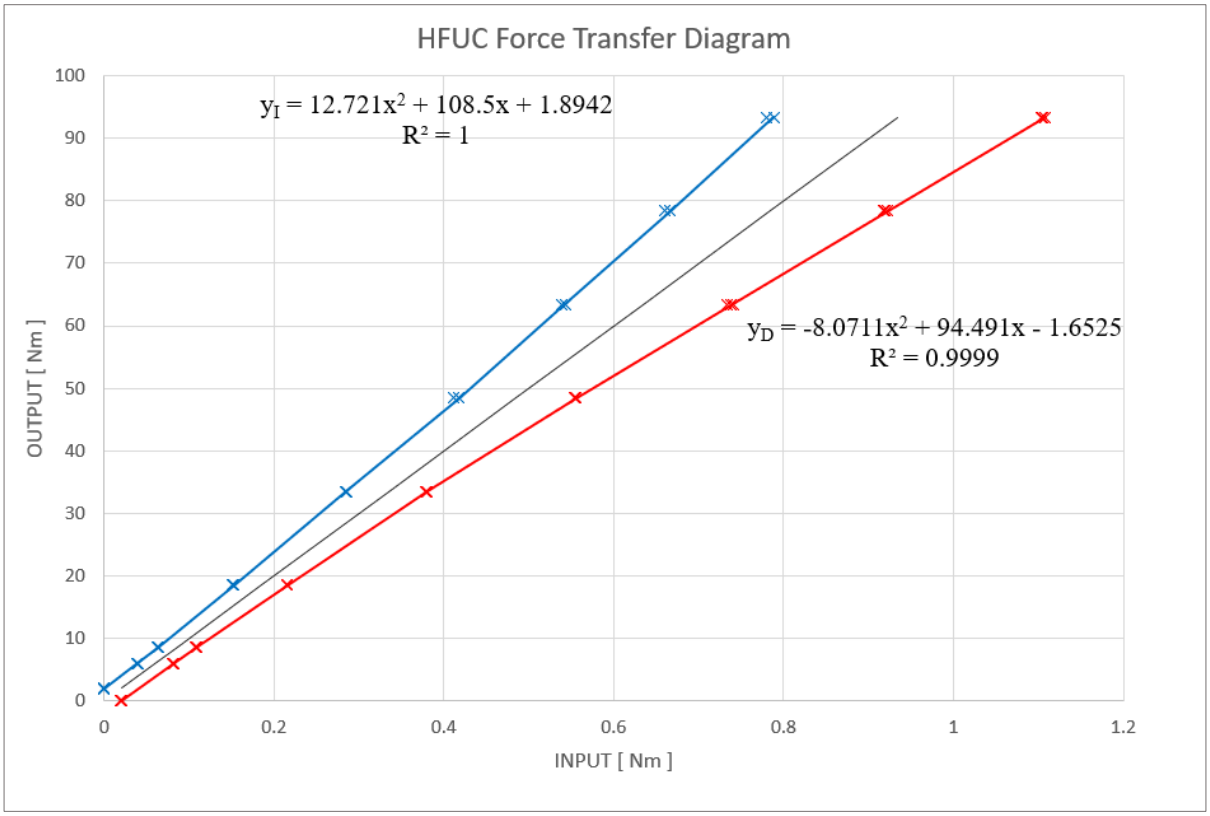


Figure 2.12 HFUC transfer diagram

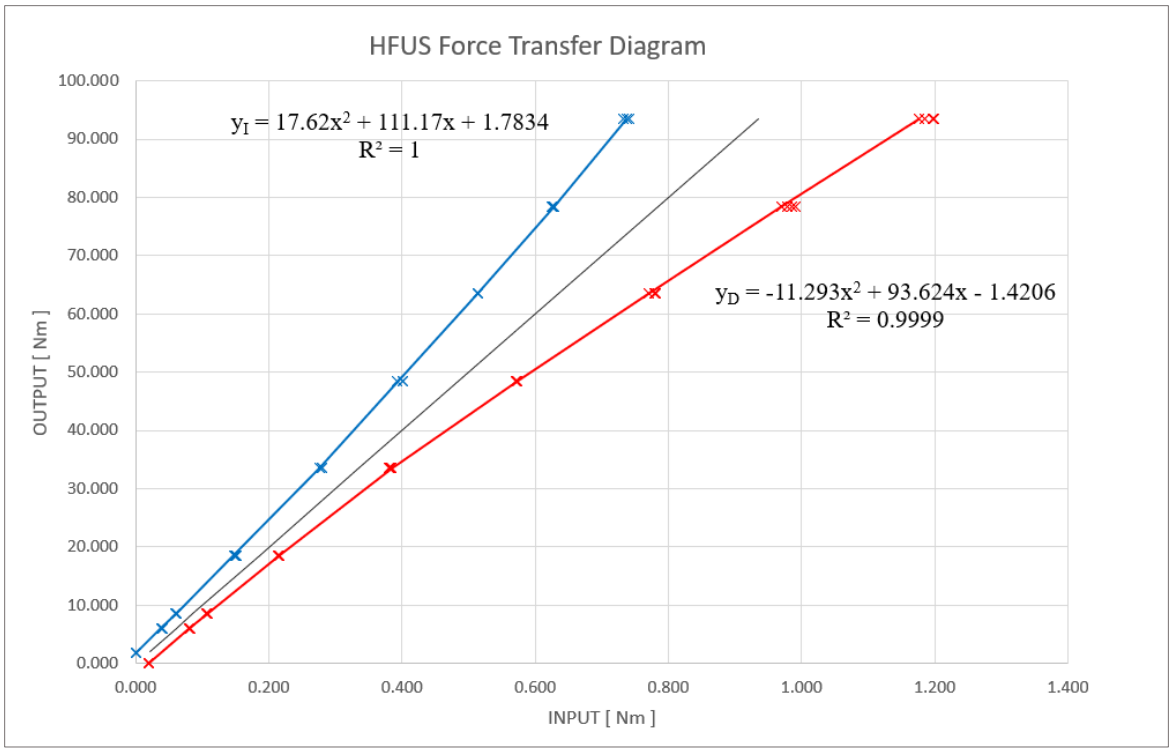


Figure 2.13 HFUS transfer diagram

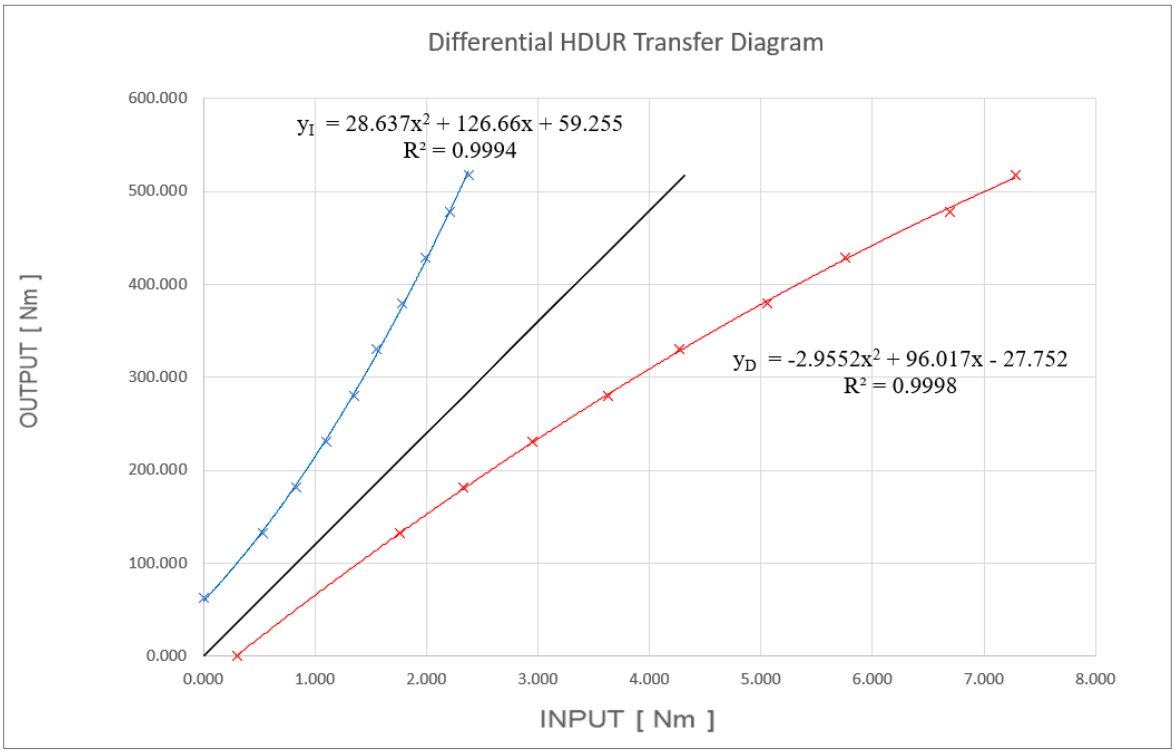


Figure 2.14 Differential HDUR transfer diagram

2.6. Conclusions and perspectives

In this chapter, it has been presented new parabolic friction torque laws for Harmonic Drive gearboxes thanks to the identification of their empirical Force Transfer Diagrams. These findings are new to our knowledge (with respect to the literature) and relevant for further studies. The simplicity of the quasi-static method used has however produced highly reproductible results and we consider it as a reliable basis to elaborate torque friction compensation models for control purposes.

Initially, we would have expected to obtain a linear relationship of the DIRECT and INDIRECT curves, as they have been in other type of transmissions, such as in the industrial robot Stäubli RX130L. Regarding Figure 2.15, we can appreciate in the left the linearity of the friction torque law at the axe three of the Stäubli RX130L, and in the right the new parabolic friction torque law found in the Harmonic Drives.

These torque friction models can be further used to improve the linearity of joint torque control at low speed by compensating the electromagnetic motor torque (current). In the context of collaborative robotics, we are expecting a more sensitive detection of undesired contacts and a reduced resistance when operating a transparent control mode whatever the load and the sense of movement.

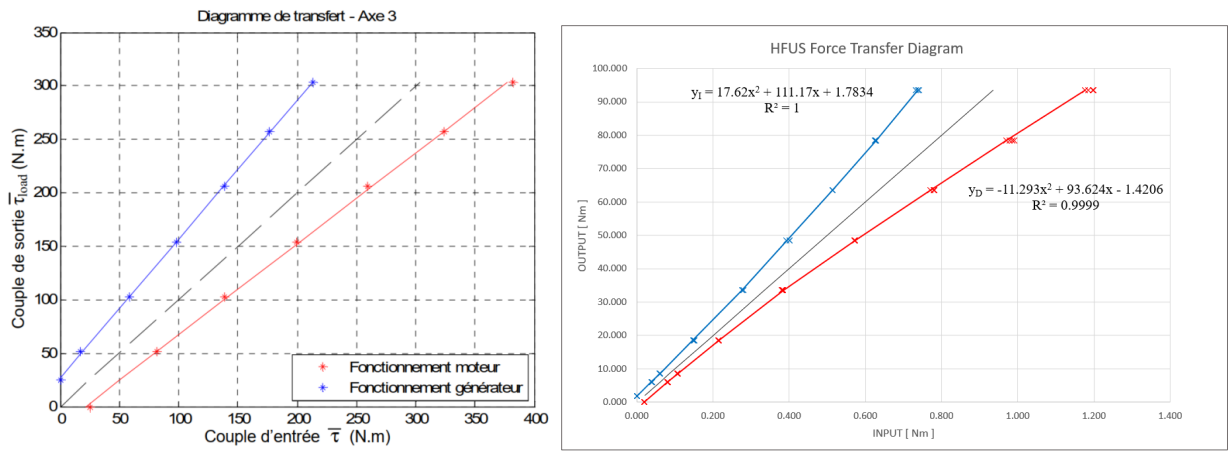


Figure 2.15 Left: Force Transfer Diagram of Axe 3 of Stäubli RX130L (Hamon, P., 2011); right: HFUS Force Transfer Diagram

Further work would be necessary to complete the friction torque model by studying the effect of speed and its eventual coupling. This test bench used in Setup A could be modified to allow studying the effect of speed independently to the applied torque.

Chapter 3

3. Offline approach towards obstacle avoidance in collaborative robots

This chapter deals directly with the topic of obstacle avoidance in collaborative robots. The topics discussed in Chapter 2 and Chapter 3 do not have direct correlation. The main reason for the difference in topics comes from the lack of accessibility/availability of equipment in the foreseen facilities and time constraints. Repercussions caused due to the COVID-19 pandemic and security issues.

3.1. Introduction

The combination of the human capabilities and dexterity in conjunction with the machine efficiency has been a dream for the past several decades. This Human-Robot Collaboration is the reason why many researchers try to overcome the adversities that this dream represents. Conventional automation technology has arrived at a bottle-neck, where this collaboration is the potential solution for the industrial dilemma.

We must as well differentiate the more general concept of Human-robot Interaction and Human-Robot Collaboration, as the first actually refers to all the possible passive and collaboration interactions, and the second one specifically refers to the conjunction work of human-robot to achieve a common goal (Tan et al., 2009).

One of the first propositions of a robot-human collaboration was proposed in 1994, by Kosuge and his team (Kosuge et al., 1994), visualizing an scenario where humans and robots worked together in coordination to accomplish a given task. They consider this as the next logical step that would improve the industry, by making the robots more human-friendly and expand their adoption to execute more complex tasks.

Since then, this topic has grown interest and development. In 1996 there was another study done by Edward Colgate (Colgate et al., 1996), where he introduced the term of cobot (suggested by Brent Gillespie): a robot that manipulates objects, providing guidance to the operator's motion to fulfill a determined task.

Initially, talking about industrial robotics in general, the aim was to improve the robots so they could execute more complex tasks with less human intervention. On that note, researchers oriented their studies towards the path-planning problem and its importance for collision avoidance: the machines were expected to move from one location to another without collision. However, this is not an easy task. The level of difficulty changes, depending on different factors such as the environment, the type of robot, the task to execute, in order for the robot to move safely around the obstacles (Gill & Zomaya, 1998). This planning is performed in two different cases: i) off-line planning, where a complete description of the workspace is known in advance (this is a global approach, finding the solutions before implementing them); and ii) on-line planning, where the robot makes decisions given the current state of the environment and moves toward the target, however, the path found may not correspond to the most efficient or optimal one.

The problematic addressed here is to find suitable trajectories for obstacle avoidance in collaborative robots, finding those with better performances in terms of overall speed along the trajectory and total time for them to accomplish the task.

Figure 3.1 UR10e collaborative robot by Universal Robots.

3.2. Obstacle avoidance in the literature

Human decision-making capabilities combined with the robots' precision and efficiency, may indeed enable a remarkable level of performance, however, this collaboration must take into consideration the security for human beings (Krüger et al., 2009; Tan et al., 2009).

Robots have been widely used in industrial manufacturing for several decades, preventing human exposure in hazardous situations or heavy labor tasks, resulting in increased efficiency and productivity. Traditional rigid manufacturing techniques, on the other hand, have proven unable to match current industrial demands, such as small batch production and customization. As a result, industrial manufacturing is shifting toward more flexible and intelligent processes: the Industry 4.0. Coexistence and collaboration between humans and robots are major attributes of the new generation of robots. They blend human intelligence in thinking, learning, and decision-making with robot precision, speed, and repeatability to quickly adapt to new work demands and unforeseen situations. Scholars from across the world are interested in human-robot collaboration.

In the past decade, collaborative robots are used for many industrial automation applications. Because of that, the standard ISO/TS15066:2016 was submitted into revision and confirmed its validity in 2019 (ISO/TS, 2016): it announces the specifications for the safety requirements of collaborative robots and the workspace environment. These revisions are done because the coexistence of human operator and robots in the same workspace is a matter that we must not neglect, but to take into consideration the potential dangers and conceive reactive safety control designs.

In human-robot collaboration, the physical gap between human and robots is eliminated, and human security protection is a major challenge in human-robot interaction. Other safety procedures should be considered, developed, and implemented to ensure the safe interaction between humans and robots (J. Chen et al., 2019).

Different collision avoidance strategies are proposed in previous decades. In (Ebert et al., 2005), an emergency stop is activated when an operator entered a hazard zone. However, productivity can be severely reduced due to potential frequent stops.

In (Schmidt & Wang, 2014), four control solutions are proposed for collision avoidance within HRI: i) speed reduction and warning, ii) calling a stop interruption, iii) moving away from the oncoming operator, and iiii) changing the path to the target at run-time.

Human-robot interaction (HRI) is a promising field of robotics research, as more and more interest and development has been devoted to this area. Physical human-robot interaction

(pHRI) is one aspect from HRI that involves the collision detection, on-line reaction and a continuous physical interaction (Alami et al., 2006; De Luca & Flacco, 2012; De Santis et al., 2008; Popov et al., 2017).

The other aspect of HRI deals with the active collision avoidance, where the predefined trajectory by the robot adapts to the changes in the workspace to avoid contact with objects or humans, which is highly desirable to guarantee human safety. To achieve this different obstacle detection methods are proposed, depending on the requirements and sensors available. Some may use depth cameras to detect the obstacle (Nascimento et al., 2021) and react accordingly or even wearable sensors to locate the presence of the human-operators with respect to the robot (Safeea et al., 2019).

In the case of manipulators and cobots, the collision-free motion planning must take into consideration the environmental constrain as the robot moves, to prevent robot links and joints from collision with these constraints or obstacles. The end-effector of the robot must preserve its trajectory within the task space while it avoids collision with the obstacles. The robot then restores its original posture and complete the preestablished trajectory and task. Chen in (J.-H. Chen & Song, 2018) considered the design of multiple control points with given spheres, distributed along the links and joints of the robot, to generate virtual force constraints on these points

Another collision-free trajectory method is proposed in (Meziane et al., 2017) applying a neural network to generate waypoints required to avoid obstacles in dynamic environments, in the context of Human-robot collaboration.

On the same note of collision-free trajectories, in (Rubio et al., 2016) proposed an algorithm that considers limits of jerk, power and torque, for each joint of a robotic manipulator. Validating their results performing simulations with a PUMA 560 robot.

The purpose of path planning is to give high precision robot motion. Some of the most popular methods for planning a collision-free path are the Rapidly-exploring Random Tree (RRT) (LaValle & Kuffner, 2001) and the Probabilistic Roadmap (Kavraki et al., 1996). They are often fast, but its efficiency decreases when they are facing unknown environments. According to Zhao in their mobile robot application (R. Zhao & Sidobre, 2015), one possible solution is to

include a layer of trajectory computation between the path planner on the low level controller, meaning that it would use preestablished paths and replan trajectories locally. Computer power must be should be sufficient, as the optimal trajectory generation must be done in real-time (Chu et al., 2018).

Trajectory time-scaling methods are proposed in (Dahl & Nielsen, 1990) and (Moreno-Valenzuela, 2006) to execute fast trajectories along a geometric path. These methods are usually used to adjust the torque or speed while the tracking path is maintained.

In (Broquère, 2011) a method is introduced that considers constraints of Human-robot interaction during the execution of movements by adapting the law of motion of the trajectories.

The cobot obstacle avoidance path planning refers to plan a feasible path of the manipulator end, from one given starting point towards a target pose. The robot is required to avoid collision with any obstacle during this operation. This problem becomes more complex when it is required 3D obstacle avoidance path planning.

Artificial Potential Fields method is proposed in 1985 by the pioneering work of Khatib (Khatib, 1985). He aimed to use low level control during on-line operations, to approach the real-time obstacle avoidance problem. This method consists in the creation and applications of virtual forces of repulsion (away from obstacles) and attraction (towards the final goal configuration), modifying then the original trajectory of the manipulator's end-effector.

The usage of artificial potential fields method is widely used as a mean for obstacle avoidance since its introduction to robotics by Khatib. However, these methods of repulsion/attraction vectors may suffer from oscillation or local minima problems (Safeea et al., 2020). For this reason, some other researchers have proposed Probabilistic Roadmaps or circulatory fields.

In Probabilistic Roadmaps, different roadmaps of the scenes are generated (learning phase), and then a feasible path is searched (query phase) (Kavraki et al., 1996). However, postprocessing before implementation is needed to smooth the trajectories. Similarly, in (Leven & Hutchinson, 2002) the Dynamic Roadmaps Method is proposed, where the roadmap generation

can be updated according to the changes in the surroundings, increasing the online efficiency of the classical Probabilistic Roadmap approaches, further developed in (Kunz et al., 2010).

Regarding the circulatory fields method, this is inspired by electromagnetism phenomena, as proposed in (Singh et al., 1996), where fictitious current elements circulate around the obstacles, without generating unwanted local minima.

One of the methods for collision-free trajectories in robot manipulators consists in the use of the configuration space, which is the transformation from Cartesian space. The robot is mapped to a point (commonly the tip) and everything in the workspace goes as well through this mapping process. The robot is then guided towards its goal, evading potential collisions, by using different methods such as roadmaps or artificial potential fields (Latombe, 1991).

In (Zhang & Sobh, 2003) the problem is formulated in terms of collision avoidance of the manipulator's links instead of points. The method consisted in continuously controlling the link closest point to the obstacle.

In 1981 Lozano-Peres addressed the "Pick and Place" problematic problem for manipulators in presence of obstacles. He proposed a method to avoid these obstacles taking into consideration the workspace and the selection of safe grasp points on objects (Lozano-Perez, 1981).

(Scoccia et al. 2021) proposed an obstacle avoidance strategy for redundant manipulators. This type of robots possesses more degrees of freedom than the needed to execute a task. The authors combined off-line path planning with on-line motion control by using the potential field method and smoothing the process by the use of interpolations with Bezier curves, validating their proposal with simulations in 2D (planar robot). Similarly, Bezier curves were used in (Chen et al. 2019b; Kawabata et al. 2015) for obstacle avoidance in their path planning, to avoid sharp edges and high accelerations in their trajectories.

