Politecnico di Torino

POLITECNICO DI TORINO

Master Degree course in Mechatronics engineering

A.y. 2022/2023

Graduation session July 2023

Modeling, identification and control of an omnidirectional wheeled manipulator for intralogistics applications in shared workspaces

Candidate

Alessandro De Toni

Supervisors

Prof. Alessandro Rizzo
M.Sc. Mazin Hamad

Acknowledgements

This thesis required a huge effort but drove me to understand more about myself and what I want to do in life, especially renovating my interest in the research field. I would like to thank my host university supervisor, Mazin Hamad, for giving me this opportunity. He greeted me in a different country with a warm welcome and has always provided support with a positive and assertive attitude. Then I would like to express my gratitude to my supervisor Alessandro Rizzo for his professionalism and willingness. A special thank goes to Vasilije Rakcevic and Alessandro Melone, with whom I shared most of my time in the office and I will always remember in a pleasant way. I would also like to thank my family, my relatives and my friends, for constituting a safe and happy spot where I can enjoy my free time and to which I owe everything for the help they don't hesitate to share. Ultimately, my biggest gratitude goes to my girlfriend Maria Grazia, for her patience, her kindness and for being a constant presence with whom I've shared the happiest moments of my life.

1 INTRODUCTION

1.1 Use Case and Challenges

Efficient material flow, also known as intralogistics, plays a crucial role in agile production. The automatization of the internal supply chain processes is a rising trend in the nowadays industry, especially with the introduction of Automated Guided Vehicles (AGVs). However, despite the potential of such robotic solutions, their complete integration into automating intralogistics processes, particularly in collaboration with humans, remains a significant challenge. This is primarily due to the limitations in flexibility, cost-effectiveness, and safety certifications, with the latter being the biggest obstacle. In fact, currently, the available solutions are often based on the assumption that navigation and manipulation tasks are performed separately, this severely simplifies the achievement of certification but hugely impacts the performances of the robot.

1.2 Proposed Solution

To address these challenges, this thesis focuses on the development of a model-based, whole-body controller for an omnidirectional-wheeled manipulator that can handle navigation and manipulation tasks simultaneously while ensuring human safety. To achieve efficient execution of manipulation tasks, it is essential to employ rigid or compliant actions. Therefore, the whole-body controller has been specifically designed with a strong emphasis on dynamics, leveraging the theory of impedance control. By adopting a model-based approach instead of alternatives like optimization-based control, we can ensure safety and reliability, mitigating the risk of failure and improving our understanding of the overall system's functioning. Safety is addressed by the control framework at the trajectory planning level by employing a validated algorithm that incorporates impact dynamics and injury biomechanics. Eventually, the required flexibility for agile production is attained through the exploitation of the structural redundancy to tackle coordinated arm-base motions. Particularly important for this final aspect is the consideration of the couplings that arise between the two subsystems that compose the mobile manipulator, a part often neglected on behalf of the hypothesis that these two will be subject to only sequential and non-simultaneous motions.

1.3 Thesis Structure

In chapter 2 the theoretical background needed to understand the proposed control framework is provided to the reader. This comprises a brief introduction to robotics, a detailed description of omnidirectional-wheeled mobile manipulators' kinematics and dynamics for the rigid case, a simple experimental method for deriving the center of mass of the robot, an explanation of impedance control, including the choice of its characteristic parameters, a way of resolving redundancy by means of artificial potential fields and null-space projections, and, eventually, an approach to hinder unsafe motions.

Subsequently in chapter 3, the proposed solution is discussed in detail by actualizing and joining the theoretical foundations introduced in chapter 2. Here a description of *RB-KAIROS+* is also provided as this was the robot utilized as an example of application.

Chapter 4 and 5 show the rigorous testing performed in MATLAB/Simulink, the developed simulation in ROS/Gazebo and also discuss the results that were obtained.

Ultimately, chapter 6 draws the conclusions of this work and adds some interesting ideas on the possible future developments.

1.4 A clarification on the notation in use

A brief discussion on the notation in use is needed to ensure an easier understanding for the reader.

Scalars, vectors and matrices In order to distinguish these 3 objects, it is made use of different notations:

- a or K - *scalar*: non-bold, both lower case and upper case letters will be used.
- $\boldsymbol{a}$ - *vector*: bold, lower case.
- $\boldsymbol{A}$ - *matrix*: bold, upper case.