In mobile robotics and UAV applications Dubins curves (Dubins, 1957) have been used to resolve the problematic of obstacle avoidance (Lin & Saripalli, 2014; Reeds & Shepp, 1990; D. Yang et al., 2013). Additionally to the Dubins curves, in (Bayar, 2017) it is proposed its use in conjunction with the Potential Field Method for the computation of collision-free paths.

Based on the foregoing analysis, it surely exhibits good qualities of path planning. We may conclude that adequate planning and control are beneficial in the field of robotics, for a successful collision avoidance to improve the efficiency of human-robot collaboration.

3.3. General background theory

3.3.1. Path planning and Trajectory planning

In robotics, path planning is a subproblem of the general motion planning problem that consists in finding collision-free paths between an initial and final configuration. While trajectory planning is an additional step where the series of robot configurations are specified as a function of time. The trajectory planning should as well take into consideration constraints such as limits on joint velocities, accelerations or torques (Lynch & Park, 2017).

In this chapter, we generate the path planning approach towards obstacle avoidance of the end-effector parting from the Cartesian space; from there we compute the joint space trajectory of the manipulator within the simulation environment, given all the constraints proposed in our experiments, the system restrictions and workspace capabilities.

3.3.2. Global planning towards Local planning

Global planning consists in the off-line generation of possible trajectories that the robot could execute, given all the constraints or obstacles present in the environment (like the Roadmap Tree methods). Advantages: low-online computation costs and better performances in terms of precision and efficiency.

Local planning, in the other hand, consists of the computation or correction of the trajectory in on-line fashion. This performance is therefore better than the global planning in case of dynamic changes in the environment, however they may fall into extreme local minima problems or undesired performances.

Our proposition is to establish the first step towards a hierarchically planning, consisting of the generation of multiple off-line trajectories, in the presence a single static obstacle, to later shift between trajectories in case of obstacle configurations updates (refining the path locally) (K. Yang et al., 2014). This chapter deals only with off-line trajectories in presence of a single static obstacle.

3.3.3. Workspace, operational space and joint space

Workspace is the specification of configurations that the end-effector of the manipulator can reach. The definition is determined by the structural constraints of the robot, and it is described as a subset of the task space that can be reached by the end-effector. The motion control task can be expressed in operational space or joint space (Lynch & Park, 2017):

Operational space, also known as task space or Cartesian space, is the space in which one can naturally express the robot's task. It is defined by the position and orientation of the end-effector of the manipulator. The trajectory controller is the fed by a steady stream of end-effector configurations $X_d(t)$

Joint space is the space representation in joint angles of the robotic manipulator. The trajectory controller is fed with a steady stream of desired joint positions $\theta_d(t)$.

3.3.4. Path continuity

Continuity notion is fundamental in the definition of a path. The continuity problem refers to the geometric continuity of a path in terms of curvature continuity and tangential continuity. There are two types of continuities (Ravankar et al., 2018): 1) Geometric continuity and 2) Parametric continuity.

Geometric continuity (G^i) guarantees that the endpoints of n segments of the path meet, and tangent vector's directions are equal.

Parametric continuity (C^i) guarantees that the endpoints of n segments of the path meet, and that both, the tangent vector's direction and magnitudes are equal.

As cited by Ravankar (Ravankar et al., 2018), parametric continuity mean smoothness both of the curve and of its parameterization. Geometric continuity simply means the smoothness of the track that the robot traverses. For example, C^1 continuity means continuity of the tangent vector, while G^1 continuity means continuity of slope; C^2 continuity means continuity of the acceleration vector, while G^2 continuity means continuity of the curvature. In terms of robot motion, the C^1 continuous motion preserves velocity, whereas C^2 continuous path preserves acceleration. For robot path planning, what matters is mainly the path's C^1 or C^2 continuity (see Figure 3.2).

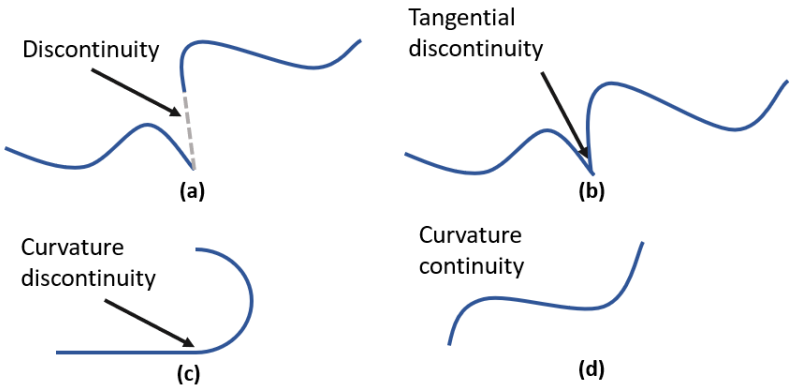


Figure 3.2 Parametric continuity paths. (a) Discontinuous segments. (b) C^0 continuity. (c) C^1 continuity. (d) C^2 continuity.

3.3.5. Dubins curves

Dubins curves (Dubins, 1957) were originally proposed in 1957 to describe a continuously differentiable path (C^1continuity type). The initial interest was to obtain a path of minimal length for a given particle with maximal curvature allowed by the system. Since then, these types of curves have been used in applications in mobile robotics and UAV's for obstacle avoidance purposes (Lin & Saripalli, 2014; Reeds & Shepp, 1990; D. Yang et al., 2013).

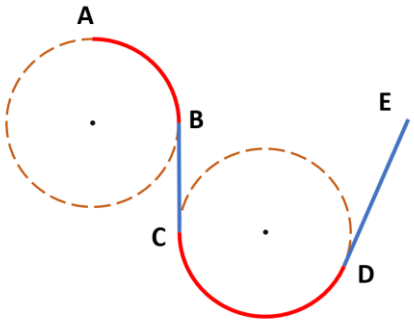


Figure 3.3 Sample path using principle of Dubins curves. Straight segments $\overline{(BC)}$ and $\overline{(DE)}$ are combined with circular arcs $\widehat{(AB)}$ and $\widehat{(CD)}$

Figure 3.3 depicts a sample smoothing path using Dubins curves, which is composed by the combination of circular arcs and straight segments. Points B, C and D are points that represent C^1 continuous motion, by having the same tangent vector of motion (magnitude and direction).

3.4. Arc path planning for obstacle avoidance

In this section, we propose a path plan that consists of a series of arcs in the presence of an obstacle. The problem consists then to obtain a smooth path, having the constraints that the different segment meeting points must represent a C^1 continuity (equal tangent direction and magnitude). The approach in here proposed consists in a variation of the Dubins curve. The difference consists in that we are not restricted of having the same curvature in all curved segments: Dubins curves applications mainly consist of the computation of the smallest path length, constrained by the maximum curvature possible by the mobile system (which can be a negligible restriction for manipulators robots).

We present our obstacle modeling for our study, and we gradually present in the following subsections the different steps that lead from the simplest 2D approach with a semicircle for obstacle avoidance towards the 3D general approach for obstacle avoidance given a predefined linear path in the operational space of a manipulator's end-effector.

3.4.1. Obstacle modeling

In order to ensure that there is not collision, we must delimitate our obstacle model. To simplify the calculation as much as possible, we will consider a single particle obstacle as first approach. In later iterations, we can consider the model of circumscribed ball method (wrapping the obstacle with the smallest cube) (M. Zhao & Lv, 2020).

3.4.2. 2D linear path case

The importance of a good path planning relays on the selection of the best way to reach from point A to point B given some performance indicators, such as smallest distance to travel or lowest working cost.

As a first approach for obstacle avoidance, we consider a linear path in the XZ plane, going from an initial 3D position A to a $3D$ target position B, where a particle obstacle O is located in-between these two positions (see Figure 3.4).

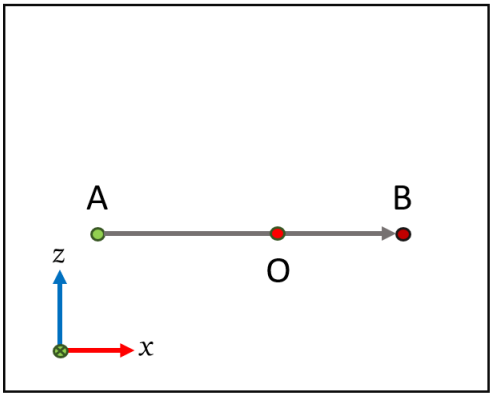


Figure 3.4 Path from A to B with an obstacle in O

We also consider that the avoiding path should be done in the superior part of the plane ABY (Y the axis direction that goes through the screen). This to delimitate the workspace, by suggesting that below the obstacle no valid path could be performed (working station).

3.4.2.1. Semicircular path

The first smooth idea that we could think of is where the new avoidance trajectory is described by a simple semicircular path with radius R, see Figure 3.5.

This new path would fulfill the requirement towards obstacle avoidance, however, the starting and ending of the diverged section are quite abrupt. For the avoidance to be done under the best conditions, the resulting trajectories should be performed without drastic changes in the direction of velocities and accelerations: we need smoother trajectories.

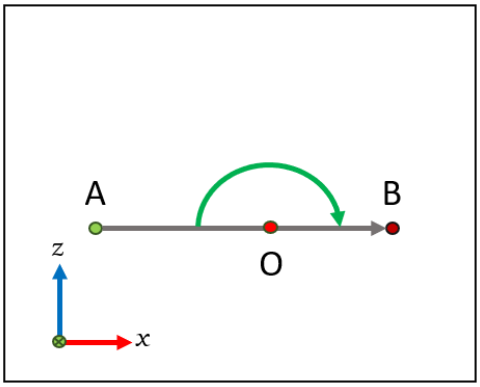


Figure 3.5. Semicircular avoiding path.

3.4.2.2. Arc path

The following idea would be to follow an arc path instead of a whole semicircular path, just like it is depicted in Figure 3.6

However, even though the resulting trajectory would be smoother than the semicircular one, the changes of resulting velocities and accelerations would be a cause of low performances.

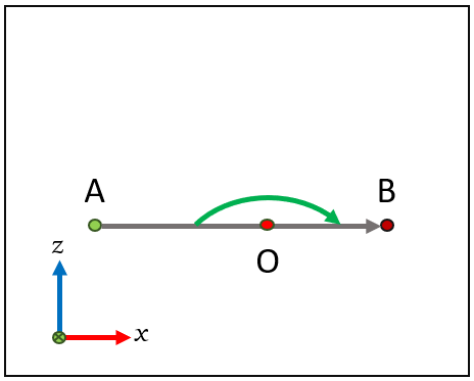


Figure 3.6. Arc avoiding path.

3.4.2.3. Consecutive arc paths

The resulting next best idea would be to concatenate a series of three arcs, resulting in a path like is depicted in Figure 3.7

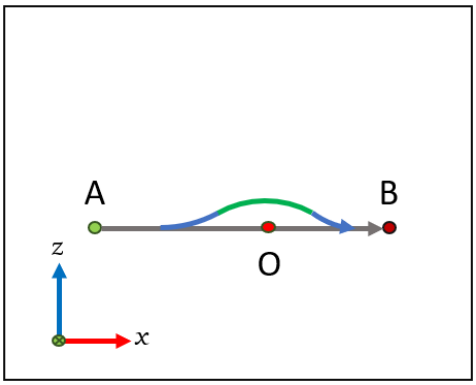


Figure 3.7. Consecutive arc paths.

Several equations can be computed to obtain a chain of arcs as depicted in Fig. 7.

Considering a plane XZ with origin in O as in Figure 3.8, we can compute these equations given four main parameters: h, r, R2, R3.

h: represents the sagitta of the main arc at the center with limits [0, r]

r: represents the minimal distance from O when the avoidance starts or ends

R_2: represents the radius of arc_2, first arc most proximate to point A

R_3: represents the radius of arc_3, first arc most proximate to point B

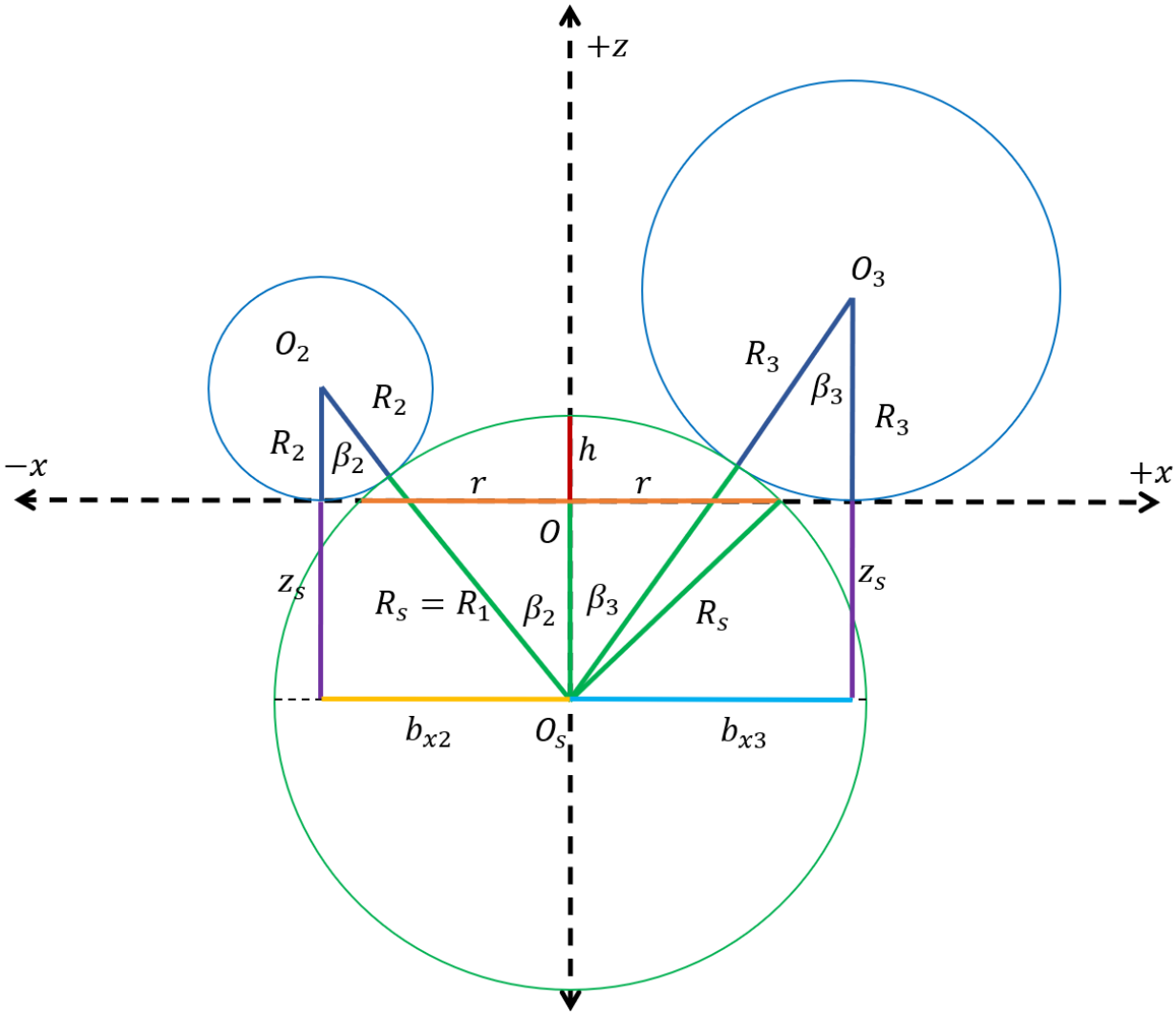


Figure 3.8. Diagram used to obtain the arcs equations.

From here, we can compute the equations shown in Table 3.1.

Table 3.1 Equations used to compute the arcs path

Arc 2	Arc 1 (green)	Arc3
$b_{x2} = \sqrt{(R_2 + R_s)^2 - (R_2 + z_s)^2}$	$R_s = \frac{1}{2h}(h^2 + r^2)$	$b_{x3} = \sqrt{(R_3 + R_s)^2 - (R_3 + z_s)^2}$
$\beta_2 = \cos^{-}(\frac{R_2 + z_s}{R_2 + R_s})$	-	$\beta_3 = \cos^{-}(\frac{R_3 + z_s}{R_3 + R_s})$
$arc_2: [-\frac{\pi}{2}, -\frac{\pi}{2} + \beta_2]$	$arc_1: [\frac{\pi}{2} + \beta_2, \frac{\pi}{2} - \beta_3]$	$arc_3: [-\frac{\pi}{2} - \beta_3, -\frac{\pi}{2}]$

Matlab implementation results is shown in Figure 3.9, with the set of parameters defined as:

$$h = 0.5r,\ r = 0.2,\ R_2 = 0.5r \text{ and } R_3 = r.$$

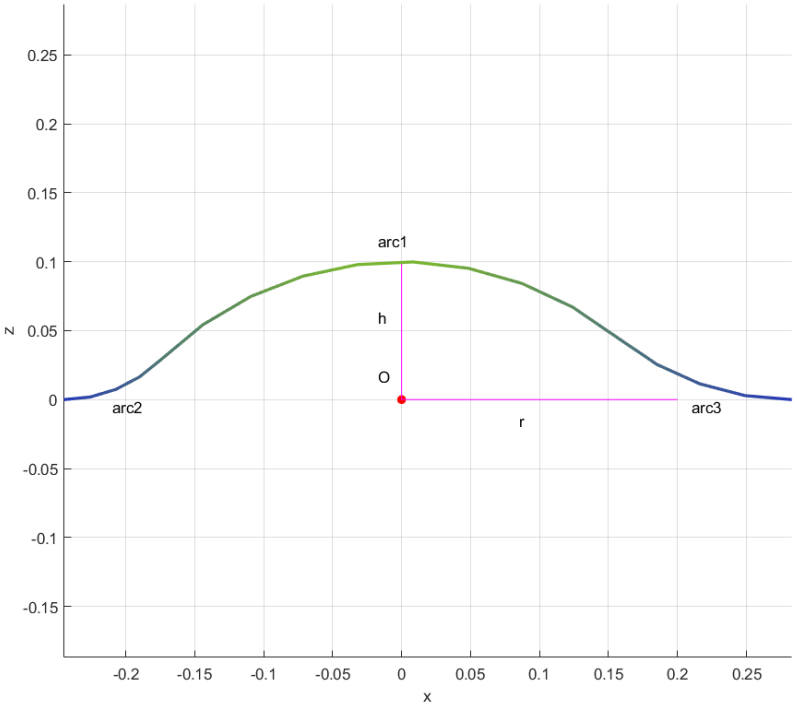


Figure 3.9. MATLAB implementation of the arcs computation

The following step is to concatenate the original initial and final positions of the path with the corresponding avoidance arcs sequence and the corresponding translation of the obstacle

and the computed arcs, to the original 2D cartesian position. In this way the depicted avoidance strategy is completed as shown in Figure 3.7.

$$A + avoidance\ arc\ sequence + B$$

3.4.3. 3D linear path case

In order to generalize the previous 2D collision avoidance path into a 3D path case, we now consider a linear path in the 3D space that goes from point A to point B. In Figure 3.10, it is shown in the black dotted line the original lineal path from A to B.

To pass from the 2D case to a 3D case, we take a finite n series of waypoints from the 2D case shown Figure 3.9 (projected in Figure 3.10 in the xz-plane), and then each one of the points is rotated towards the new coordinate system orientation, and then these points are translated to the 3D case scenario.

A series of equations and functions were developed in MATLAB, by computing quaternions, cross products, dot products, etc.

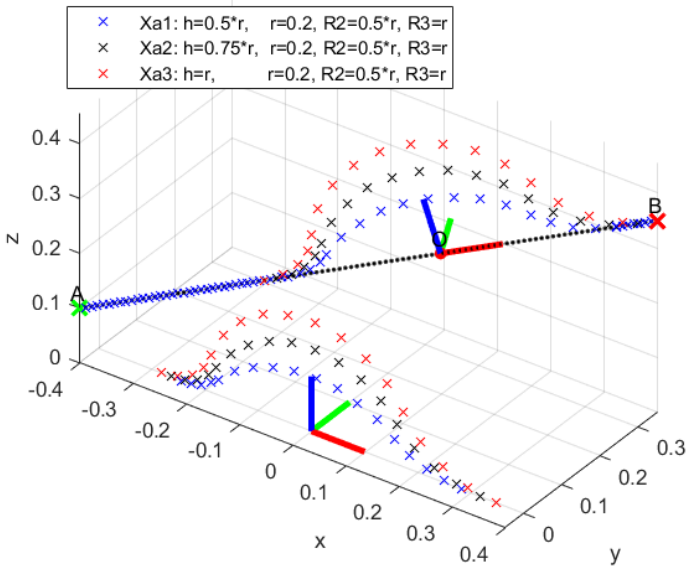


Figure 3.10. Several arcs paths changing the parameter h.

Path planning for obstacle avoidance is just one aspect of the manipulator control. The next step is to verify the kinematic feasibility of this path in the manipulator, in terms of workspace capabilities. Once a suitable path is confirmed for execution, a trajectory in terms of the joint variables is to be calculated. Then the control signals for the actuators controlling the manipulator must be generated, based on the trajectory and the feedback of the robot's status (Gill & Zomaya, 1998).

3.5. Cartesian trajectory generation

To assess the validity of the previous collision-free paths in collaborative robots, we use the UR10e robot description, constraining the workspace of the given operational space such that:

- The manipulator's end-effector should maintain the same orientation along the lineal path and the collision-free path.
- The end-effector lowest coordinate in $\boldsymbol{z}$ is limited by the original lineal path. This to prevent the manipulator to compute trajectories below the original lineal path. The avoidance occurs only above or alongside the $\boldsymbol{z}$ coordinate of the original path.
- Due to the complex structure of the manipulator, it should also be considered in the workspace to avoid collision with itself during operation.

All the above considerations are not an easy task to circumscribe the trajectory conditions for our approach. For that reason, and to ease the reproducibility and access of the computations here proposed, we use a set of open-source tools available in the robotics community.

3.5.1. Tools and libraries for trajectories computation and simulations

3.5.1.1. ROS, MoveIt! and Rviz

ROS is an open source robot operating system (Quigley et al., 2009). It provides a structured communication layer that allows to send and to receive messages above the host operating systems. Many researchers all around the world have used this and it is up today one of the most popular frameworks in robotics.

Some advantages of ROS (Niryo, 2018):

- ROS is modular. There are packages for everything: to compute trajectories for example.
- Python or C++. We can choose the language of our preference. ROS has mainly targeted the use of these two, but the possibilities expand these using certain libraries.
- ROS is open source. Most of the packages available are free to access and its BSD license allows to modify and to use them for commercial purposes.
- Simulation tools. Rviz and MoveIt! are some example tools that allows to perform simulations and advance in projects without using the real robot.

MoveIt! is a motion planning framework that works alongside ROS (Coleman et al., 2014). It has been integrated successfully with many robots, including the UR family robots. It uses OMPL motion planning plugins, OROCOS Kinematics and Dynamics Library (KDL) for inverse and forward kinematics, Fast Collision Library (FCL) for collision detection (see Figure 3.11).

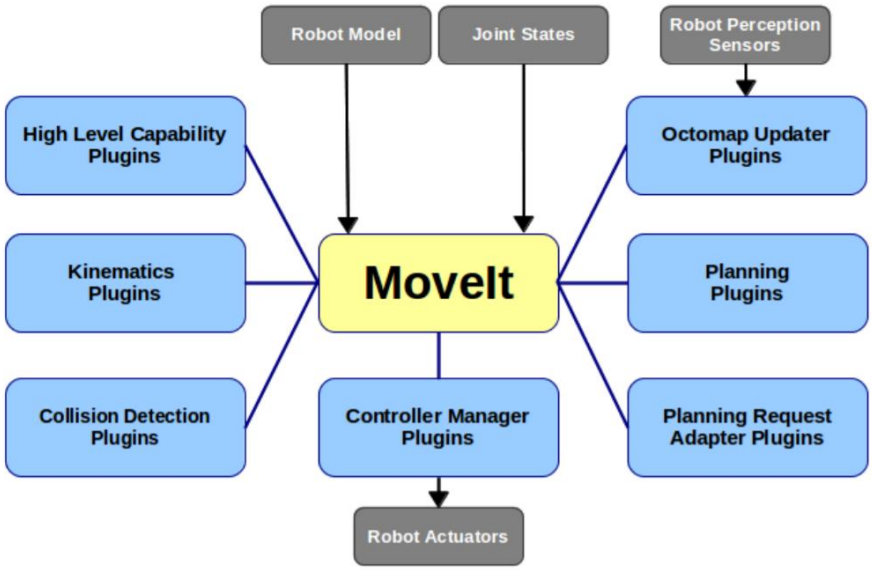


Figure 3.11 MoveIt plugin-centric framework (Coleman et al., 2014)

Rviz is a three-dimensional visualizer integrated in the core of ROS, used to visualize robots, the environment and sensor data. It can be used for simulations and real-time visualization when on-line with the robot. It is configurable and may integrate plugins: MoveIt! for example.

3.5.1.2. Advantages and disadvantages found with MoveIt! approach

Advantages:

- **Motion planning generation.** Once everything is configured for a given system. Motion planning can be done straightforward.
- **Robot model format.** Robot representation by using Unified Robotic Description Format (URDF) to describe the three dimensional geometric properties of the robot, its kinematics and some other properties: joint limits, sensors, dynamic properties, visualization meshes, etc.
- **Setup Assistant.** To ease the setup of the workspace and initial configuration requirements: collision matrix, planning group, robot poses, virtual joints, end effector semantics.
- **Self-Collision Matrix.** First step of the Setup Assistant. It allows the generation of self-collision matrix, used for future planning, and improve the collision detection.
- **GUI for plan visualization.** By using Rviz or Gazebo, we can simulate trajectories and tasks, on-line visualization, and interaction with the robot.

Disadvantages:

- Generated trajectories can only be generated given the paths or waypoints in Cartesian space or joint space. Time, speed, or accelerations can not be directly constrained or parametrized for the trajectories generated (MoveIt, 2018).
- Final time variations of a given trajectory from point A to point B can only be achieved through path planning itself. Ideally, MoveIt generates trajectories with the maximum speed and acceleration for each joint, delimited in the configuration files of the robot.

3.5.1.3. Universal Robots ROS Driver and UR10e collaborative robot

Universal Robots ROS Driver is provided as open-source and it provides a stable interface between the UR robots and ROS (fmauch, 2019/2019). It uses the Real-Time Data Exchange (RTDE) and it allows to extract the factory calibration information of the robot inside

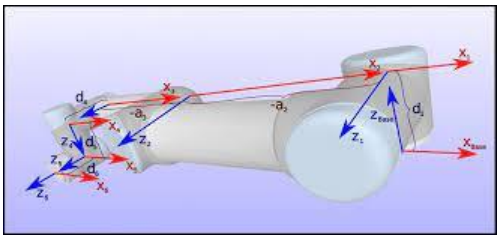
ROS. It provides all the necessary configuration files for a transparent interaction and to integrate the use of the MoveIt! framework.

3.5.1.3.1 UR10e collaborative robot

UR10e is a versatile collaborative robot with a payload lift of 12.5 kg and reach of 1300mm (UR, n.d.-b). It is suited for numerous applications in the era of Industry 4.0.

Figure 3.12 shows the Denavit-Hartenberg parameters diagram description. In Table 3.2 are listed the values for each one of the parameters for each corresponding joint. This description is included as well in the Universal Robots ROS Driver, in the URDF files.

(a)



(b)

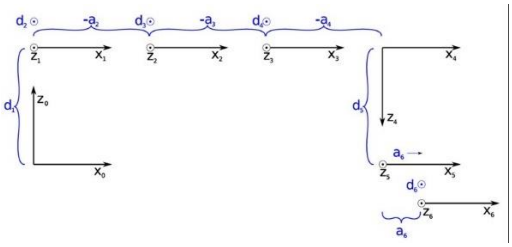


Figure 3.12 Denavit-Hartenberg diagram description

Table 3.2 UR10e Denavit-Hartenberg parameters

UR10e							
Kinematics	theta [rad]	a [m]	d [m]	alpha [rad]	**Dynamics**	Mass [kg]	Center of Mass [m]
Joint 1	0	0	0.1807	π/2	Link 1	7.369	[0.021, 0.000, 0.027]
Joint 2	0	-0.6127	0	0	Link 2	13.051	[0.38, 0.000, 0.158]
Joint 3	0	-0.57155	0	0	Link 3	3.989	[0.24, 0.000, 0.068]
Joint 4	0	0	0.17415	π/2	Link 4	2.1	[0.000, 0.007, 0.018]
Joint 5	0	0	0.11985	-π/2	Link 5	1.98	[0.000, 0.007, 0.018]
Joint 6	0	0	0.11655	0	Link 6	0.615	[0, 0, -0.026]

3.5.2. Trajectory generation architecture with ROS-MoveIt!

ROS modular elements are called nodes, they are programming units that perform computation. Nodes communicate between each other passing messages through topics or

services. The topics are channels with publish/subscription semantics: a node sends messages when publishing to a topic and a node receives messages when subscribing to topic (and the information is being published in said topic). With services, in a similar manner, a node can request certain information to another (client/service semantics)(ROS Wiki, 2014).

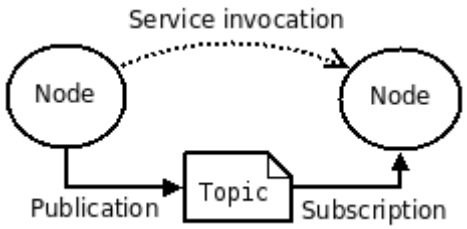


Figure 3.13 Basic ROS node communication layout (ROS Wiki, 2014)

The main node of MoveIt is called *move_group*. This node uses all the individual elements together to provide the interface of a set of ROS services and actions. This node can be configured through the ROS Param Server, getting as well the URDF and SRDF configuration files that describe the environment and the robot itself.

Figure 3.14 depicts the basic architecture and interaction between the principal nodes when using MoveIt with ROS, in addition to our Obstacle Avoidance Planner.

Essentially, the *move_group* node interacts with the robot to get current state information (joint positions, etc.) and to talk to the robot controllers through the *JointTrajectoryAction* interface.

For this communication to happen, first the path must be predefined by the user, in this case the initial step is through our Obstacle Avoidance Planner. Then the waypoints information of our precomputed path is given to the *move_group_interface* to validate the proposed path. This is achieved through the execution of several services (Inverse Kinematics, Planning Scene, Plan Validity, etc.) to finally determine the corresponding trajectory, considering all the environment and robot constraints present in the configuration files.

Just then, the *move_group* can authorize or not the execution of a trajectory. In the real robot is through the *JointTrajectoryAction* services, taking the state information of the robot and communicating through the ROS Robot Controllers; in simulation environment this can be

performed through the GUI – Rviz interface, by using *fake* controllers, *fake* robot actuators and state positions.

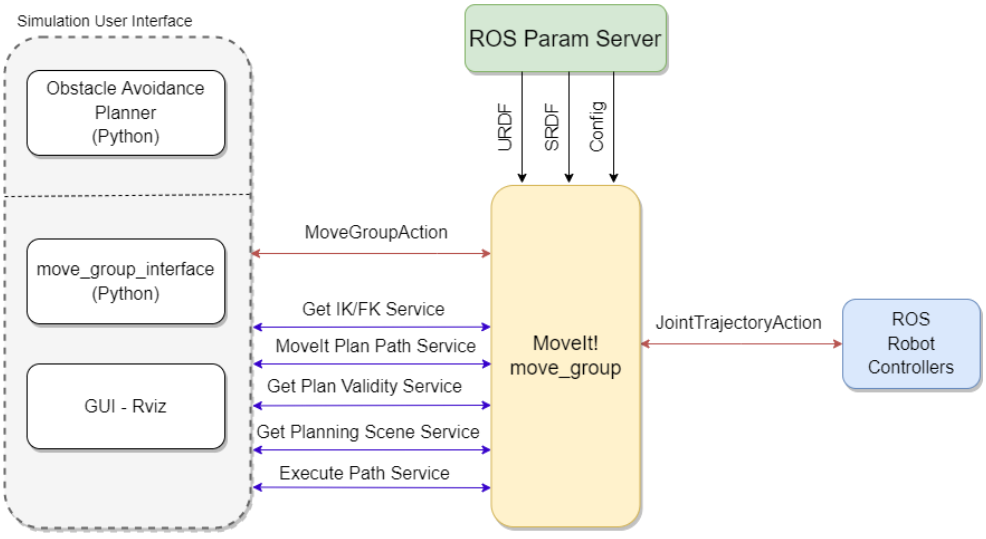


Figure 3.14 Basic architecture with ROS-MoveIt! + Obstacle Avoidance Planner

3.6. Simulations and results

We initially propose a linear path from pose $\boldsymbol{A}$ to pose $\boldsymbol{B}$. We constrained the end-effector to maintain the same orientation along the different trajectories. The obstacle avoidance motion is to be done above the original linear trajectory ($\boldsymbol{z}$-axis). The proposed poses for validation are the following (position in meters and roll-pitch-yaw in radians):

$$A = \left[0.4, \quad -0.4, \quad 0.25, \quad 0, \quad -\pi, \quad \frac{\pi}{2}\right]^T$$

$$B = \left[0.5, \quad +0.4, \quad 0.46, \quad 0, \quad -\pi, \quad \frac{\pi}{2}\right]^T$$

Our main interest is to reduce the final time (total task duration) of the trajectory, while safely avoiding any given particle obstacle. When a particle obstacle is set in the A-B linear path, three different additional trajectories are generated and simulated to compare the final times results and their overall performances.

For the path planner algorithm, the $\boldsymbol{y}$ coordinate of the obstacle is introduced as input, and for the simulations this $\boldsymbol{O_y}$ is set at $-0.1m$.

We vary the sagitta $\boldsymbol{h}$ by a factor of $\boldsymbol{r}$, to obtain three different obstacle avoidance trajectories:

$r = 0.20\, m$

$h = \{1.00r, 0.75r, 0.50r\}\, m$

The resulting main arc of the avoidance trajectory with $\boldsymbol{h} = 1.00r$ has a semi-circular basis of radius $\boldsymbol{r}$. The followings trajectories main arcs with $\boldsymbol{h} = \{0.75r, 0.50r\}$ have larger radius, but they travel closer to the original linear trajectory (or obstacle, for that matter).

Figure 3.15 shows the resulting trail trajectories simulations with the UR10e robot, describing 4 different cases:

(a) Linear trajectory from $\boldsymbol{A}$ to $\boldsymbol{B}$

(b) Obstacle avoidance trajectory with $r = 0.20\, m$, $h = 1.00r$ and $\boldsymbol{O_y} = -0.1\, m$.

(c) Obstacle avoidance trajectory with $r = 0.20\, m$, $h = 0.75r$ and $\boldsymbol{O_y} = -0.1\, m$.

(d) Obstacle avoidance trajectory with $r = 0.20\, m$, $h = 0.50r$ and $\boldsymbol{O_y} = -0.1\, m$.

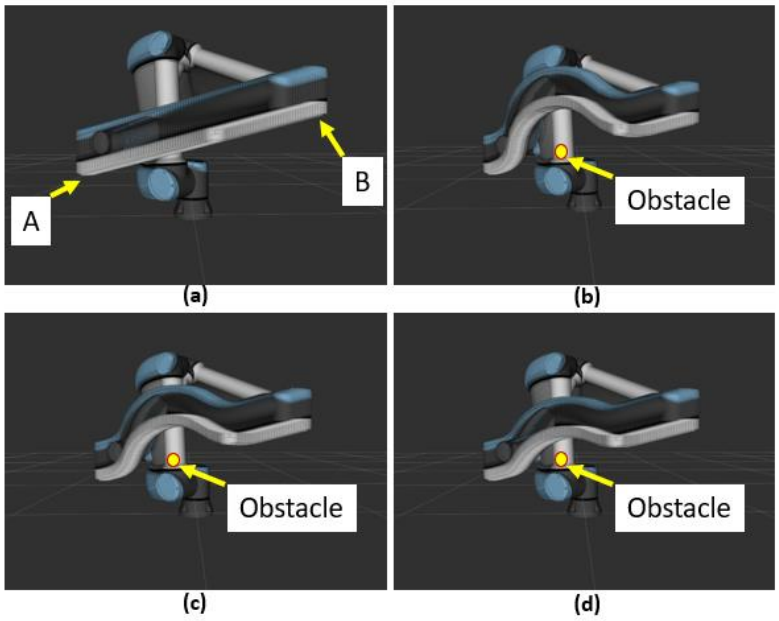


Figure 3.15 UR10e trail trajectories: (a) linear trajectory; obstacle avoidance trajectories: (b) with h = 1.00r, (c) with h=0.75r, (d) with h=0.50r, with r=0.20m and Oy=-0.1m

Figure 3.16 shows the resulting position trajectories in the Cartesian space for each one of the cases previously presented: linear case in black (a), avoidance trajectory cases (b) in red, (c) in magenta and (d) in blue.

Similarly, Table 3.3 presents the total distance traveled by the end-effector, the total time taken to complete the trajectory, and its average speed for each one of the four trajectory cases. Comparing the average speed among the trajectories gives a first insight of the overall performance of the different trajectories. We can conclude that the case (d) has the best performance between the obstacle avoidance trajectories, in terms of time and average speed; however, case (d) decreased 44.7% (0.1577 m/s) in speed with respect from the original linear trajectory (a) (0.2851 m/s). In terms of time, case (d) took 91.33% (5.5901 s) longer to finish with respect to the case (a) (2.9216 s).

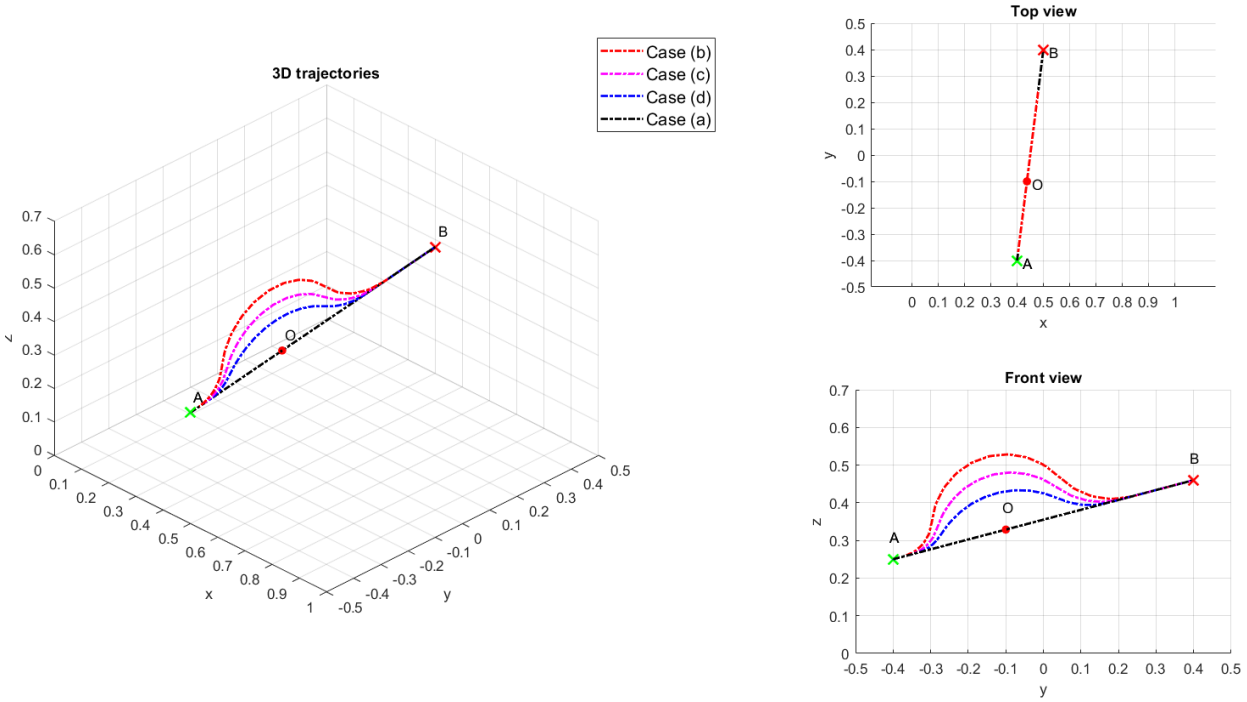


Figure 3.16 Resulting end-effector positions in the Cartesian space

Table 3.3 Total distance traveled, total time and mean speed for each case trajectory

Trajectory	Total distance (m)	Total time (s)	Average speed (m/s)
Case (a) Linear	**0.8331**	**2.9216**	**0.2851**
Case (b) with $r = 0.20m$, $h = 1.00r$, $O_y = -0.1m$	**0.9888**	**6.8056**	**0.1453**
Case (c) with $r = 0.20m$, $h = 0.75r$, $O_y = -0.1m$	**0.9300**	**6.2226**	**0.1495**
Case (d) with $r = 0.20m$, $h = 0.50r$, $O_y = -0.1m$	**0.8816**	**5.5901**	**0.1577**

If we look the resulting joint positions and joint velocities for each one of the cases, we can derive more information of their overall performance during motion.

Figure 3.17 depicts the case (a) linear trajectory, with its respective joint positions and joint velocities. This is the ideal case without obstacles, and we can verify that their shapes are smooth and without any position or velocity drastic changes.

Figure 3.18 depicts the obstacle avoidance motion of case (b), joint positions and joint velocities are as well shown. At first glance, positions are smooth in time, however, if we look at the joint velocities, we can perceive that there are oscillations in almost every joint, except for joint 5, (joint 1 and 6 have almost identical motion, which is why one curve covers the other). From the robot motion simulation in Figure 3.15, we cannot exactly perceive this behavior, but from these plots, we can interpret that the trajectory performs small "pauses" along the waypoints, changing the velocity direction of joints causing the apparent oscillations.

Similar behavior is depicted in Figure 3.19 and Figure 3.20 for the cases (c) and (d), respectively. Therefore, a better joint velocity behavior could be an asset for a better overall performance, which could lead to a better linear speed and improving final times. The improvement of these joint velocities will be addressed in part in Chapter 4, parting initially from the validation of results here obtained.