A notable exception is the notation of the twist vector $\mathcal{V}$.

Coordinates and frames of reference The notation $\{A\} = (x, y, z)$ is exploited to define a right-handed 3D frame of reference with axes x, y, z. The same notation will be extended to 2D framers.

In order to state that a vector or a matrix is written in frame $\{A\}$, the vector or the matrix will be preceded by a superscript ^{A}, e.g. ${}^{A}\boldsymbol{p}$. It is true that in this way the writing is heavier but the resulting clarity makes it worth.

Cross product through matrix multiplication An alternative way of writing the cross product is by means of skew-symmetric matrices.

$$\boldsymbol{a} \times \boldsymbol{b} = S(\boldsymbol{a})\boldsymbol{b} = S(\boldsymbol{b})^T\boldsymbol{a}$$

where

- $S(\boldsymbol{a})$ - *Skew-simmetric matrix corresponding to vector* $\boldsymbol{a}$: $S(\boldsymbol{a}) = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}$

2 STATE OF THE ART

2.1 An introduction to robotics

Robotics aims to design machines that can ease human life. In order to fulfil this objective, robots need to be able to perform tasks and this, many times, means executing a motion in a controlled and smart way.

To execute a motion, it is first needed to describe it. The initial step to do so is to define the location of the robot. Since robots can be quite complex systems formed by several rigid bodies, one of these rigid bodies will be selected (usually, in the case of manipulators, this coincides with the end-effector) and its location in space will be utilized to depict the location of the whole robot. It is very well known that a rigid body in Cartesian space has 6 degrees of freedom (DOF), this is why robot location is usually depicted through a 6×1 vector $\boldsymbol{x}$ called *pose* in which all the information of position and orientation is embedded.

The location of the selected rigid body will depend on the movements of the other rigid bodies that compose the robot. In order to describe their state, a vector $\boldsymbol{q} \in \mathbb{R}^n$, with $n =$ the number of rigid bodies, is introduced and named *configuration space coordinates*[1] or *generalized coordinates* or even *joint space coordinates*.

As a consequence we can state that $\boldsymbol{x}$ is a function of the configuration space variables $\boldsymbol{q}$ and their relation is described by the *direct kinematics equation* [1],

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{p} \\ \boldsymbol{\phi} \end{bmatrix} = \boldsymbol{f}(\boldsymbol{q}) \tag{2.1}$$

where $\boldsymbol{p}$ is used to denote the position and ϕ indicates the orientation [2].

Now, a motion, or a trajectory, is always linked to time, and it actually can be defined as the evolution in time of the pose of the robot, hence,

$$\boldsymbol{x}(t) = \boldsymbol{f}(\boldsymbol{q}(t)) \tag{2.2}$$

It is of great interest to understand how fast the trajectory changes over time, this is done by analysing the twist of the robot, which is also called *task space velocity* $\dot{\boldsymbol{x}}(t)$. Again, this is linked to the rate of change of the configuration variables, e.g. the *configuration velocities* $\dot{\boldsymbol{q}}(t)$. The relation that allows retrieving the task velocity $\dot{\boldsymbol{x}}(t)$ from $\dot{\boldsymbol{q}}$ consists in

[1] As intuitively understandable, the name comes from the fact that it describes the configuration of the robot

[2] Position and orientation can be described through several different representations, a brief discussion on representation choice can be found in 2.1.1

$$\dot{\boldsymbol{x}}(t) = \begin{bmatrix} \dot{\boldsymbol{p}} \\ \dot{\boldsymbol{\phi}} \end{bmatrix} = \frac{d\boldsymbol{f}(\boldsymbol{q}(t))}{dt} = \frac{d\boldsymbol{f}(\boldsymbol{q})}{d\boldsymbol{q}} \frac{d\boldsymbol{q}(t)}{dt} = \boldsymbol{J}_A(\boldsymbol{q}(t))\dot{\boldsymbol{q}}(t) \tag{2.3}$$

Where $\boldsymbol{J}_A$ is the *analytical* Jacobian and depends on the representation chosen for $\boldsymbol{x}(t)$. A clarification on this aspect will be provided in 2.1.1.