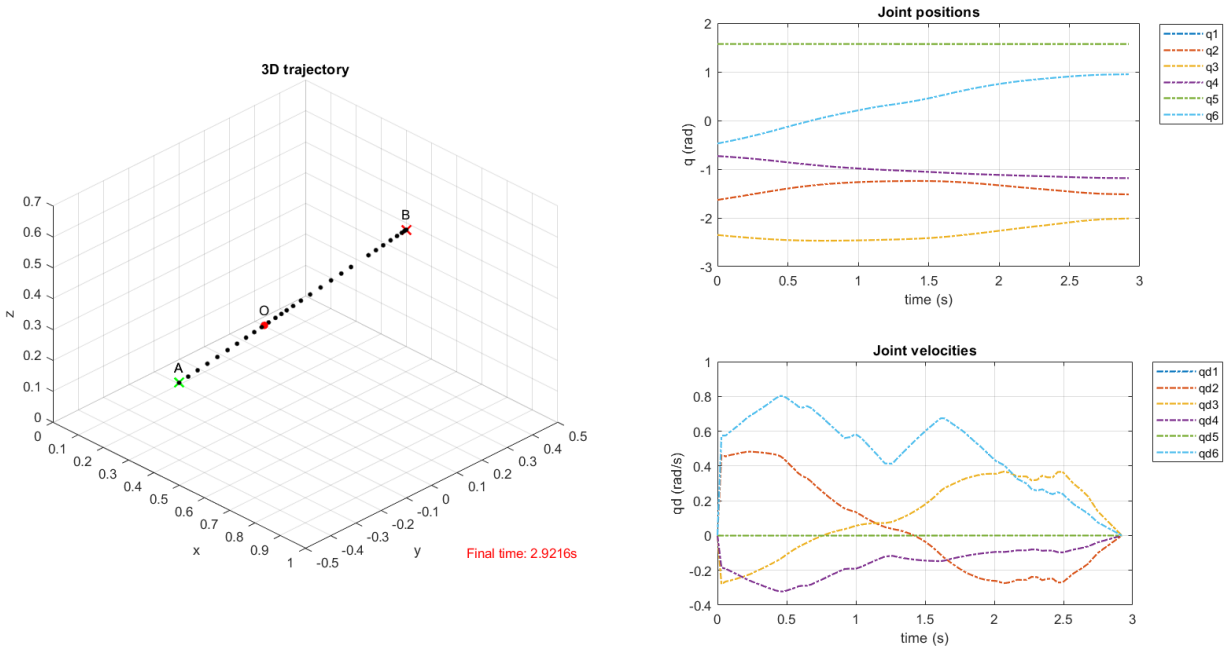


Figure 3.17 Case (a): 3D position trajectory, joint positions, and joint velocities

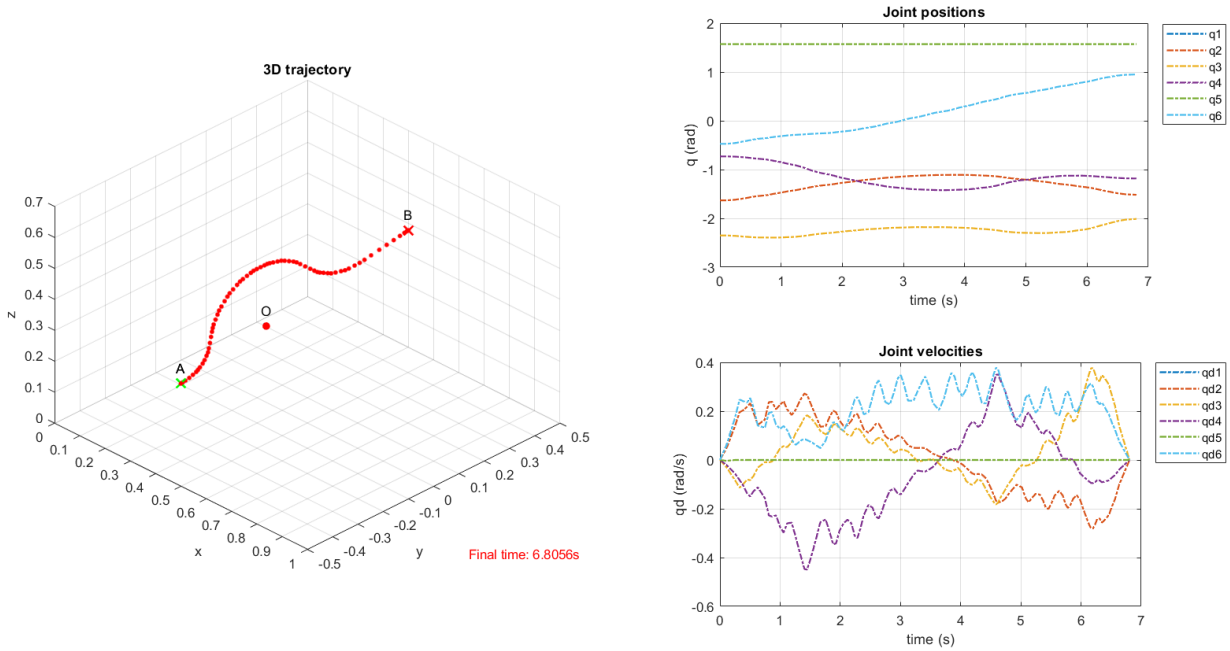


Figure 3.18 Case (b): 3D position trajectory, joint positions, and joint velocities

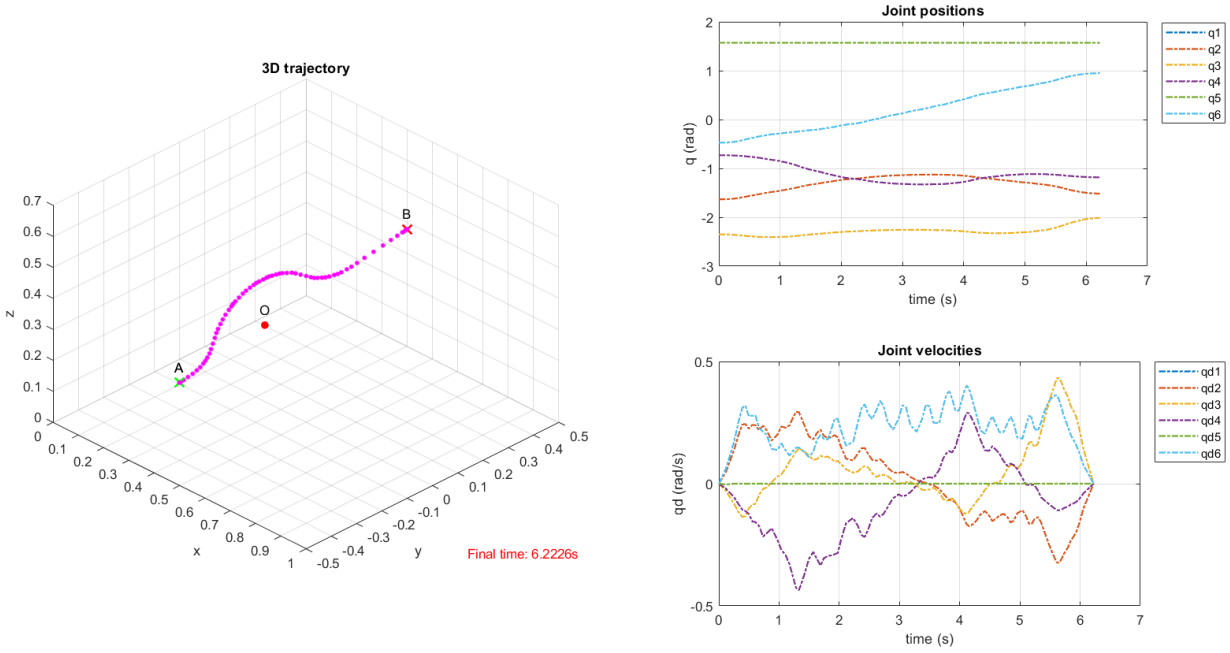


Figure 3.19 Case (c): 3D position trajectory, joint positions, and joint velocities

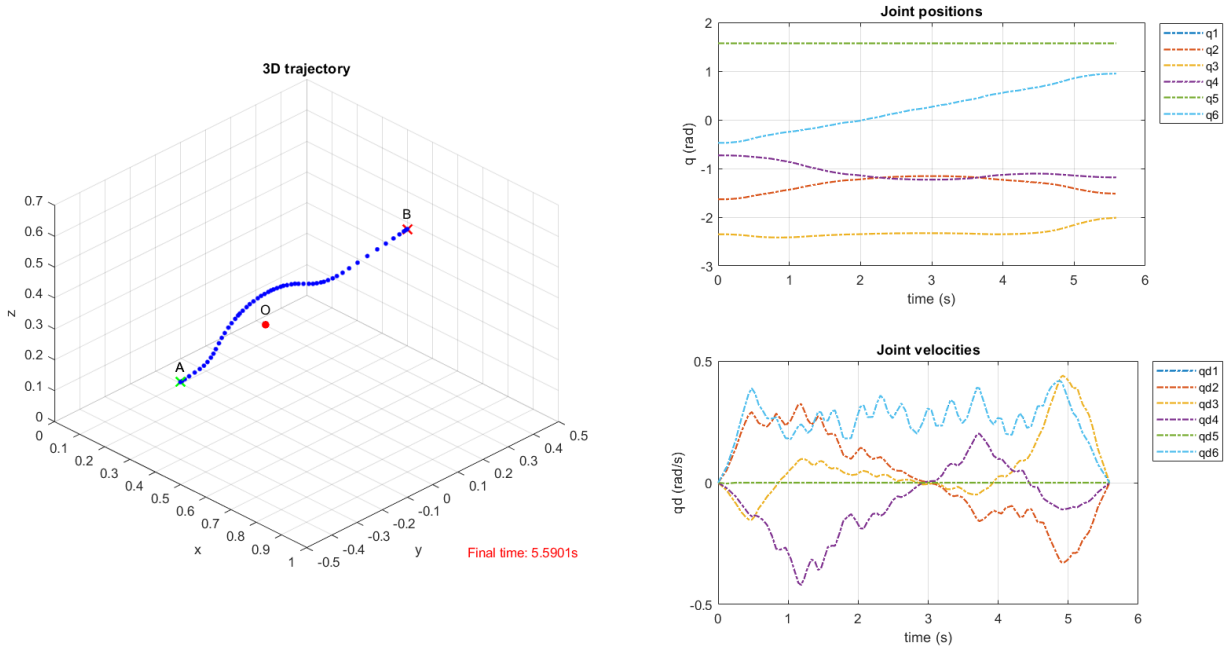


Figure 3.20 Case (d): 3D position trajectory, joint positions, and joint velocities

3.7. Conclusions and perspectives

The obstacle avoidance problem can be approached from several manners, as seen in the literature. Here, we proposed an offline approach to focus in terms of path continuity, final time and average speed of the end-effector.

The collision-free path proposed, a Dubins curve variation, has the advantage of low computation cost (it is uses radius, sagittas, tangents, etc.) and good overall continuity in terms of curvature and tangential vectors, resulting in a C1 continuous path that effectively avoids a given obstacle.

The obstacle particle model allows to examine in detail the performance of the trajectory, the advantage lays in the simplicity, yet scalability, that this approach can allow. Later it would be of interest to use the circumscribed ball model, it would be just needed to adapt the parameters of the arcs to the new obstacle description. The real interest in this chapter was to simply evaluate the performance of collision-free paths and find the best resulting trajectory.

The resulting simulation is prepared accordingly to be easily transferred to a real case scenario. In this sense, the use of already available tools in the field of robotics comes to be a great asset. The recent development of the ROS framework in conjunction with MoveIt! is of great help to push forward into the robotics research problems, however this may come with some disadvantages.

MoveIt! with ROS and the UR10e, allows to seemingly setup all the necessary tools without much problem for simulation and real implementation, thanks to the documentation and the robotics community. However, as this may be good as an entry point, it comes with big problem: we can-not control the Cartesian speed of the end-effector trajectory, only the path without time parametrization.

The resulting simulations here presented are retrieved from the combination of the main tools previously mentioned in conjunction with our obstacle avoidance planner.

As first results, they are promising. They validate the viability of the planner, and some important conclusions can be extracted, in terms of the expected performances in the real case scenario. They show that the obstacle avoidance problem in fact decreases the time

performances, but we found that by simply adjusting one parameter (the sagitta) to the path planner, we can improve those performances. The average speed can be increased, but it could be further improved if we adapt the strategy.

These results are verified and performed under real condition in Chapter 4. Further analysis and computations are proposed, to address the problematics or disadvantages encountered so far in the simulations.

Chapter 4

4. Experimental results and improvements

4.1. Introduction

To continue with the approach proposed in this work, the speed control of the end-effector is crucial to improve the obstacle avoidance trajectories. One of the main setbacks found in the simulation framework from Chapter 3 is that the linear speed of the end-effector depends entirely on the MoveIt! trajectories computation. With that method alone we are then unable to set any given linear speed.

The main purposes of controlling the end-effector linear speed are

i) to assure the security during the avoiding obstacle process,
ii) to improve the total duration of the trajectories, and
iii) to improve the overall motion performance.

Regarding security and safety requirements, collaborative robots used for industrial applications are submitted to the standard ISO 10218-1:2011 and the TS 15066 technical specification (Saenz et al., 2018). Based on these specifications, and according to (Haddadin et al., 2009), a limitation of $0.25 m/s$ of the Tool Center Point (TCP or end-effector) would be adequate to allow human-robot interaction. This speed is intended to allow operators enough time to withdraw or to stop the robots manually (ISO 10218-1, 2011).

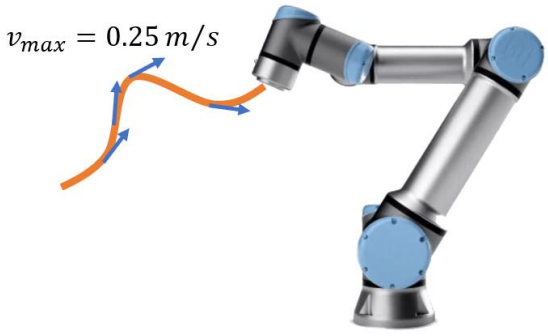


Figure 4.1 Maximal linear speed in human-robot collaboration

Hence, during collaboration tasks, the maximum linear speed that could lead to the most efficient times is $v_{max} = 0.25m/s$. The main ideas that this chapter present are a) the comparison of the initial simulation results from Chapter 3 with the resulting trajectories in the real robot, b) the methodology used to improve the precomputed trajectories by setting a linear speed argument in the algorithm, and c) the evaluation of the results when the end-effector linear speed is controlled at three different values.

4.2. UR10e setup and initial results

4.2.1. Architecture setup

The advantage of using the simulation framework proposed in Chapter 3 is that it allows to transparently execute the same core ROS nodes and libraries already developed. In order to take advantage of the ROS ecosystem, a basic connection through a switch is used as displayed in Figure 4.2. This basic configuration allows to further develop the architecture of larger systems, by adding sensors, actuators, machines, etc., to the same operational network.

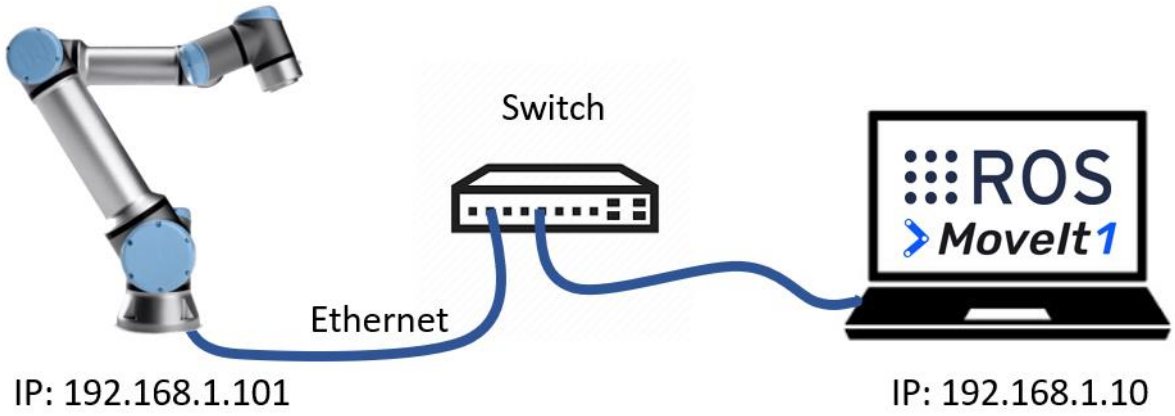


Figure 4.2 Architecture setup: UR with ROS

For the case of UR robots, they are calibrated inside the factory to give the exact inverse and forward kinematics. It is recommended to extract the same calibration information from the robot. In that manner, the factory values can be used by ROS to guaranty the precision of the end-effector. Further steps to launch the necessary ROS driver modules are described in the repository of the Universal Robots ROS Driver (fmauch, 2019/2019). After successfully connecting and setting up all the equipment's correctly, we can control the robot through the

ROS action server, with any client interface such as MoveIt! or the *rqt_joint_trajectory_controller*.

The first results here discussed are based on the same trajectories obtained in the simulations with the MoveIt! framework. Simulations and real robot executions are then compared for validation.

4.2.2. Results with MoveIt! trajectories

The following results replicate those from the simulations in Section 3.6.

As a reminder, the main considerations are:

- Initial pose $\boldsymbol{A} = \left[0.4,\ -0.4,\ 0.25,\ 0, -\pi,\ \frac{\pi}{2}\right]^T$
- Final pose $\boldsymbol{B} = \left[0.5,\ +0.4,\ 0.25,\ 0, -\pi,\ \frac{\pi}{2}\right]^T$
- End-effector with same orientation all along the trajectories
- Obstacle coordinate at $\boldsymbol{O_y} = -0.1\ m$
- Half of the chord from the main avoiding arc $\boldsymbol{r} = 0.20\ m$
- The sagitta of the main avoiding arc $\boldsymbol{h} = \{1.00r, 0.75r, 0.50r\}\ m$
- The 4 cases studied are the ones shown in Figure 4.3

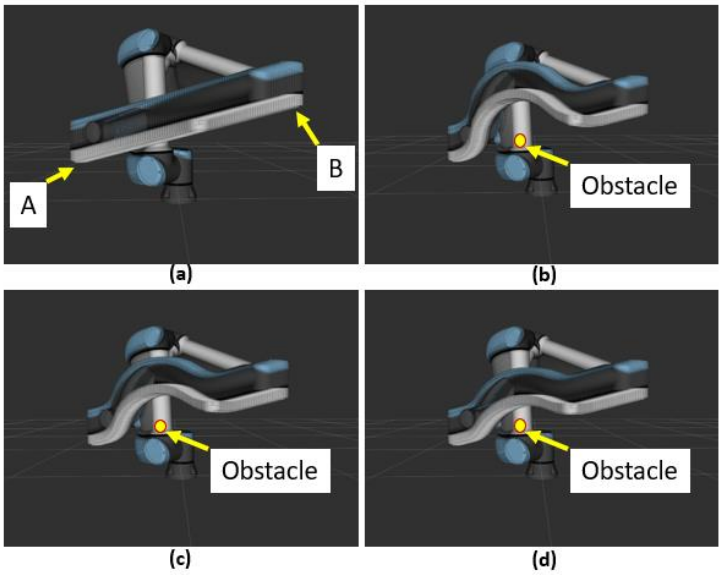


Figure 4.3 Trajectory cases: (a) linear; obstacle avoidance: (b) with h = 1.00r, (c) with h=0.75r, (d) with h=0.50r

Table 4.1 presents the results of the four cases: those from the simulations and those obtained experimentally with the robot UR10e. They present the total distance traveled by the end-effector, the total time taken to complete the trajectories, and their average speed.

Looking at total times and averages speeds between simulations and MoveIt! trajectories executed by the UR10e we can determine a difference of less than 3%. We can consider as good results, because this confirms that the simulations can be reliable to what could happen in real case scenarios.

The percentages between parenthesis in the table reflect the comparison between the linear **case (a) (100%)** vs. the rest of the cases from the same column with +/- a given percentage. This helps to further analyze the performances of the different cases within the same conditions.

From the total time percentages columns, we can conclude that the best performing trajectory is case (d), for both, in the simulations and MoveIt! trajectories actual execution. If we compare the total time columns between each other, we determine that in real the real case scenario cases (b), (c) and (d), are much closer to the ideal case (a), than those from the simulations.

Table 4.1 Quantitative assessment of simulation trajectories and robot actual trajectories execution

Trajectory	Total distance (m)	SIM Total time (s)	SIM Average speed (m/s)	MoveIt! Total time (s)	MoveIt! Average speed (m/s)
Case (a) Linear	**0.833**	**2.921 (100%)**	**0.285 (100%)**	**3.0 (100%)**	**0.278 (100%)**
Case (b) with $h = 1.00r$	**0.989**	**6.806 (+133%)**	**0.145 (-49%)**	**6.76 (+125%)**	**0.146 (-47%)**
Case (c) with $h = 0.75r$	**0.930**	**6.223 (+113%)**	**0.149 (-48%)**	**6.19 (+106%)**	**0.150 (-46%)**
Case (d) with $h = 0.50r$	**0.882**	**5.590 (+91%)**	**0.158 (-45%)**	**5.58 (+86%)**	**0.158 (-43%)**

In terms of average speed, the MoveIt! trajectories actual execution also performs slightly better in comparison to the simulations, with a difference of around 2%. In this regard, MoveIt!

obstacle avoiding trajectory case (d) is the one performing the best, by being the fastest among the three cases (b), (c) and (d).

This leads to the following topic: the maximal speed under collaborative operations. For the ideal linear trajectory of case (a) the average speeds surpass the maximal allowable speed of $v_{max} = 0.25m/s$ (Haddadin et al., 2009; ISO 10218-1, 2011; ISO/TS, 2016), having as average speeds of $0.285m/s$ and $0.278m/s$ (for the simulation and the real scenario, respectively).

Figure 4.4, Figure 4.5, Figure 4.6 and Figure 4.7 depict the different real cases scenarios with the MoveIt trajectories, presenting in each one the 3D position trajectory, the joint positions, joint velocities and the estimated end-effector linear speed.

Comparing these figures with those obtained from the simulations in Chapter 3, we can verify that the joint positions and velocities behave in similar manner, presenting the same potential problems: oscillations in their velocities.

The estimated end-effector linear speed for each of these figures, give us a better idea of their motion performance. We can see that in all four cases there are oscillations with considerable amplitude, which results in the effector slowing down its speed at different points of the trajectory, which we judge as inefficient.

Table 4.2 Motion duration with linear speeds greater than 0.25m/s and their momentary peaks

Trajectory	Total distance (m)	MoveIt! Total time (s)	MoveIt! Average speed (m/s)	Time above 0.25m/s (s)	Peak speed (m/s)
Case (a) Linear	**0.833**	**3.0 (100%)**	**0.278 (100%)**	**2.0**	**0.38**
Case (b) with $h = 1.00r$	**0.989**	**6.76 (+125%)**	**0.146 (-47%)**	**0.8**	**0.31**
Case (c) with $h = 0.75r$	**0.930**	**6.19 (+106%)**	**0.150 (-46%)**	**0.9**	**0.35**
Case (d) with $h = 0.50r$	**0.882**	**5.58 (+86%)**	**0.158 (-43%)**	**1.0**	**0.37**

By looking at these plots, we can confirm that each one of the four cases surpass the reglementary maximal linear speed $v_{max} = 0.25m/s$. Case (a) in Figure 4.4 is the one

surpassing this speed for approximately $2s$ in total which is 66% of its whole trajectory, having a speed peak of approximately $0.38m/s$. For case (d) in Figure 4.7, it surpasses this speed for $1s$, and with a peak of approximately $0.37m/s$. Table 4.2 presents more clearly this information.

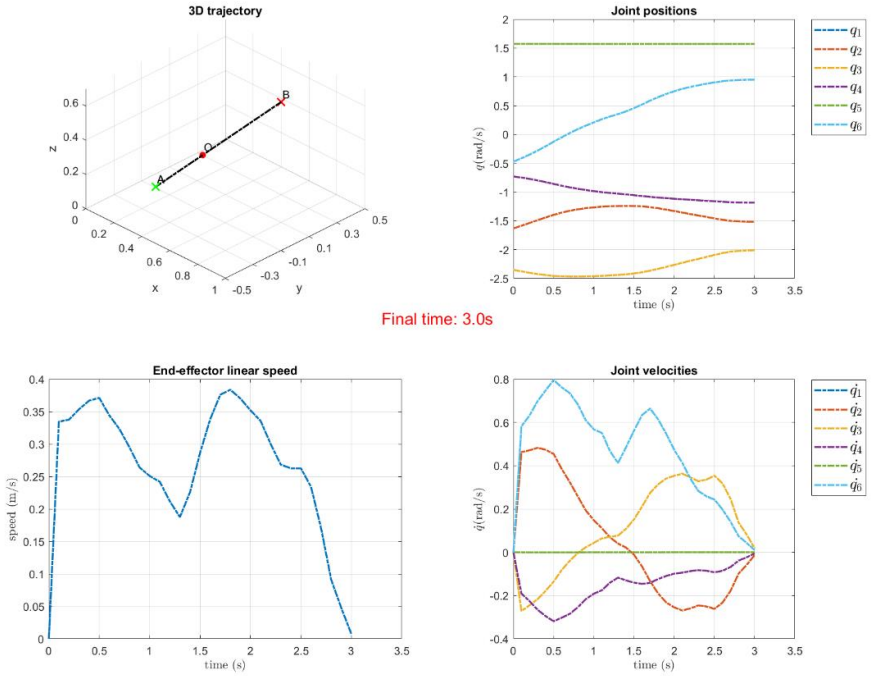


Figure 4.4 MoveIt! case (a): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

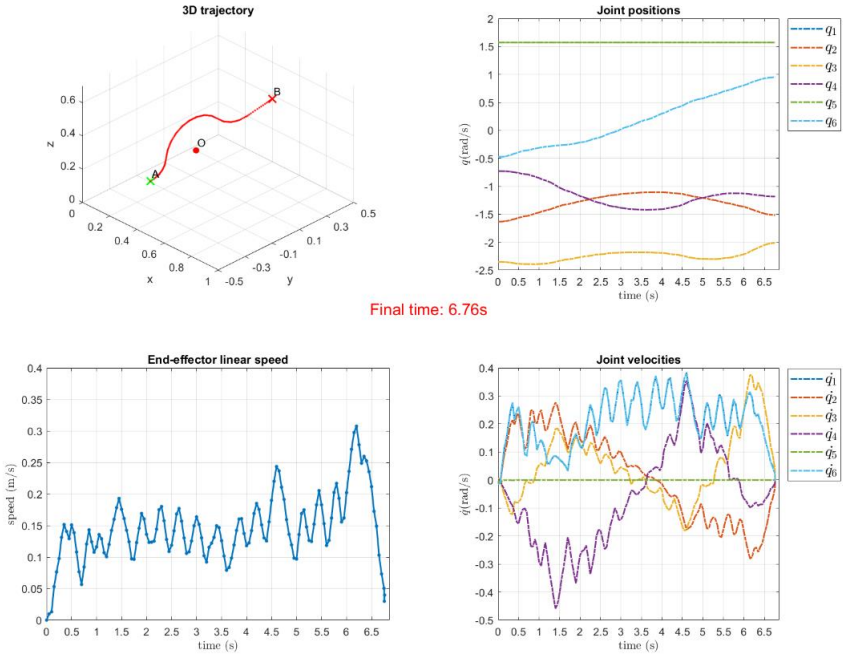


Figure 4.5 MoveIt! case (b): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

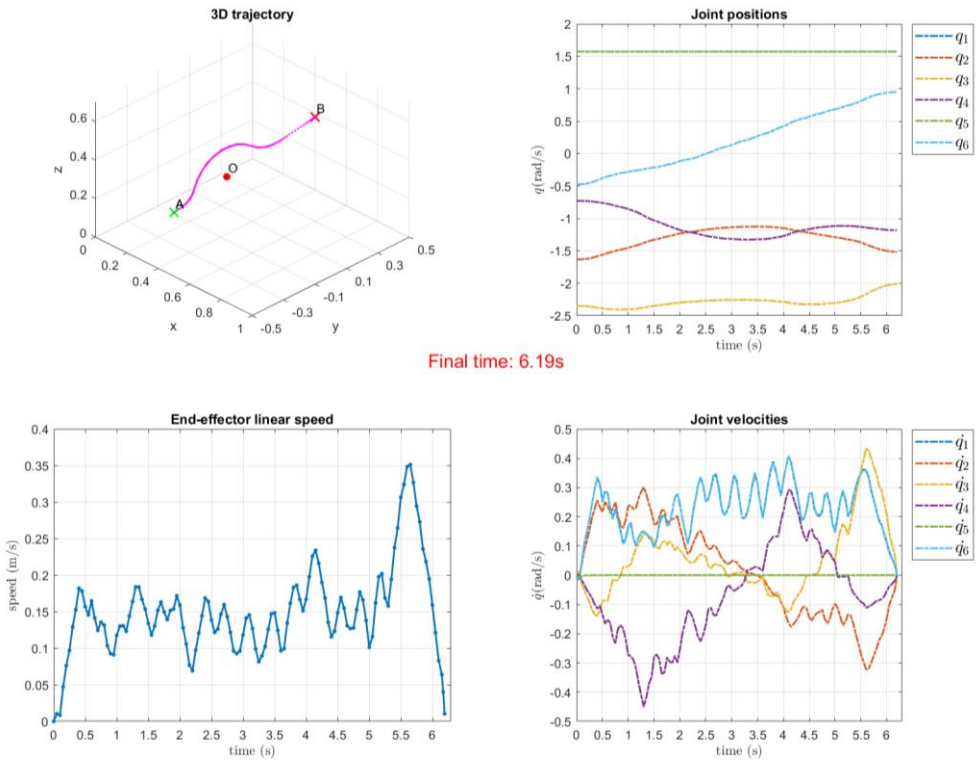


Figure 4.6 MoveIt! case (c): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

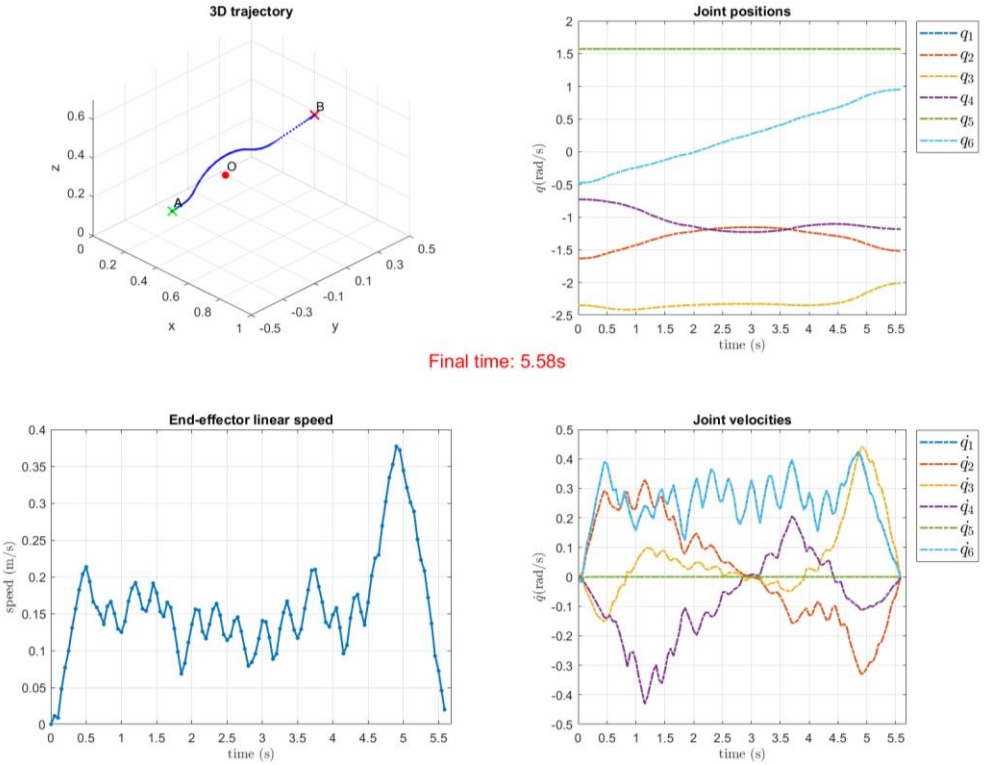


Figure 4.7 MoveIt! case (d): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

These findings show that the trajectories should be treated with further computations, in order to meet the regulations given by ISO standards, and additionally, to improve the overall performance by reducing the resulting oscillation amplitude of the linear end-effector speed.

4.3. New waypoints computation for linear speed control

Interpolation methods have been widely used in robotics to describe trajectories. In Chapter 3 initially we generated our path plan by following certain smoothing conditions from the Dubins curves variation. Then these waypoints were introduced to the MoveIt! framework to obtain the resulting trajectories. However, the trajectory generation methodology lacked the ability to control the linear speed of the desired trajectory. Therefore, the investigation of interpolations methods to generate more flexible trajectories was the next logical step.

In spite of the speed control limitation being the main setback from the previous trajectories computation, they still possess two valuable attributes: i) the different joint configurations for each of the Cartesian waypoints are computed, so that the ROS action server can receive the joint trajectory waypoints, and ii) these joint trajectory waypoints have been assessed by the MoveIt! framework assuring the viability of the execution of a given path, considering the predefined workspace.

In this section we propose the use of the Piecewise Cubic Hermite Interpolating Polynomial to obtain the new trajectories. The purpose is to be able to modify the time-scale from the waypoints from previous computed trajectories by introducing a maximal linear speed parameter, in order to smooth the end-effector motion and control its maximum linear speed.

4.3.1. Polynomial interpolation method

Interpolation technique was first proposed by Waring in 1779 (Waring, 1779) and then by Lagrange in 1795 (Jeffreys & Jeffreys, 1999). The problem consists on finding new data points from a given discrete set of known data points (Steffensen, 2013). Interpolation for robotics is widely used for path smoothing and trajectories generation. Related work can be found in (R. Zhao & Sidobre, 2015): they propose a sequence of cubic polynomial functions to describe the trajectories, using an intermediate planner to generate non-linear time-scaling functions or to replan new trajectories for obstacle avoidance.

4.3.1.1. Piecewise Cubic Hermite Interpolating Polynomial

Piecewise Cubic Hermite Interpolating Polynomial, also called PCHIP, is based on the Lagrange interpolation and it is only locally polynomial. It uses low-degree polynomials at each of the intervals. Splines are a type of PCHIP that guarantee a piecewise cubic and second derivative as zero at the end points. However, they may present overshoots, or going "off-path". In the contrary, the simple PCHIP is not as smooth as the spline, but it guarantees that the interpolated joint positions do not violate joint limits, as long as the waypoints do not.

1.1.1.1.1. PCHIP definition

Let $l(x)$ be an n^{th} degree polynomial with zeros at $x_1, \dots, x_n$. Then the fundamental Hermit interpolating polynomials (Weisstein, n.d.) of the first and second kinds are defined by

$$h_v{}^{(1)}(x) = \left[1 - \frac{l''(x_v)}{l'(x_v)}(x - x_v)\right][l_v(x)]^2$$

And

$$h_v{}^{(2)}(x) = (x - x_v)[l_v(x)]^2$$

For $v = 1,2, \dots n$, where the fundamental polynomials of Lagrange interpolation are defined by

$$l_v(x) = \frac{l(x)}{l'(x_v)(x - x_v)}$$

4.3.2. PCHIP from original MoveIt! waypoints and time data points

The interaction between some of the main algorithms and modules for the PCHIP calculations are shown in Figure 4.8. First the resulting Cartesian waypoints matrix P of dimension $6 \times n_{pp}$ from the path planning are introduced to the MoveIt! framework to compute the resulting trajectory matrix $Traj_m$ of dimension $6 \times n_m$ and the MoveIt! time points array t_m of $1 \times n_P$. These are then used to compute the PCHIP interpolation and resulting in the $Traj_P$ of dimension $6 \times n_P$ and the PCHIP time points t_P of dimension $1 \times n_P$, being $n_P = n_m + 2$ to smooth the first and last data points and force zero velocities.

P: $6 \times n_{pp}$ matrix with the path plan Cartesian positions $[x, y, z, roll, pitch, yaw]'$ and n_{pp} data points.

$Traj_m$: $6 \times n_m$ matrix with the MoveIt! joint positions $[q_1, q_2, q_3, q_4, q_5, q_6]'$ and n_m data points.

t_m: $1 \times n_m$ array with MoveIt! time points.

$Traj_P$: $6 \times n_P$ matrix with the PCHIP joint positions $[q_1, q_2, q_3, q_4, q_5, q_6]'$ and n_P data points.

t_P: $1 \times n_P$ array with PCHIP time points.

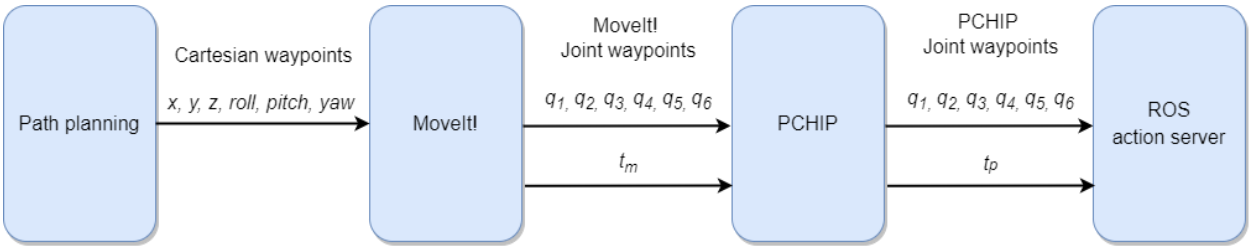


Figure 4.8 PCHIP from original MoveIt! waypoints and time data points

4.3.2.1. Initial PCHIP results from MoveIt! waypoints and time points

Figure 4.10 shows the corresponding results from the PCHIP interpolation.

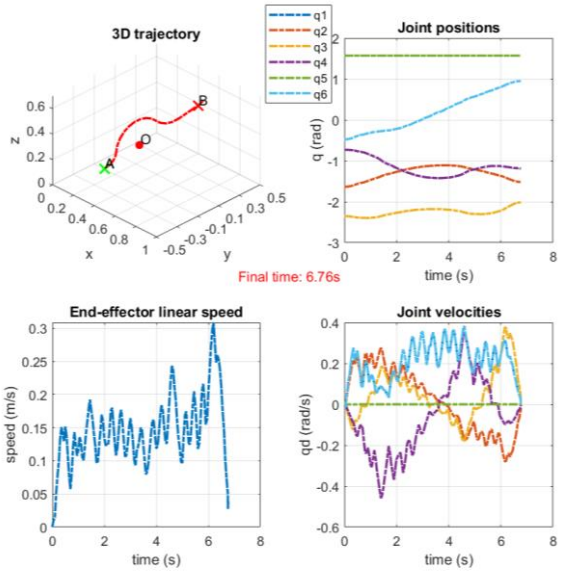


Figure 4.9 MoveIt! results for case (b)

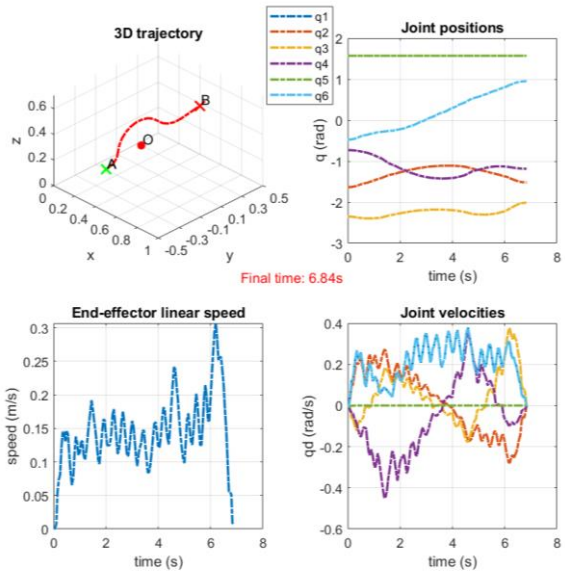


Figure 4.10 PCHIP results with MoveIt trajectory for case (b)

Figure 4.9 is shown to compare the PCHIP with the previous MoveIt! results.

As it can be seen both are quite similar results, this most certainly is because the time points used for interpolation inside the PCHIP block were based on the MoveIt! time points t_m with a slight off-set of $t_{offset} = 0.01s$. This tells us that MoveIt! framework probably uses a cubic interpolation as well. The only visible differences come from the initial and final portions of the trajectories, due to the inclusion of the first and last data points used to force zero velocities at the extremes.

The next step is then to change the PCHIP trajectory so that we can control the linear speed of the end-effector. This is achievable if the $Traj_m$ is transformed into Cartesian positions, and from there modify the time scale given a linear speed control argument.

4.3.3. PCHIP with linear speed control

The previous approach is then further developed. The idea is to introduce a maximal linear speed that controls the end-effector behavior.

In order to have more control of the linear speed of the end-effector, the resulting general block diagram changes as it is shown in Figure 4.11. In here we add two user input values: desired maximum speed v_{in} and the rise time T_{rt} (which is also used as braking time) of the end-effector.

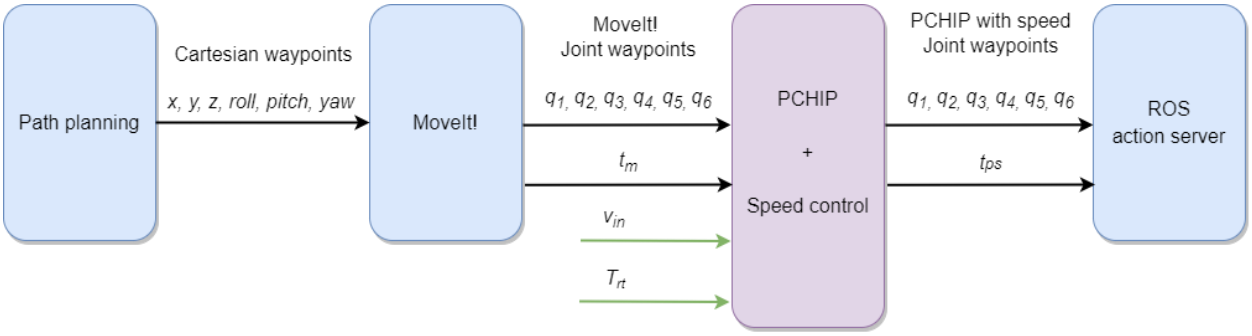


Figure 4.11 General block diagram for PCHIP with linear speed control

Detailed information of the PCHIP + Speed control block is presented in Figure 4.12. In here the forward kinematics block receives the MoveIt! joint waypoints and converts them into Cartesian waypoints P_{pos} (the computation is necessary because $n_{pp} \neq n_m$). The linear speed control block computes the resulting speed trajectory $Traj_S$ with its respective speed time points

array t_S, having as input the P_{pos}, t_m, v_{in} and T_{rt}. Afterwards, the PCHIP block computes the PCHIP-speed $Traj_{PS}$ with its respective PCHIP-speed time points array t_{PS}.

P_{pos}: $3 \times n_{pp}$ matrix with $[x, y, z]'$ vectors

$Traj_S$: $6 \times n_S$ matrix with the speed control joint positions $[q_1, q_2, q_3, q_4, q_5, q_6]'$ and n_S data points.

t_S: $1 \times n_S$ array with speed control time points.

$Traj_{PS}$: $6 \times n_{PS}$ matrix with the PCHIP-speed joint positions $[q_1, q_2, q_3, q_4, q_5, q_6]'$ and n_{PS} data points.

t_{PS}: $1 \times n_{PS}$ array with PCHIP-speed time points.

Experimental results from the global block diagram configuration are presented in the following section.