There is also another way to define the Jacobian which is representation-independent [2] and is retrieved by studying the geometry of the robot. We will refer to this Jacobian as *geometric Jacobian* or *body Jacobian* $\boldsymbol{J}_b$. Practically, the difference between the two is made plain by,

$$\mathcal{V}(t) = \begin{bmatrix} \dot{\boldsymbol{p}}(t) \\ \boldsymbol{\omega}(t) \end{bmatrix} = \boldsymbol{J}_b(\boldsymbol{q}(t))\dot{\boldsymbol{q}}(t) \tag{2.4}$$

where $\dot{\boldsymbol{p}}(t)$ is the linear velocity of the rigid body, while $\boldsymbol{\omega}(t)$ is its angular velocity.

From now on the variable t will be often omitted to simplify the notation and when the discussion is valid for both the Jacobians, a generic $\boldsymbol{J}$ will be adopted.

Lastly, accelerations can also be retrieved by simply differentiating 2.3,

$$\ddot{\boldsymbol{x}}(t) = \dot{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{q}} + \boldsymbol{J}(\boldsymbol{q})\ddot{\boldsymbol{q}} \tag{2.5}$$

In the case in which it is desired to obtain $\dot{\boldsymbol{q}}$ from $\dot{\boldsymbol{x}}$, we talk about the *inverse kinematic problem* [1] that if $\boldsymbol{J}$ is invertible can be simply solved as

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}(\boldsymbol{q})\dot{\boldsymbol{x}} \tag{2.6}$$

If the robot is *redundant*, meaning that $n > 6$, $\boldsymbol{J}$ will not be a square matrix and, thus, not invertible. In this case, a *pseudo-inverse* can be adopted to solve the problem. As a solution to this problem, in section 2.3.2, a *dynamically consistent pseudo-inverse* is presented.

2.1.1 Task space representation

Nowadays, to represent robot kinematics in task space there are several and very well-known methods. To describe a position, except for special cases, Cartesian coordinates (x, y, z) appear to be the simplest, yet effective representation. Coming to the orientation, instead, the choice is not that straightforward.

In principle, a rotation matrix $\boldsymbol{R}$ could be directly adopted, and the direct kinematics problem would be solved by means of the *homogenous transformation matrix* $\boldsymbol{T}(\boldsymbol{q})$,

$$\boldsymbol{T}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{R}(\boldsymbol{q}) & \boldsymbol{p}(\boldsymbol{q}) \\ \boldsymbol{0}_{3\times3} & 1 \end{bmatrix} \tag{2.7}$$

However, this solution is often avoided because of the high number of parameters needed,

due to its difficult integration in control loops and due to its counter-intuitive description of the orientation.

Much simpler and very similar to Cartesian coordinates in their clarity and compactness is the use of Euler angles. Euler angles constitute a *minimal representation*[3] [1] and are easy to integrate in control, nevertheless they are prone to mathematical singularities and they determine an impedance behaviour that is representation dependant [3].

Consequently, this led researchers to find a different representation not subject to the aforementioned problems that will be introduced in the next paragraph.

Unit quaternions representation A unit quaternion is a non-minimal representation of the orientation defined as: [3]

$$\mathbf{q} = \{\eta, \boldsymbol{\epsilon}\} = \eta + \epsilon_1 i + \epsilon_2 j + \epsilon_3 k, \quad \boldsymbol{\epsilon} \in \mathcal{R}^3, \tag{2.8}$$

$$||\mathbf{q}||^2 = \eta^2 + \boldsymbol{\epsilon}^T \boldsymbol{\epsilon} = 1 \tag{2.9}$$

A more meaningful definition from a physical perspective is the subsequent:

$$\mathbf{q} = \{cos(\theta/2), \boldsymbol{r} sin(\theta/2)\} \tag{2.10}$$

where $\boldsymbol{r}$ is the axis around which a rotation θ is occurring.