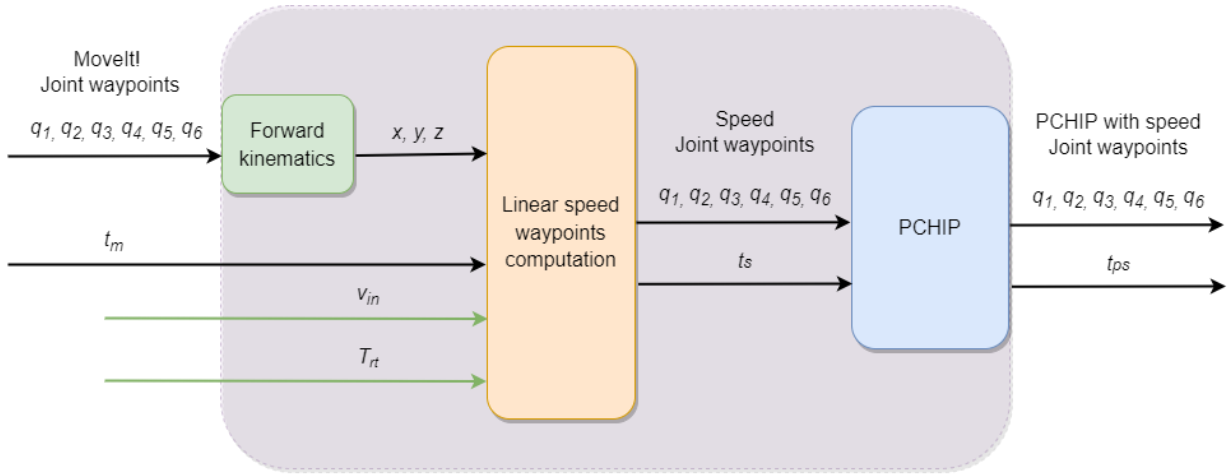


Figure 4.12 PCHIP with linear speed control block from previous figure in detail

4.4. Results with linear speed control

The following results consist of three set of trajectories with a maximal input linear speed of $v_{in} = \{0.149, 0.200, 0.250\} m/s$, respectively.

The results are discussed in terms of total distance traveled, total time and average speed for each one of the four cases previously seen (cases (a), (b), (c) and (d)). They as well present some comparisons with the original MoveIt! results from section 4.2.2.

4.4.1. Maximal linear speed of 0.149 m/s

The first set of trajectories comes from maximal input linear speed of $v_{in} = 0.149m/s$. This represents 60% of the maximal linear speed $v_{max} = 0.250m/s$.

Table 4.3 presents these global results in the same terms that have been discussed previously. We can verify that the improvement between MoveIt! trajectories and PCHIP-speed are remarkable. Case (b), (c) and (d) are much closer to the ideal case (a) in PCHIP, than in MoveIt! (by looking at the percentages differences).

In terms of total time, the PCHIP-speed avoiding trajectory of case (d) is as well the one performing the best between the three avoiding trajectory cases, having just a delay of 9% with respect to the ideal linear trajectory case (a).

In terms of average speeds, they all stay approximately in the same range, and all below $0.25m/s$.

Table 4.3 MoveIt! trajectories vs PCHIP-speed trajectories with $v_{in} = 0.149m/s$: total distance traveled, total time and average speed comparison for each case trajectory

Trajectory	Total distance (m)	MoveIt! Total time (s)	MoveIt! Average speed (m/s)	PCHIP 0.149m/s Total time (s)	PCHIP 0.149m/s Average speed (m/s)
Case (a) Linear	**0.833**	**3.0 (100%)**	**0.278 (100%)**	**5.7 (100%)**	**0.146 (100%)**
Case (b) with $h = 1.00r$	**0.989**	**6.76 (+125%)**	**0.146 (-47%)**	**6.93 (+22%)**	**0.143 (-2%)**
Case (c) with $h = 0.75r$	**0.930**	**6.19 (+106%)**	**0.150 (-46%)**	**6.54 (+15%)**	**0.142 (-3%)**
Case (d) with $h = 0.50r$	**0.882**	**5.58 (+86%)**	**0.158 (-43%)**	**6.23 (+9%)**	**0.142 (-3%)**

The individual PCHIP-speed plot performances are shown in Figure 4.13, Figure 4.14, Figure 4.15 and Figure 4.16.

Comparing the avoiding trajectories of PCHIP-speed with those from MoveIt!, we can confirm that the joint velocities do not oscillate as before but reach to quasi-stables velocities at certain given moments. This means that the joints are not compensating much the joint positions by changing the direction of motion.

Similarly, if we look at the estimated end-effector linear speed, none of them surpasses the v_{max} of 0.25m/s. In the middle phase we can see that they stay in average at the given input speed of $v_{in} = 0.149\text{m/s}$.

The only noticeable arguable problem of the avoiding trajectories is that they present some apparent undesired performances at the end of the motion.

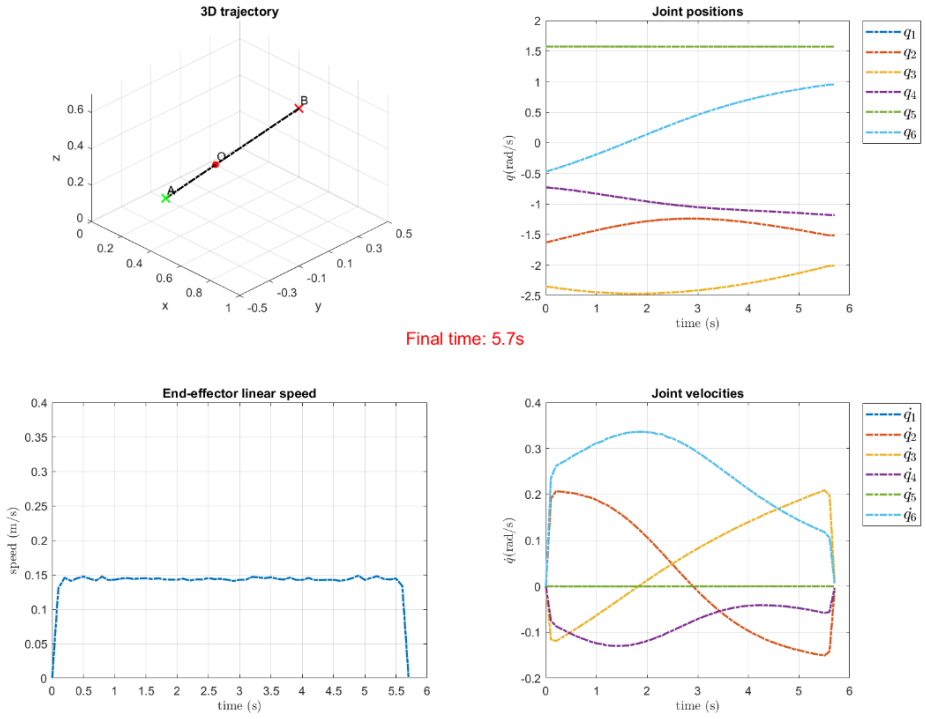


Figure 4.13 PCHIP-speed trajectory with $v_{in} = 0.149m/s$ for case (a): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

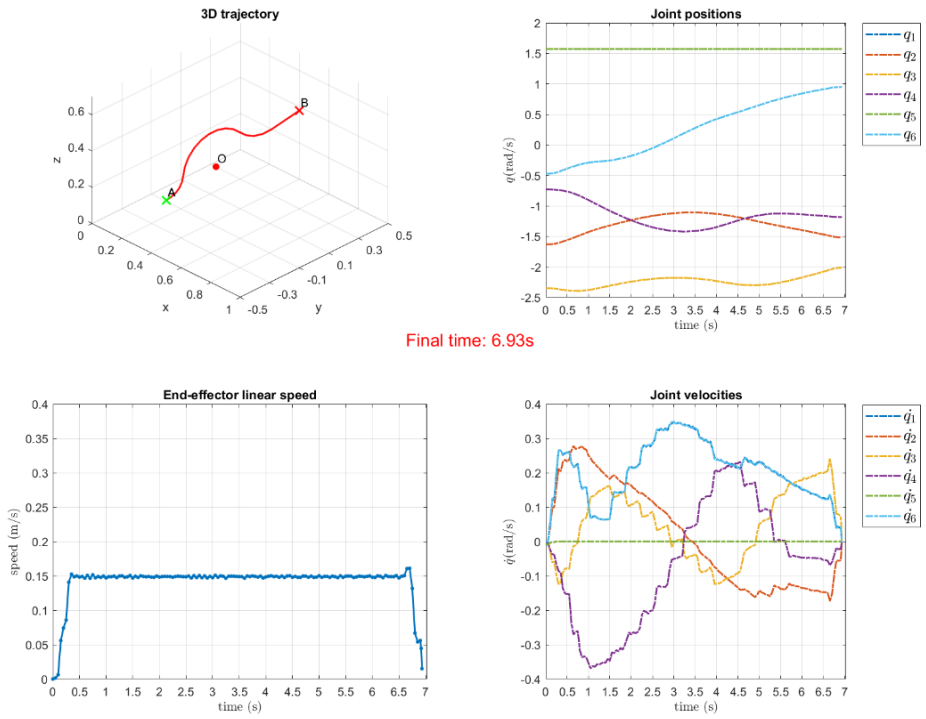


Figure 4.14 PCHIP-speed trajectory with $v_{in} = 0.149m/s$ for case (b): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

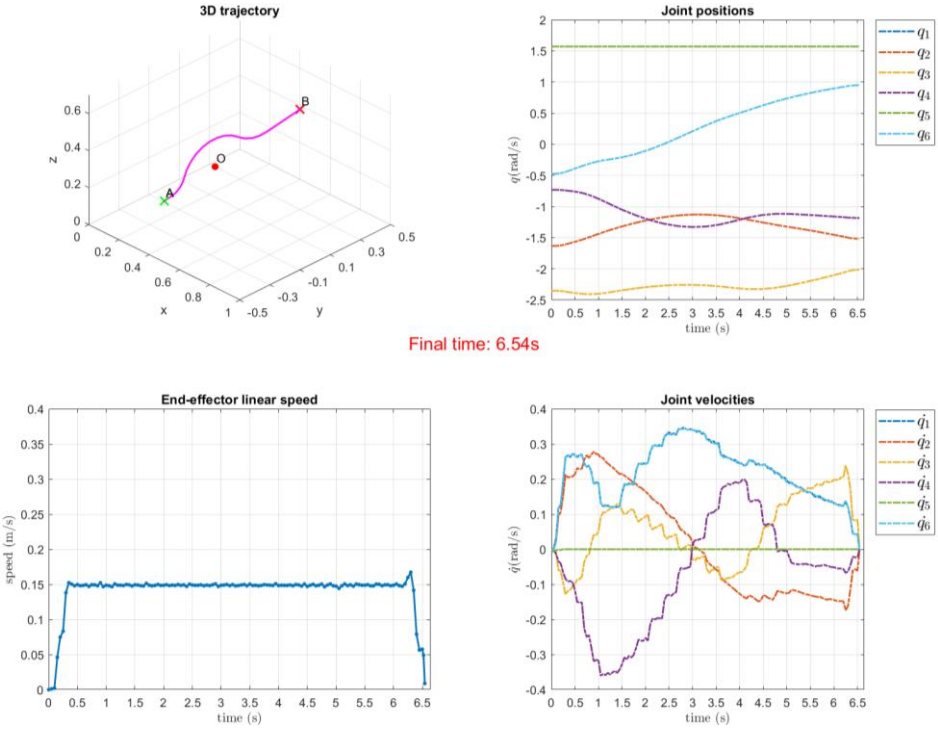


Figure 4.15 PCHIP-speed trajectory with $v_{in} = 0.149m/s$ for case (c): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

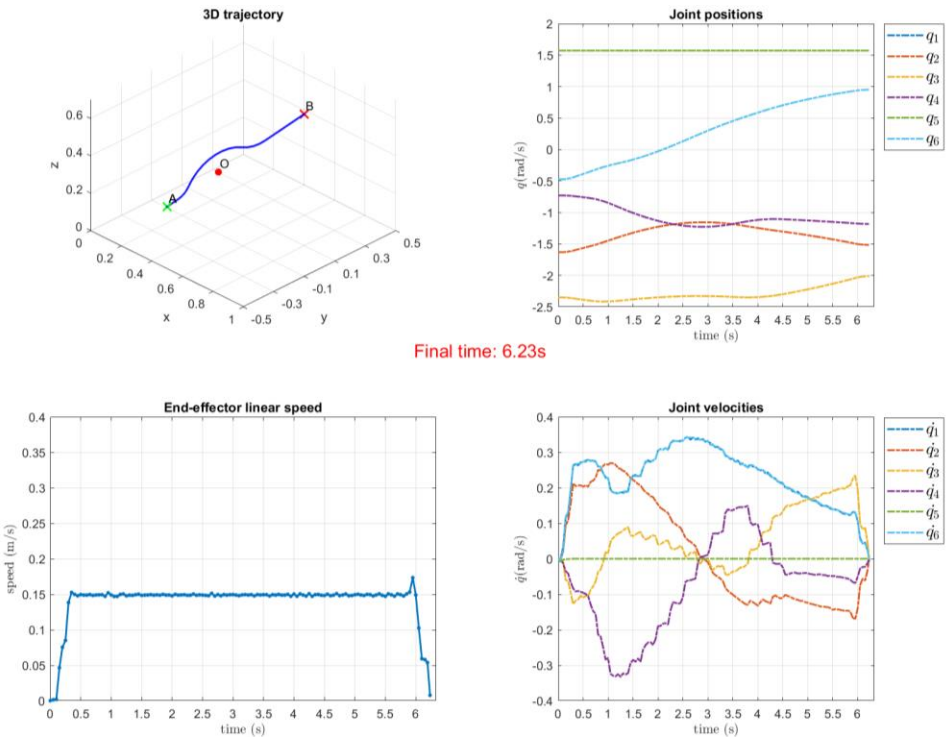


Figure 4.16 PCHIP-speed trajectory with $v_{in} = 0.149m/s$ for case (d): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

4.4.2. Maximal linear speed of 0.200 m/s

Similarly, the second set of trajectories comes from a maximal input linear speed of $v_{in} = 0.200m/s$. This represents 80% of the maximal linear speed $v_{max} = 0.250m/s$.

Table 4.4 presents these global results in the same terms that have been discussed previously. And again, the best avoiding trajectory is case (d) in terms of total trajectory time, having a delay of +10% with respect to case (a). In general, the new trajectories, outperform the MoveIt! original trajectories, when comparing the avoiding trajectories with their respective ideal linear trajectory.

In terms of average speeds, they all stay approximately in the same range, and all below $0.25m/s$.

The individual PCHIP-speed plot performances are shown in Figure 4.17, Figure 4.18, Figure 4.19 and Figure 4.20

Joint velocities performance results are quite similar to those from PCHIP-speed trajectories with $v_{in} = 0.149m/s$, which means that the same conclusion apply: no more questionable oscillations.

In all cases, the estimated end-effector linear speed, in the middle section, stays at an average speed of $v_{in} = 0.200\text{m/s} < v_{max}$. And again, the only arguable problem of the avoiding trajectories is that they present a slight unwanted performance at the end of the motion.

Table 4.4 MoveIt! trajectories vs PCHIP-speed trajectories with v_in= 0.200m/s: total distance traveled, total time and average speed comparison for each case trajectory

Trajectory	Total distance (m)	MoveIt! Total time (s)	MoveIt! Average speed (m/s)	PCHIP 0.200m/s Total time (s)	PCHIP 0.200m/s Average speed (m/s)
Case (a) Linear	**0.833**	**3.0 (100%)**	**0.278 (100%)**	**4.3 (100%)**	**0.194 (100%)**
Case (b) with $h = 1.00r$	**0.989**	**6.76 (+125%)**	**0.146 (-47%)**	**5.34 (+24%)**	**0.185 (-5%)**
Case (c) with $h = 0.75r$	**0.930**	**6.19 (+106%)**	**0.150 (-46%)**	**5.05 (+17%)**	**0.184 (-5%)**
Case (d) with $h = 0.50r$	**0.882**	**5.58 (+86%)**	**0.158 (-43%)**	**4.75 (+10%)**	**0.185 (-5%)**

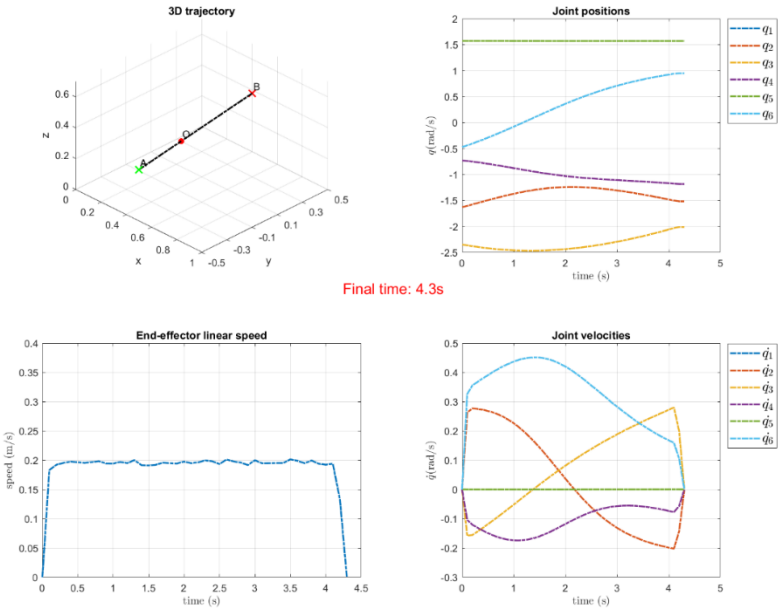


Figure 4.17 PCHIP-speed trajectory with $v_{in} = 0.200m/s$ for case (a): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

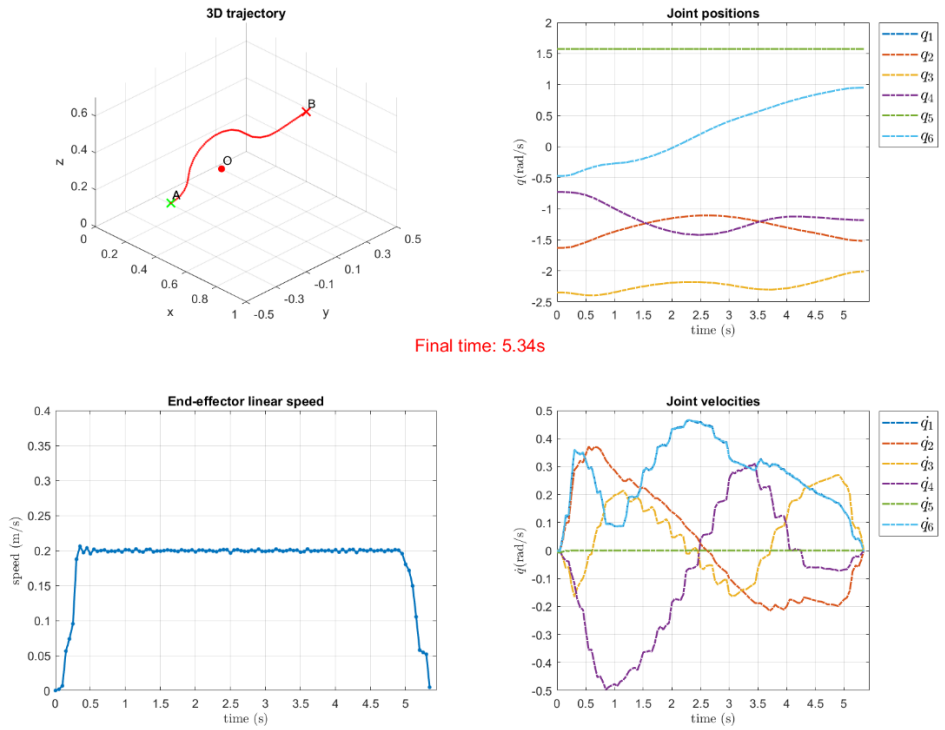


Figure 4.18 PCHIP-speed trajectory with $v_{in} = 0.200m/s$ for case (b): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

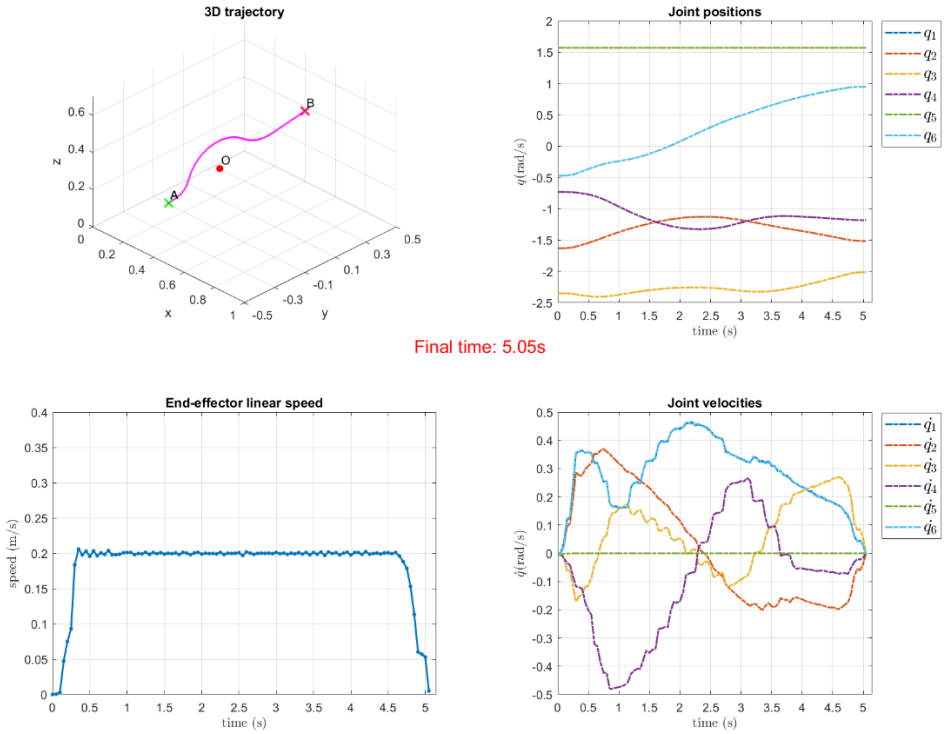


Figure 4.19 PCHIP-speed trajectory with $v_{in} = 0.200m/s$ for case (c): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

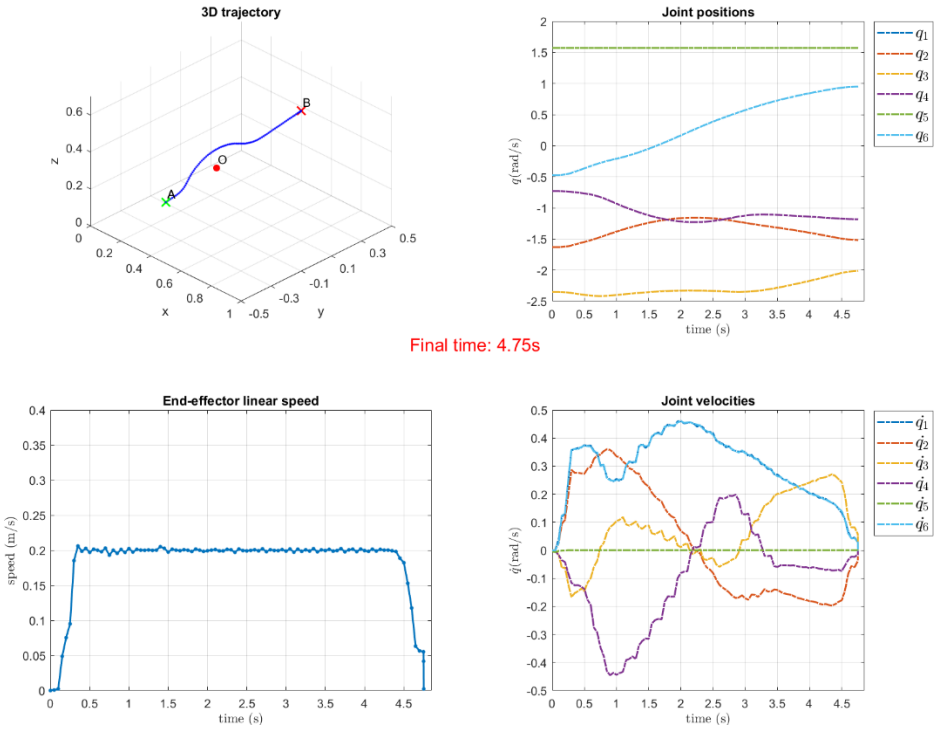


Figure 4.20 PCHIP-speed trajectory with $v_{in} = 0.200m/s$ for case (d): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

4.4.3. Maximal linear speed of 0.250 m/s

The third set of trajectories comes from maximal input linear speed of $v_{in} = 0.250m/s$. This is set to the same as the maximal linear speed $v_{max} = 0.250m/s$.

Table 4.5 presents these global results in the same terms that have been discussed previously. The best avoiding trajectory is case (d) in terms of total trajectory time, having a delay of +13% with respect to case (a). In general, the new trajectories, outperform the MoveIt! original trajectories, when comparing the avoiding trajectories with their respective ideal linear trajectory.

In terms of average speeds, they all stay approximately in the same range, and all below $0.25m/s$.

The individual PCHIP-speed plot performances are shown Figure 4.21, Figure 4.22, Figure 4.23 and Figure 4.24

Joint velocities performance results are quite similar to those from PCHIP-speed trajectories with $v_{in} = \{0.149, 0.200\} m/s$, which means that the same conclusion apply: no more questionable oscillations.

Table 4.5 MoveIt! trajectories vs PCHIP-speed trajectories with $v\_in = 0.149 m/s$: total distance traveled, total time and average speed comparison for each case trajectory

Trajectory	Total distance (m)	MoveIt! Total time (s)	MoveIt! Average speed (m/s)	PCHIP 0.250m/s Total time (s)	PCHIP 0.250m/s Average speed (m/s)
Case (a) Linear	**0.833**	**3.0 (100%)**	**0.278 (100%)**	**3.5 (100%)**	**0.238 (100%)**
Case (b) with $h = 1.00r$	**0.989**	**6.76 (+125%)**	**0.146 (-47%)**	**4.37 (+25%)**	**0.226 (-5%)**
Case (c) with $h = 0.75r$	**0.930**	**6.19 (+106%)**	**0.150 (-46%)**	**4.14 (+18%)**	**0.225 (-6%)**
Case (d) with $h = 0.50r$	**0.882**	**5.58 (+86%)**	**0.158 (-43%)**	**3.95 (+13%)**	**0.223 (-6%)**

In all cases, the estimated end-effector linear in the middle section stays at an average speed of $v_{in} = 0.250\text{m/s}$, with maximum peak values of $0.26 m/s$. This could be arguable a passable maximal momentary linear speed with an error of less than 5% with respect from $v_{max} = 0.25 m/s$.

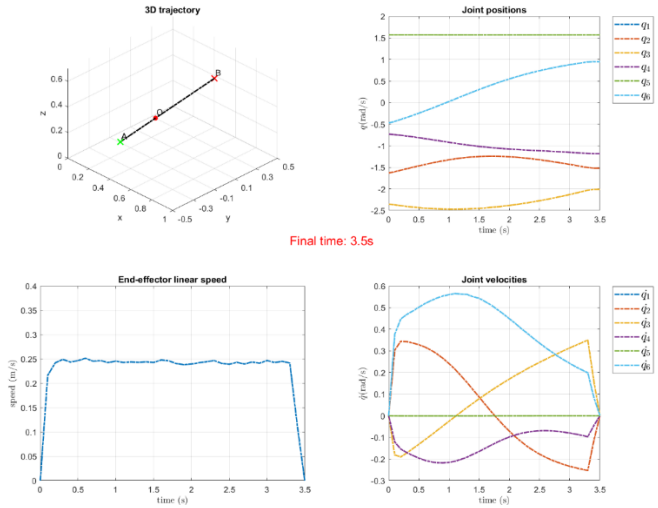


Figure 4.21 PCHIP-speed trajectory with $v_{in} = 0.250 m/s$ for case (a): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

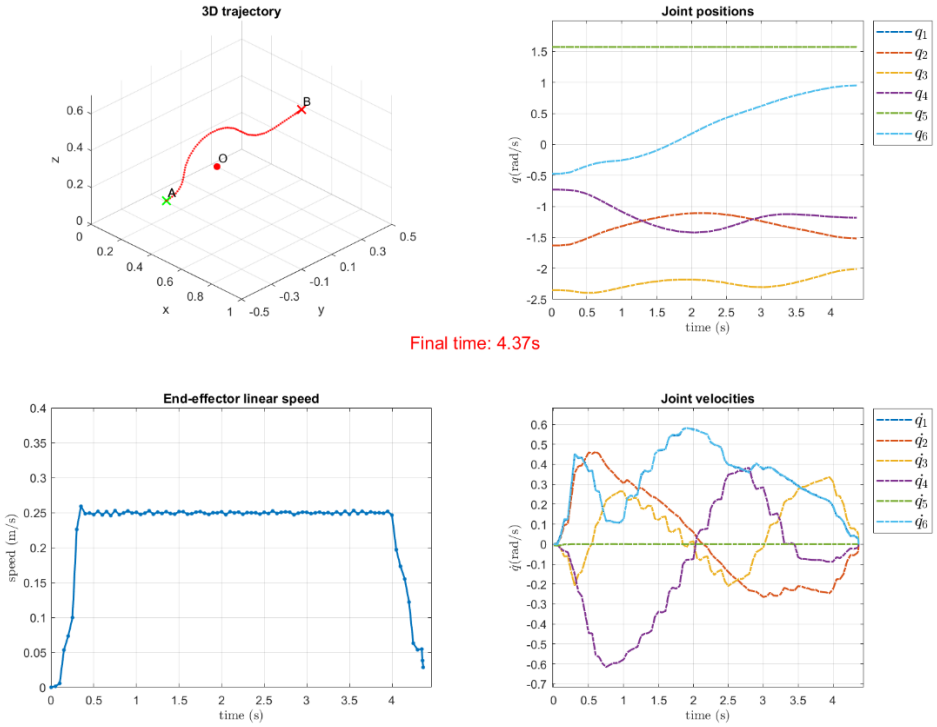


Figure 4.22 PCHIP-speed trajectory with $v_{in} = 0.250m/s$ for case (b): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

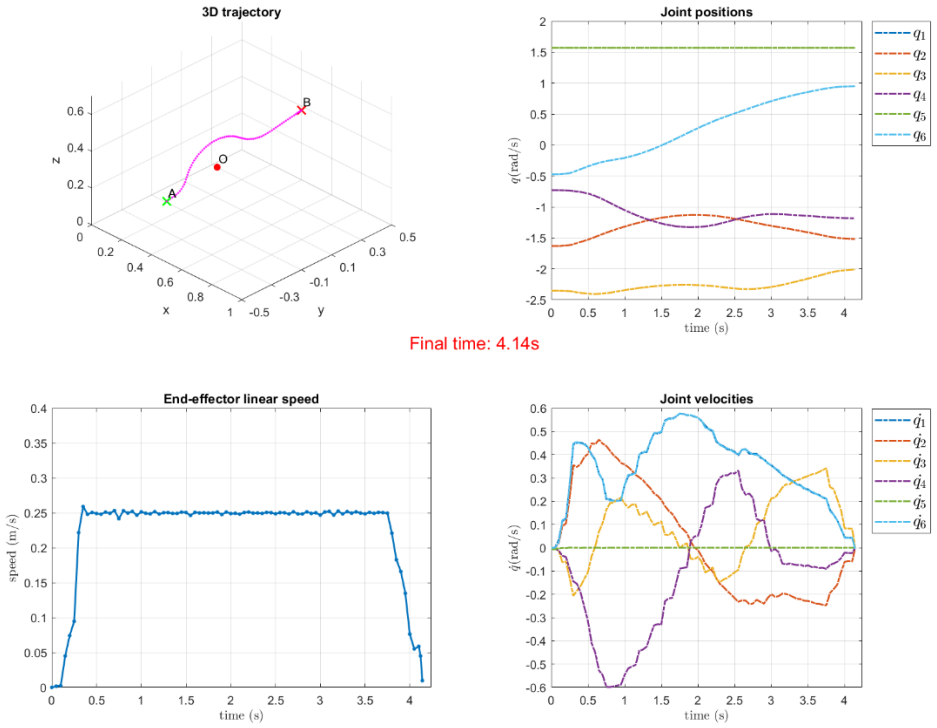


Figure 4.23 PCHIP-speed trajectory with $v_{in} = 0.250m/s$ for case (c): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

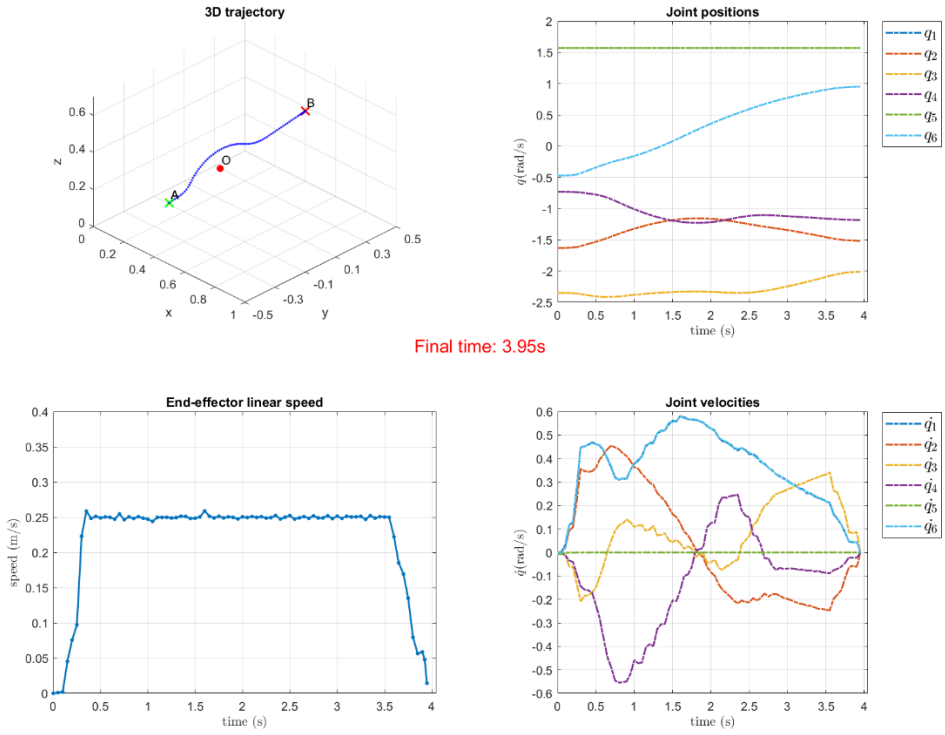


Figure 4.24 PCHIP-speed trajectory with $v_{in} = 0.250m/s$ for case (d): 3D position trajectory, joint positions, joint velocities, and estimated end-effector linear speed

And again, the only arguable problem of the avoiding trajectories is that they present some apparent undesired performances at the end of the motion.

4.5. Conclusions and perspectives

This chapter presented different trajectories implementations with the UR10e collaborative robot for validation. The trajectories are evaluated principally in terms of motion total duration, average linear speeds, momentary linear speeds during motion, and joint velocities performances.

The simulation framework proposed in Chapter 3 became one of the most important parts for the development of the work here presented. It provided an initial proof of concept, and to transparently execute the same core ROS nodes in real case scenarios. The initial results confirm the trajectories obtained through the simulations, with better overall performances. However, they do not meet the safety speed limits requirements.

MoveIt! framework allows the obtention of valid trajectories based on the initial workspace configuration, but the lack of flexibility for the linear speed manipulation became one

of the principal setbacks of this methodology. For this reason, a new approach using interpolations methods are proposed. The joint trajectories that are already computed and validated are then recalculated with PCHIP, to control the linear speed of the end-effector by user input.

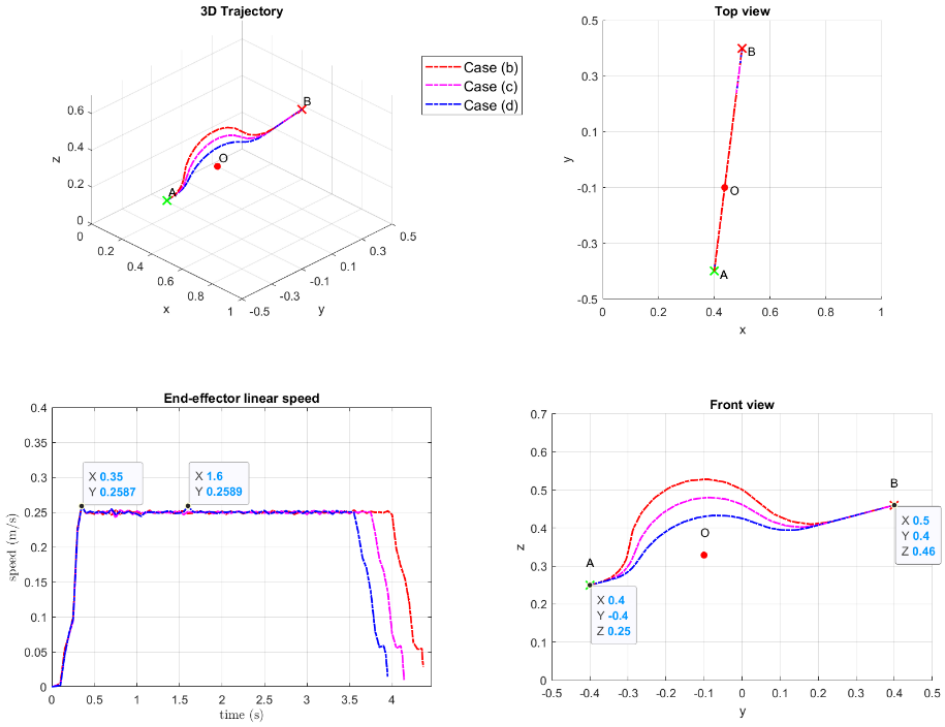


Figure 4.25 PCHIP-speed obstacle avoidance trajectories with $v_{in} = 0.250m/s$

The three different set of results obtained from maximal linear user input $v_{in} = \{0.149, 0.200, 0.250\}m/s$ allow to appreciate the different performances of the four cases scenarios (a) linear ideal trajectory, obstacle avoidance trajectories (b), (c) and (d), with a sagitta of $h = \{1.00r, 0.75r, 0.50r\}$ and $r = 0.20m$. All of them presented considerable improvements in terms of average speeds, without exceeding the maximal reglementary speed $v_{max} = 0.25m/s$. The only arguable case could be for trajectories PCHIP with $v_{in} = 0.250m/s$ in section 4.4.3, where some momentary peaks exceeded the value with less than 5%, but it could be considered as sufficient, as in average they stay at speed of $0.25m/s$ (see Figure 4.25).

Obstacle avoidance trajectories for case (d) were the best trajectories in terms of total time motion. These results can be justified because they are proportionally dependent from the

total distance traveled, and when $h = 0.50r$, they perform the shortest distance of the 3 different obstacle avoidance trajectory cases.

So, in conclusion, the best performing trajectory would be when $v_{in} = 0.250m/s$ and $h = 0.50r$ in terms of less total motion times.

Now there are still some issues that could be addressed in future iterations, such as to improve the beginning and ending phases of the trajectories, as they may perform some undesired behaviors. But considering that the obstacle avoidance trajectories are executed under exceptional and uncommon circumstances, one can consider as admissible these type performances, if they happen every once in a while.

This leads to the following point: up until now, everything has been done off-line with virtual obstacles. One possible following step would be to generate an extensive database of allowable trajectories, for different case scenarios, with obstacle at different points, and then implement a decision-making algorithm to choose the best trajectory, changing from one trajectory to another if required. This could be related work to the methodology of the rapidly-exploring random tree (RRT) algorithm based on random sampling.

Chapter 5

5. Conclusion and perspectives

The thesis project here presented deals with the study of collaborative robots for industrial purposes, with the aim to improve their model and control. This study was conducted considering two main research axes:

I. The friction modeling of Harmonic Drive gearboxes
II. The obstacle avoidance problem in collaborative robots

Regarding the Harmonic Drive study, we used an empirical approach for the obtention of the load-dependent friction laws with remarkable results. These laws were based on the Force Transfer Diagrams, which are a real practical way to describe the behavior and performances of gearboxes under different load conditions. This allowed me to understand better the internal mechanics of the gearboxes, which somehow match with the issues found in the literature review.

Nevertheless, based on prior results obtained in the thesis of (Hamon, P., 2011), where robotic joints are studied with similar approach, we are expecting something similar linear input-output relationships. Typically, a common robotic gearbox would present a Force Transfer Diagram with linear relationship of the input torque vs. output torque, but what we found in our research with the Harmonic Drive is that the input-output torque curves are parabolic. This can explain the non-linearities and undesired behaviors of the systems using Harmonic Drives previously seen in the literature. In this sense, the obtained Force Transfer curves can help at the improvement of the whole dynamic model of a given system, and by consequence the foreseen control.

The only limitation of the empirical method used is that it must be done for every joint of the robot, because two given gearboxes would not present the exact same parameters (although quite close behavior). This shall be done with a predefined routine and methodology, following certain specifications, to obtain the correspondent parameters that would define the load-dependent friction models for every joint.

Concerning the second major axe of this research, we successfully developed an offline trajectory generator to address the obstacle avoidance problem of collaborative robots. The arcs path planning computation, based on Dubins curve variation, resulted in a viable option to obtain paths with C^1 continuity.

The simulation framework and the synergy between MoveIt! and ROS frameworks, possess great advantages, as they facilitate the validation of desired trajectories, while avoiding collisions with predefined elements in the workspace. Robotic systems tend to get more complex as the requirements in hand increase. Therefore, having a predefined simulation ecosystem is a must if one wants to develop, to test, and to modify robotic tasks without affecting too much the production line scheduling or when simply the robots are unavailable. However, simulations usually differ from reality, which is why a thoroughly testing session with the real system is desired.

Through the different frameworks and methodologies here proposed, we can reduce the difficulty that new collaborative environments may present and easily generate new trajectories. As it has been stated before in Chapters 3 and 4, the trajectories are generated considering the elements present in the workspace, including the robot links of the robot itself, so that the resulting joint configurations guarantee collision-free motions.

Additionally, we successfully manage to overcome some of the limitations present in the initial experiment results from the simulated trajectories by effectively controlling the linear speed of the robot's end-effector. In that note, we are able to perform motions at the maximum optimal speed that the standard ISO/TS 15016:2016 stipules under collaboration conditions, guaranteeing minimal motion durations of the trajectories investigated.

Collaborative robots have been designed to push forward the next step into the future factory, by combining strength, stability, repeatability, and efficiency with the ability, flexibility, dexterity, and decision-making capacity of humans. However, it is evident that there is yet much to explore about the models and methodologies nowadays used in this field. This thesis manages to obtain interesting results and to study some important aspects towards the improvement of human-robot collaboration, but in reality, there is a vast domain of applications and

enhancements that could be interesting to discover and implement, to help advance the industrial production and the robotics community.

5.1. Perspectives

The study on the influence of friction in torque transmission should be integrated in the dynamic models of the robots. The first part of this manuscript focused on obtaining this influence of three different Harmonic Drive gearboxes, which resulted in having a particular parabolic behavior. However, the methodology only focused at quasi-static level. It would be of interest to do similar study with a dynamic system by adding a motor and an external torque sensor to verify the results under different loads and speeds. By grouping the contact friction and the inertia of the transmission, the transparency in force would be defined as the parabolic defect of the force-force transfer function of the mechanism as a function of the speed and acceleration parameters.

Initially it could be considered a single Harmonic Drive joint with a single link, later on, perhaps the conception of a two-link robot with Harmonic Drive joints could be of interest, as we would know exactly the specifications of each one of the components, without depending of robotic manufacturers on sharing their information.

Moreover, the objective of this, as stated before, is to improve the dynamic model of collaborative robots. If we are to study the torque transmission influence, it would be needed to be performed in robots whose joints could be controlled in torque (like KUKA LBR iiwa robots). We are unable to do this procedure with the Universal Robot model we had at our disposal. A more accurate model would allow a better collision detection (faster and safer), a better overall control of the robot when operating in hand-guiding mode, or simply an improved dynamic control when different payloads are used.