The definition as a special case of angle axis makes it eligible to the application of the Rodrigues' formula which leads to the two-to-one[4] relation with rotation matrices[3]:

$${}^0_1\boldsymbol{R} = 2\boldsymbol{\epsilon}_{10}\boldsymbol{\epsilon}_{10}^{\mathrm{T}} + (2\eta_{10}^2 - 1)\boldsymbol{I} + 2\eta_{10}\boldsymbol{S}(\boldsymbol{\epsilon}_{10}) \tag{2.11}$$

Even more important from a practical standpoint is the inverse relation:

$$\eta = \frac{1}{2}\sqrt{R_{11} + R_{22} + R_{33} + 1} \tag{2.12}$$

$$\boldsymbol{\epsilon} = \frac{1}{2}\begin{bmatrix} \mathrm{sgn}(R_{32} - R_{23})\sqrt{R_{11} - R_{22} - R_{33} + 1} \\ \mathrm{sgn}(R_{13} - R_{31})\sqrt{R_{22} - R_{33} - R_{11} + 1} \\ \mathrm{sgn}(R_{21} - R_{12})\sqrt{R_{33} - R_{11} - R_{22} + 1} \end{bmatrix} \tag{2.13}$$

where η has been chosen to be ≥ 0

Representing a pose with quaternions Eventually, quaternions can be employed to represent the pose of a rigid body in this way,

[3] "Minimal" because at least 3 parameters are needed to describe an orientation in three-dimensional space

[4] $\{\eta, \boldsymbol{\epsilon}\}$ and $\{-\eta, -\boldsymbol{\epsilon}\}$ lead to the same rotation

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{p} \\ \mathbf{q} \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ \eta \\ \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{bmatrix} \tag{2.14}$$

and its rate of change as,

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{p}} \\ \dot{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\eta} \\ \dot{\epsilon}_1 \\ \dot{\epsilon}_2 \\ \dot{\epsilon}_3 \end{bmatrix} \tag{2.15}$$

The derivative of the quaternion can be obtained through the so-called *quaternion propagation* formula [3]:

$$\dot{\eta} = -\frac{1}{2}\boldsymbol{\epsilon}^{\mathrm{T}}\boldsymbol{\omega} \tag{2.16}$$

$$\dot{\boldsymbol{\epsilon}} = \frac{1}{2}(\eta\boldsymbol{I} - S(\boldsymbol{\epsilon}))\boldsymbol{\omega} \tag{2.17}$$

where $\boldsymbol{\omega}$ is the angular velocity of the rigid body.

Recalling 2.3, 2.4 and applying 2.16, 2.17, it is easy to prove that:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{p}} \\ \dot{\mathbf{q}} \end{bmatrix} = \boldsymbol{J}_A(\boldsymbol{q})\dot{\boldsymbol{q}} = \begin{bmatrix} \boldsymbol{I}_{3\times3} & \mathbf{0}_{4\times3} \\ \mathbf{0}_{3\times3} & \boldsymbol{E}(\mathbf{q}) \end{bmatrix} \boldsymbol{J}_b\dot{\boldsymbol{q}} \tag{2.18}$$

where $\boldsymbol{E}(\mathbf{q}) = \frac{1}{2}\begin{bmatrix} -\boldsymbol{\epsilon}^T \\ (\eta\boldsymbol{I} - S(\boldsymbol{\epsilon}))\boldsymbol{\omega} \end{bmatrix}$

A 7 element $\boldsymbol{x}, \dot{\boldsymbol{x}}, \ddot{\boldsymbol{x}}$ might be difficult to tackle, especially inside control laws, luckily, via a simple trick it is possible to trace back to a 6×1 vector as explained in the next paragraph.

A simplified orientation error via quaternions' representation For control, it is interesting to investigate how to represent the error between the current orientation and the desired one.

Assuming that ${}^S_E\boldsymbol{R}$ and ${}^S_D\boldsymbol{R}$ are the matrices that describe the current and the desired orientations of the end effector in the space frame S, from eq. 2.11 it is evident that we can

describe the same orientations via the quaternions $\mathfrak{q}_{es} = \{\eta_{es}, \boldsymbol{\epsilon}_{es}\}$ and $\mathfrak{q}_{ds} = \{\eta_{ds}, \boldsymbol{\epsilon}_{ds}\}$.

The orientation error[5] is characterized by a rotation that brings the end-effector from a frame to another, therefore, in this case, this coincides with ${}^D_E\boldsymbol{R} = {}^S_D\boldsymbol{R}^{-1}{}^S_E\boldsymbol{R}$. Analogously to what has just been done, we can associate this rotation with a quaternion that we will call *quaternion error*

$$ {}^D\mathfrak{q}_{ed} = {}^D\mathfrak{q}_{ds}^{-1} * {}^D\mathfrak{q}_{es} \tag{2.19} $$

where $\mathfrak{q}_{ds}^{-1} = \{\eta_{ds}, -\boldsymbol{\epsilon}_{ds}\}$ is the quaternion corresponding to the rotation ${}^S_D\boldsymbol{R}^{-1}$. Please notice that everything is reported to the desired frame {D} this, of course, is not mandatory for defining the orientation error but can simplify the definition of the stiffness matrix [4] of the impedance controller.

Actually, the most important information about orientation is embedded in the imaginary part of ${}^D\mathfrak{q}_{ed}$ as Natale and Luh, Walker, and Paul state in their works [3][5]. This leads to the first simplification that allows to define the orientation error $\boldsymbol{e}_O$ as a 3×1 vector,

$$ \boldsymbol{e}_O = {}^D\boldsymbol{r}_{ed} sin(\theta_{ed}/2) = {}^D\boldsymbol{\epsilon}_{ed} \tag{2.20} $$

Now, one could think to directly derive the rate of change of this error by simply writing $\dot{\boldsymbol{e}}_O = {}^D\dot{\boldsymbol{\epsilon}}_{ed}$, however, this is an unnecessary complication that would only increase the computational burden and lead to the employment of an elaborate analytical Jacobian as showed in 2.18.

This can be avoided by applying a small approximation as explained in the subsequent part.

As depicted in [5], consider a sufficiently small time interval $(t - t_j)$, such that the relative angular velocity between $\{D\}$ and $\{E\}$ can be regarded as constant and equal to $\boldsymbol{\omega}_{ed}$. Consequently, this rotation can be represented through an angle and an axis as follows:

$$ \boldsymbol{r}_{ed}(t_j) = \frac{\boldsymbol{\omega}_{ed}}{||\boldsymbol{\omega}_{ed}||} \quad \text{and} \quad \theta = ||\boldsymbol{\omega}_{ed}||(t - t_j), \quad t_j \leq t \leq t_{j+1} $$

recalling 2.20, $\boldsymbol{e}_O$ can be rewritten as,

$$ \boldsymbol{e}_O = \frac{\boldsymbol{\omega}_{ed}}{||\boldsymbol{\omega}_{ed}||} sin(||\boldsymbol{\omega}_{ed}||(t - t_j)/2) $$

Whereas its derivative is

$$ \dot{\boldsymbol{e}}_O = \frac{1}{2}\boldsymbol{\omega}_{ed} cos(||\boldsymbol{\omega}_{ed}||(t - t_j)/2) \simeq \frac{1}{2}\boldsymbol{\omega}_{ed} = \frac{1}{2}(\boldsymbol{\omega}_{es} - \boldsymbol{\omega}_{ds}) $$

In order to avoid the factor $\frac{1}{2}$, a factor 2 is added in the expression of $\boldsymbol{e}_O$.

From the possibility of writing $\dot{\boldsymbol{e}}_O$ as a simple difference between angular velocity, it follows that the introduction of the analytical Jacobian is useless and the body Jacobian can be directly exploited.

[5] error is defined in Ott fashion [4] $\boldsymbol{e} = \boldsymbol{x}_e - \boldsymbol{x}_d$ to make it coherent with the other parts of this thesis

2.2 Kinematics of omnidirectional mobile manipulator

Kinematics is the branch of mechanics that aims to describe motion without considering its causes.

In this section, the direct kinematics and the derivation of the geometric Jacobian of an omnidirectional vehicle equipped with mecanum wheels will be presented and the same will be done for a serial-link manipulator. Eventually, these two models will be joined together to compose the kinematics of a mobile manipulator.

2.2.1 Kinematics of a manipulator

The most interesting part of the motion of a manipulator is constituted by the end-effector, which is the ultimate part of the robot in which, usually, a tool is applied.

Manipulators are constituted of multiple links attached through joints. Each link will contribute to the motion of the end-effector and modelling this contribution through generic geometrical intuition can be complex, thus, to help in this regard a convention has been developed.

The Denavit-Hartenberg convention defines a set of simple rules to attach frames to the joints of the manipulator thus allowing to shrink the number of parameters, used to describe the kinematic relationship between two adjacent links i and $i-i$, to just 4. These constitute the so-called *D-H parameters* $(\alpha_i, a_i, \theta_i, d_i)$ [2]. A detailed description on how to apply this set of rules can be found in [2] for the standard D-H convention and in [6] for the modified D-H convention.