The trajectories proposed for obstacle avoidance can as well be further explored. Our approach was to generate and validate off-line trajectories, while reducing as much as possible time losses. More types of path planning algorithms could be proposed to have larger databases. Taking as a starting point our equations, some other variations could be explored, for example if the system realize that the obstacle is static and it has been for t_δ seconds, the new path could

start the curved evasion maneuver from pose A. There are also some other parameters, besides the sagitta h, that could be changed in order to obtain different trajectories.

The idea behind our off-line approach, is to be able to generate a database of plausible optimal trajectories, for different case scenarios. In order to do that, different trajectories must be explored as stated before, but also, the perception of the workspace by some sensors, like 3D cameras or LIDAR (Light Detection and Ranging) sensors for future decision making algorithms. Which leads us to explore the modeling of more complex obstacles and adapt the trajectories depending on the volume that is obstructing a given ideal trajectory.

Additionally, changing between trajectories given possible obstacles shall be explored. The system should be able to easily adapt and select the best suitable precomputed trajectory. This method could be explored with a mix of others proposed in the literature, for example with the potential field method or the Rapidly-Exploring Random Trees (RRT).

Finally, if the working cell is equipped with large number of sensors, it would be relevant to use this information in the control at a level of integration as close as possible to the actuators. This other strategy would make it possible to control the level of kinetic energy of the robot at the level of each joint and to define the best execution of the task without shock.

References

Abba, G., & Sardain, P. (2005). MODELING OF FRICTIONS IN THE TRANSMISSION ELEMENTS OF A ROBOT AXIS FOR ITS IDENTIFICATION. *IFAC Proceedings Volumes*, *38*(1), 7–12. https://doi.org/10.3182/20050703-6-CZ-1902.01272

Alami, R., Albu-Schaeffer, A., Bicchi, A., Bischoff, R., Chatila, R., De Luca, A., De Santis, A., Giralt, G., Guiochet, J., Hirzinger, G., Ingrand, F., Lippiello, V., Mattone, R., Powell, D., Sen, S., Siciliano, B., Tonietti, G., & Villani, L. (2006). Safe and dependable physical human-robot interaction in anthropic domains: State of the art and challenges. *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1–16. https://doi.org/10.1109/IROS.2006.6936985

Bayar, G. (2017). Reference Path Generation and Obstacle Avoidance for Autonomous Vehicles Based on Waypoints, Dubins Curves and Virtual Force Field Method. *International Journal of Applied Mathematics, Electronics and Computers*, *1*, 1–6. https://doi.org/10.18100/ijamec.2017527489

Blanchet, K. (2021). *Au coeur de l'interaction humain-robot collaboratif: Comment concevoir une assistance personnalisée au profil utilisateur ?* Institut Polytechnique de Paris.

Broquère, X. (2011). *Planification de trajectoire pour la manipulation d'objets et l'interaction Homme-robot*. Université de Toulouse.

Chaoui, M. D. (2020). *Contribution à la commande robuste d'un manipulateur utilisé en meulage* [Thesis]. Université de Lorraine.

Chen, J., Bi, S., Chen, L., & Xi, N. (2019). Event-based planning and control for active collision avoidance in human-robot collaboration. *2019 IEEE 9th Annual International Conference*

on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER), 1354–1357. https://doi.org/10.1109/CYBER46603.2019.9066674

Chen, J.-H., & Song, K.-T. (2018). Collision-Free Motion Planning for Human-Robot Collaborative Safety Under Cartesian Constraint. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 4348–4354. https://doi.org/10.1109/ICRA.2018.8460185

Chu, X., Hu, Q., & Zhang, J. (2018). Path Planning and Collision Avoidance for a Multi-Arm Space Maneuverable Robot. *IEEE Transactions on Aerospace and Electronic Systems*, *54*, 217–232. https://doi.org/10.1109/TAES.2017.2747938

Coleman, D., Sucan, I., Chitta, S., & Correll, N. (2014). Reducing the Barrier to Entry of Complex Robotic Software: A MoveIt! Case Study. *ArXiv:1404.3785 [Cs]*. http://arxiv.org/abs/1404.3785

Colgate, J. E., Wannasuphoprasit, W., & Peshkin, M. A. (1996). Cobots: Robots For Collaboration With Human Operators. *Proceedings of the 1996 ASME International Mechanical Engineering Congress and Exposition - Atlanta, GA, USA*, 433–439. http://www.scopus.com/inward/record.url?scp=0030402971&partnerID=8YFLogxK

Dahl, O., & Nielsen, L. (1990). Torque-limited path following by online trajectory time scaling. *IEEE Transactions on Robotics and Automation*, *6*(5), 554–561. https://doi.org/10.1109/70.62044

De Luca, A., & Flacco, F. (2012). Integrated control for pHRI: Collision avoidance, detection, reaction and collaboration. *2012 4th IEEE RAS EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, 288–295. https://doi.org/10.1109/BioRob.2012.6290917

De Santis, A., Siciliano, B., Luca, A., & Bicchi, A. (2008). An atlas of physical human-robot interaction. *Mechanism and Machine Theory*, *43*, 253–270. https://doi.org/10.1016/j.mechmachtheory.2007.03.003

Dubins, L. E. (1957). On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents. *American Journal of Mathematics*, *79*(3), 497–516. https://doi.org/10.2307/2372560

Ebert, D., Komuro, T., Namiki, A., & Ishikawa, M. (2005). Safe human-robot-coexistence: Emergency-stop using a high-speed vision-chip. *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2923–2928. https://doi.org/10.1109/IROS.2005.1545242

El Zaatari, S., Marei, M., Li, W., & Usman, Z. (2019). Cobot programming for collaborative industrial tasks: An overview. *Robotics and Autonomous Systems*, *116*, 162–180. https://doi.org/10.1016/j.robot.2019.03.003

fmauch. (2019). *Universal_Robots_ROS_Driver* [C++]. Universal Robots A/S. https://github.com/UniversalRobots/Universal_Robots_ROS_Driver (Original work published 2019)

Gandhi, P. S., Ghorbel, F. H., & Dabney, J. (2002). Modeling, identification, and compensation of friction in harmonic drives. *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.*, *1*, 160–166. https://doi.org/10.1109/CDC.2002.1184485

Garrec, P. (2011). *Telerobotics research and development at CEA-LIST*. ANS EPRRSD - 13th Robotics & Remote Systems for Hazardous Environments and 11th Emergency Preparedness & Response.

Garrec, P., Friconneau, J.-P., & Louveaux, F. (2004). *Virtuose 6D: A new force-control master arm using innovative ball-screw actuators*.

Ghorbel, F. H., Gandhi, P. S., & Alpeter, F. (2001). On the Kinematic Error in Harmonic Drive Gears. *Journal of Mechanical Design*, *123*(1), 90–97. https://doi.org/10.1115/1.1334379

Gill, M. A. C., & Zomaya, A. Y. (1998). A parallel collision-avoidance algorithm for robot manipulators. *IEEE Concurrency*, *6*(1), 68–78. https://doi.org/10.1109/4434.656781

Glagowski, T., Pedram, H., & Shamash, Y. (1992). Human factors in robot teach programming. In *Human-Robot Interaction* (pp. 16–47). CRC Press.

Goertz, R., & Bevilacqua, F. (1952). A FORCE-REFLECTING POSITIONAL SERVOMECHANISM. *Nucleonics, Vol 10, Part II*. https://www.semanticscholar.org/paper/A-FORCE-REFLECTING-POSITIONAL-SERVOMECHANISM-Goertz-Bevilacqua/9a6fb8e8a22cd6eca970049690a5e217b494a617

Grosz, B. J. (1996). Collaborative Systems (AAAI-94 Presidential Address). *AI Magazine*, *17*(2), 67–67. https://doi.org/10.1609/aimag.v17i2.1223

Haddadin, S., Albu-Schäffer, A., & Hirzinger, G. (2009). Requirements for Safe Robots: Measurements, Analysis and New Insights. *The International Journal of Robotics Research*, *28*(11–12), 1507–1527. https://doi.org/10.1177/0278364909343970

Hamon, P. (2011). *Dynamic modeling and identification of robots with a dry friction model depending on load and velocity*. Université de Nantes.

Hamon, P., Gautier, M., & Garrec, P. (2010). Dynamic identification of robots with a dry friction model depending on load and velocity. *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 6187–6193. https://doi.org/10.1109/IROS.2010.5649189

Han, B., Ma, J., & Li, H. (2016). Research on nonlinear friction compensation of harmonic drive in gimbal servo-system of DGCMG. *International Journal of Control, Automation and Systems*, *14*(3), 779–786. https://doi.org/10.1007/s12555-014-0430-8

Hauschild, J. P., Heppler, G. R., & McPhee, J. J. (2004). *Friction Compensation of Harmonic Drive Actuators*. 683–692.

IFR. (2019). *International Federation of Robotics*. https://ifr.org/

IFR. (2020, September). *IFR presents World Robotics Report 2020*. IFR International Federation of Robotics. https://ifr.org/ifr-press-releases/news/record-2.7-million-robots-work-in-factories-around-the-globe

ISO 10218-1. (2011). *ISO 10218-1:2011(en), Robots and robotic devices—Safety requirements for industrial robots—Part 1: Robots*. https://www.iso.org/obp/ui/#iso:std:iso:10218:-1:ed-2:v1:en

ISO/TS. (2016). *ISO/TS 15066:2016*. https://www.iso.org/standard/62996.html

Jeffreys, H., & Jeffreys, B. S. (1999). *Methods of mathematical physics* (3rd ed). Cambridge University Press.

Kammerer, N., & Garrec, P. (2013, May 23). *Dry friction modeling in dynamic identification for robot manipulators: Theory and experiments*. 2013 IEEE International Conference on Mechatronics (ICM), Vicenza, Italy. https://ieeexplore.ieee.org/document/6518574

Kavraki, L. E., Svestka, P., Latombe, J.-C., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, *12*(4), 566–580. https://doi.org/10.1109/70.508439

Khalil, W., & Dombre, E. (1999). *Modélisation identification et commande des robots*. Hermès Science Publications.

Khatib, O. (1985). Real-time obstacle avoidance for manipulators and mobile robots. *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, *2*, 500–505. https://doi.org/10.1109/ROBOT.1985.1087247

Kircanski, N. M., & Goldenberg, A. A. (1997). An Experimental Study of Nonlinear Stiffness, Hysteresis, and Friction Effects in Robot Joints with Harmonic Drives and Torque Sensors. *The International Journal of Robotics Research*, *16*(2), 214–239. https://doi.org/10.1177/027836499701600207

Kosuge, K., Yoshida, H., Taguchi, D., Fukuda, T., Hariki, K., Kanitani, K., & Sakai, M. (1994). Robot-human collaboration for new robotic applications. *Proceedings of IECON'94 - 20th Annual Conference of IEEE Industrial Electronics*, *2*, 713–718. https://doi.org/10.1109/IECON.1994.397872

Krüger, J., Lien, T. K., & Verl, A. (2009). Cooperation of human and machines in assembly lines. *CIRP Annals*, *58*(2), 628–646. https://doi.org/10.1016/j.cirp.2009.09.009

Kunz, T., Reiser, U., Stilman, M., & Verl, A. (2010). Real-time path planning for a robot arm in changing environments. *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 5906–5911. https://doi.org/10.1109/IROS.2010.5653275

Latombe, J.-C. (1991). *Robot motion planning* (2. print). Springer Science+Business Media.

LaValle, S. M., & Kuffner, J. J. (2001). Randomized Kinodynamic Planning. *The International Journal of Robotics Research*, *20*(5), 378–400. https://doi.org/10.1177/02783640122067453

Leven, P., & Hutchinson, S. (2002). A Framework for Real-time Path Planning in Changing Environments. *The International Journal of Robotics Research*, *21*(12), 999–1030. https://doi.org/10.1177/027836490202101200l

Liao, H., Fan, S., & Fan, D. (2016). Friction compensation of harmonic gear based on location relationship. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, *230*(8), 695–705. https://doi.org/10.1177/0959651816650083

Lin, Y., & Saripalli, S. (2014). Path planning using 3D Dubins Curve for Unmanned Aerial Vehicles. *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, 296–304. https://doi.org/10.1109/ICUAS.2014.6842268

Lozano-Perez, T. (1981). Automatic Planning of Manipulator Transfer Movements. *IEEE Transactions on Systems, Man, and Cybernetics*, *11*(10), 681–698. https://doi.org/10.1109/TSMC.1981.4308589

Lynch, K. M., & Park, F. C. (2017). *Modern robotics: Mechanics, planning, and control*. Cambridge University Press.

Ma, D., Yan, S., Yin, Z., & Yang, Y. (2018). Investigation of the friction behavior of harmonic drive gears at low speed operation. *2018 IEEE International Conference on Mechatronics and Automation (ICMA)*, 1382–1388. https://doi.org/10.1109/ICMA.2018.8484324

Malik, A. A., & Bilberg, A. (2019). Developing a reference model for human–robot interaction. *International Journal for Interactive Design and Manufacturing (IJIDeM)*. https://doi.org/10.1007/s12008-019-00591-6

Markel, H. (2011). The Origin Of The Word "Robot." *Science Friday*. https://www.sciencefriday.com/segments/the-origin-of-the-word-robot/

Matthias, B. (2015, March 11). *ISO/TS 15066—Collaborative Robots—Present Status*.

Meziane, R., Otis, M. J.-D., & Ezzaidi, H. (2017). Human-robot collaboration while sharing production activities in dynamic environment: SPADER system. *Robotics and Computer-Integrated Manufacturing*, *48*, 243–253. https://doi.org/10.1016/j.rcim.2017.04.010

Moreno-Valenzuela, J. (2006). Tracking control of on-line time-scaled trajectories for robot manipulators under constrained torques. *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, 19–24. https://doi.org/10.1109/ROBOT.2006.1641155

MoveIt. (2018). *Time Parameterization—Moveit_tutorials Noetic documentation*. https://ros-planning.github.io/moveit_tutorials/doc/time_parameterization/time_parameterization_tutorial.html#

Müsser, C. (1959). *Strain wave gearing* (Patent No. 2 906 143).

Nascimento, H., Mujica, M., & Benoussaad, M. (2021). Collision Avoidance Interaction Between Human and a Hidden Robot Based on Kinect and Robot Data Fusion. *IEEE Robotics and Automation Letters*, *6*(1), 88–94. https://doi.org/10.1109/LRA.2020.3032104

Niryo. (2018, January 11). *8 reasons why you should use ROS for robotics projects*. 8 Reasons Why You Should Use ROS for Robotics Projects. https://niryo.com/2018/01/8-reasons-use-ros-robotics-projects/

Ponce Quiroga, C. W., Garrec, P., Antoine, J.-F., Raharijaona, T., & Abba, G. (2021). Load-dependent Friction Laws of Three Models of Harmonic Drive Gearboxes Identified by Using a Force Transfer Diagram. *2021 12th International Conference on Mechanical and Aerospace Engineering (ICMAE)*, 239–244. https://doi.org/10.1109/ICMAE52228.2021.9522368

Popov, D., Klimchik, A., & Mavridis, N. (2017). Collision detection, localization amp; classification for industrial robots with joint torque sensors. *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 838–843. https://doi.org/10.1109/ROMAN.2017.8172400

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., & Ng, A. (2009). *ROS: An open-source Robot Operating System*. *3*.

Ravankar, A., Ravankar, A., Kobayashi, Y., Hoshino, Y., & Peng, C.-C. (2018). Path Smoothing Techniques in Robot Navigation: State-of-the-Art, Current and Future Challenges. *Sensors*, *18*(9), 3170. https://doi.org/10.3390/s18093170

Reeds, J., & Shepp, L. (1990). *OPTIMAL PATHS FOR A CAR THAT GOES BOTH FORWARDS AND BACKWARDS*. https://doi.org/10.2140/PJM.1990.145.367

ROS Wiki. (2014). *ROS/Concepts—ROS Wiki*. http://wiki.ros.org/ROS/Concepts

Rubio, F., Llopis-Albert, C., Valero, F., & Suñer, J. L. (2016). Industrial robot efficient trajectory generation without collision through the evolution of the optimal trajectory. *Robotics and Autonomous Systems*, *86*, 106–112. https://doi.org/10.1016/j.robot.2016.09.008

Saenz, J., Elkmann, N., Gibaru, O., & Neto, P. (2018). Survey of methods for design of collaborative robotics applications- Why safety is a barrier to more widespread robotics uptake. *Proceedings of the 2018 4th International Conference on Mechatronics and Robotics Engineering*, 95–101. https://doi.org/10.1145/3191477.3191507

Safeea, M., Béarée, R., & Neto, P. (2020). Collision Avoidance of Redundant Robotic Manipulators Using Newton's Method. *Journal of Intelligent & Robotic Systems*, *99*(3–4), 673–681. https://doi.org/10.1007/s10846-020-01159-3

Safeea, M., Neto, P., & Bearee, R. (2019). On-line collision avoidance for collaborative robot manipulators by adjusting off-line generated paths: An industrial use case. *Robotics and Autonomous Systems*, *119*, 278–288. https://doi.org/10.1016/j.robot.2019.07.013

Schmidt, B., & Wang, L. (2014). Depth camera based collision avoidance via active robot control. *Journal of Manufacturing Systems*, *33*(4), 711–718. https://doi.org/10.1016/j.jmsy.2014.04.004

Shi, Z., Li, Y., & Liu, G. (2017). Adaptive torque estimation of robot joint with harmonic drive transmission. *Mechanical Systems and Signal Processing*, *96*, 1–15. https://doi.org/10.1016/j.ymssp.2017.03.041

Singh, L., Stephanou, H., & Wen, J. (1996). Real-time robot motion control with circulatory fields. *Proceedings of IEEE International Conference on Robotics and Automation*, *3*, 2737–2742 vol.3. https://doi.org/10.1109/ROBOT.1996.506576

Spenlé, D., & Gourhant, R. (2007). Guide du calcul en mécanique. In *Hachette Technique*.

Staretu, I. (2021). Robotic Arms with Anthropomorphic Grippers for Robotic Technological Processes. *Proceedings*, *63*(1), 77. https://doi.org/10.3390/proceedings2020063077

Statista. (2022, January). *Collaborative robot market size worldwide 2020-2030*. Statista. https://www.statista.com/statistics/748234/global-market-size-collaborative-robots/

Steffensen, J. F. (2013). *Interpolation*. Dover Publications.

Tadayoni, A., Xie, W.-F., & Gordon, B. W. (2011). Adaptive control of harmonic drive with parameter varying friction using structurally dynamic wavelet network. *International Journal of Control, Automation and Systems*, *9*(1), 50–59. https://doi.org/10.1007/s12555-011-0107-5

Tan, J. T. C., Duan, F., Zhang, Y., Watanabe, K., Kato, R., & Arai, T. (2009). Human-robot collaboration in cellular manufacturing: Design and development. *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 29–34. https://doi.org/10.1109/IROS.2009.5354155

Tuttle, T. D., & Seering, W. P. (1996). A nonlinear model of a harmonic drive gear transmission. *IEEE Transactions on Robotics and Automation*, *12*(3), 368–374. https://doi.org/10.1109/70.499819

UR. (n.d.-a). *About Universal Robots [Our History]*. Retrieved March 2, 2022, from https://www.universal-robots.com/about-universal-robots/our-history/

UR. (n.d.-b). *UR10e Collaborative industrial robot—Payload up to 12.5 kg*. Retrieved January 10, 2022, from https://www.universal-robots.com/products/ur10-robot/

Vertut, J., & Coiffet, P. (1984). *Les Robots. Tome 3A, Téléopération. Évolution des technologies* (Vol. 3A). Hermes publishing France.

Wannasuphoprasit, W., Akella, P., Peshkin, M., & Colgate, J. E. (1998). Cobots: Proceedings of the 1998 ASME International Mechanical Engineering Congress and Exposition. *American Society of Mechanical Engineers (Paper)*, 1–7.

Waring, E. (1779). Problems concerning Interpolations. By Edward Waring, M. D. F. R. S. and of the Institute of Bononia, Lucasian Professor of Mathematics in the University of Cambridge. *Philosophical Transactions of the Royal Society of London*, *69*, 59–67.

Weisstein, E. W. (n.d.). *Hermite's Interpolating Polynomial* [Text]. Wolfram Research, Inc. Retrieved February 21, 2022, from https://mathworld.wolfram.com/

Yamamoto, M., Iwasaki, M., Hirai, H., Okitsu, Y., Sasaki, K., & Yajima, T. (2009). Modeling and compensation for angular transmission error in harmonic drive gearings. *IEEJ*

Transactions on Electrical and Electronic Engineering, *4*(2), 158–165. https://doi.org/10.1002/tee.20393

Yang, D., Li, D., & Sun, H. (2013). 2D Dubins Path in Environments with Obstacle. *Mathematical Problems in Engineering*, *2013*, 1–6. https://doi.org/10.1155/2013/291372

Yang, K., Moon, S., Yoo, S., Kang, J., Doh, N., Kim, H., & Joo, S. (2014). Spline-Based RRT Path Planner for Non-Holonomic Robots. *Journal of Intelligent and Robotic Systems*, *73*, 763–782. https://doi.org/10.1007/s10846-013-9963-y

Zhang, W., & Sobh, T. M. (2003). Obstacle Avoidance for Manipulators. *Systems Analysis Modelling Simulation*, *43*(1), 67–74. https://doi.org/10.1080/0232929031000116344

Zhao, M., & Lv, X. (2020). Improved Manipulator Obstacle Avoidance Path Planning Based on Potential Field Method. *Journal of Robotics*, *2020*, 1–12. https://doi.org/10.1155/2020/1701943

Zhao, R., & Sidobre, D. (2015). *Dynamic Obstacle Avoidance using Online Trajectory Time-Scaling and Local Replanning*. 414–421.

Zou, C., Tao, T., Jiang, G., Mei, X., & Wu, J. (2017). A harmonic drive model considering geometry and internal interaction. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, *231*(4), 728–743. https://doi.org/10.1177/0954406215621097