Here will be reported only the final result obtained using the modified D-H parameters, that is, the homogenous transformation matrix that relates link $i-i$ to link i:

$$
{}^{i-1}_{i}\boldsymbol{T} = \left[\begin{array}{ccc|c}
\cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\
\sin\theta_i\cos\alpha_{i-1} & \cos\theta_i\cos\alpha_{i-1} & -\sin\alpha_{i-1} & -d_i\sin\alpha_{i-1} \\
\sin\theta_i\sin\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & \cos\alpha_{i-1} & d_i\cos\alpha_{i-1} \\
\hline
0 & 0 & 0 & 1
\end{array}\right] \tag{2.21}
$$

If the joint to which the link is attached allows only translational motions (*prismatic* joint), only the parameter d_i will be a variable while all the others will depend on the geometry of the robot and hence, will be fixed. On the other hand, a joint that allows only a rotational motion (*revolute* joint) implies that θ_i is the only parameter that varies.

Putting together all of the joint variables the set of configuration space variables $\boldsymbol{q}_a$ is obtained.

By means of $\boldsymbol{q}_a$ it is possible to determine each transformation matrix ${}^{i-1}_{i}\boldsymbol{T}$ and starting from the first joint every matrix is multiplied for the one corresponding to the subsequent pair resulting in a kinematic chain that describes the pose of the end-effector as a function of the joint variables.

$$
{}^{0}_{n}\boldsymbol{T}(\boldsymbol{q}_a) = {}^{0}_{1}\boldsymbol{T}(q_{a_1}){}^{1}_{2}\boldsymbol{T}(q_{a_2})\ldots{}^{n-1}_{n}\boldsymbol{T}(q_{a_n}) \tag{2.22}
$$

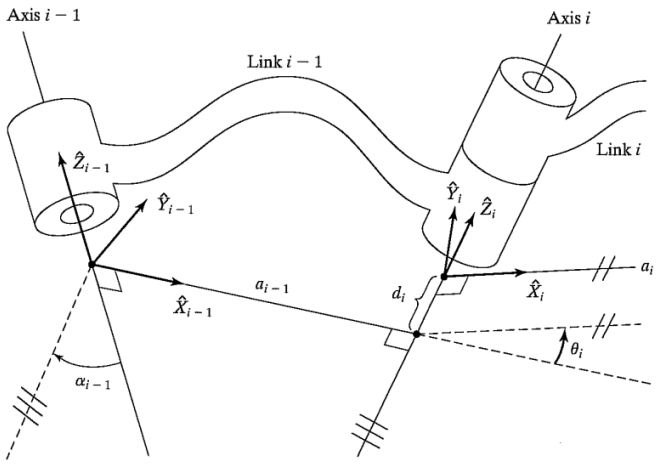


Figure 2.1: Link frames in modified D-H convention. Image taken from [6]

Jacobian of the manipulator From the geometrical description of the arm, it is possible to derive the cause-effect relation between the end-effector velocities, which belong to the operational space, and the joint variables rate of change $\dot{\boldsymbol{q}}_a$ [2]. Each joint contribution is given by [6]:

$$ {}^0\boldsymbol{J}^i(q_{a_i}) = \begin{bmatrix} {}^0\boldsymbol{z}_i \times ({}^0\boldsymbol{p}_e - {}^0\boldsymbol{p}_i) \\ {}^0\boldsymbol{z}_i \end{bmatrix} \tag{2.23} $$

where ${}^0\boldsymbol{z}_i$ is the z-axis of link frame i, ${}^0\boldsymbol{p}_i$ is the position of the link frame i origin and ${}^0\boldsymbol{p}_e$ is the position of the end-effector frame (refer to figure 2.1 for a complete understanding).

Stacking up all of the links' ${}^0\boldsymbol{J}^i$ the *geometric Jacobian* (also called *body Jacobian*) ${}^0\boldsymbol{J}_a(\boldsymbol{q}_a)$ of the manipulator is formed. Therefore a simple way to retrieve the end-effector twist is given by

$$ {}^0\mathcal{V} = {}^0\boldsymbol{J}_a(\boldsymbol{q}_a)\dot{\boldsymbol{q}}_a \tag{2.24} $$

2.2.2 Kinematics of an omnidirectional vehicle

Omnidirectional vehicles are, by definition, vehicles that can move in any direction without being subject to any constraint. Depending on the type of wheel adopted, different considerations can be done while attempting the modelling of their kinematics. In this case, a model that assumes the presence of 4 mecanum wheels will be provided.

The mechanical design of a mechanum wheel allows these two directions of motion [7]:

- *Driving* - movement along the direction perpendicular to the axis of rotation.
- *Sliding* - movement along a line characterized by a 45° inclination with respect to the axis of rotation.

[6] For modified D-H convention

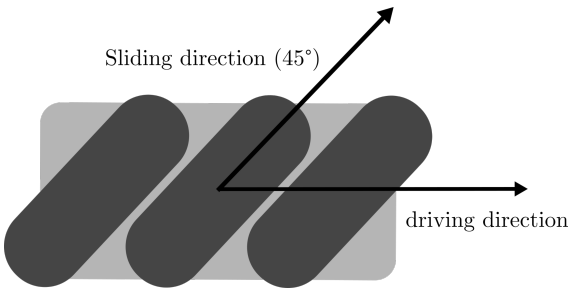


Figure 2.2: Simplified schema of the mechanum wheel. Figure adapted from [7]

It is clear that the most general motion will be the result of a combination of a movement along both *driving* and *sliding* directions [7]:

$$ {}^{W}\boldsymbol{v} = \begin{bmatrix} {}^{W}v_x \\ {}^{W}v_y \end{bmatrix} = v_{drive} \begin{bmatrix} 1 \\ 0 \end{bmatrix} + v_{slide} \begin{bmatrix} -sin(\frac{\pi}{4}) \\ cos(\frac{\pi}{4}) \end{bmatrix} \tag{2.25} $$

where W is a frame of reference centered on the wheel's geometrical center.

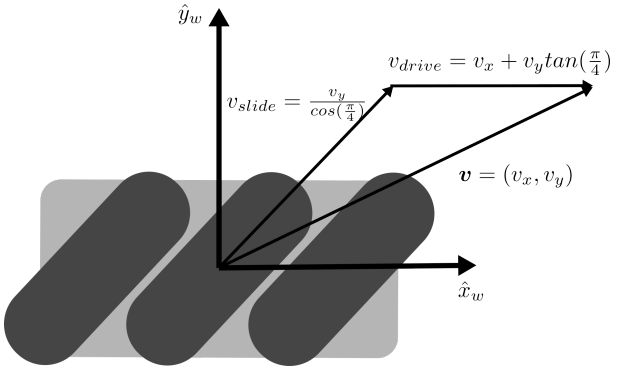


Figure 2.3: Wheel motion capabilities. Figure adapted from [7]

Dividing v_{drive} by the wheel radius r, the angular velocity ω is obtained,

$$ \omega = \frac{v_{drive}}{r} = \frac{1}{r}(v_x + v_y tan\gamma) \tag{2.26} $$

Defining a body frame $\{V\} = (x_v, y_v)$, centered in the geometrical center of the mobile platform and rotated by an angle ϕ with respect to a fixed spatial frame $\{S\} = (x, y)$, it is possible to obtain the existing relationship between the velocities of the vehicle ${}^{S}\mathcal{V}_v = (\dot{x}_v, \dot{y}_v, \dot{\phi}_v)$[7] and the wheels' angular velocity ω_i as [7]:

[7] where "v" subscript is used to differentiate the vehicle configuration coordinates from the arm configuration coordinates that will be introduced later

$$\omega_i = \boldsymbol{h}_i(\phi)\dot{\boldsymbol{q}}_{\boldsymbol{v}} = \begin{bmatrix} \frac{1}{r_i} & \frac{tan(\frac{\pi}{4})}{r} \end{bmatrix} \begin{bmatrix} 1 & 0 & -y_{w,i} \\ 0 & 1 & x_{w,i} \end{bmatrix} \begin{bmatrix} cos\phi_v & sin\phi_v & 0 \\ -sin\phi_v & cos\phi_v & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x}_v \\ \dot{y}_v \\ \dot{\phi}_v \end{bmatrix} \tag{2.27}$$

where the vector $\boldsymbol{h}_i(\phi_v)$ is:

$$\boldsymbol{h}_i(\phi_v) = \frac{1}{r_i cos\gamma_i} \begin{bmatrix} cos(\gamma_i + \phi_v) \\ sin(\gamma_i + \phi_v) \\ x_{w,i}sin(\gamma_i) - y_{w,i}cos(\gamma_i) \end{bmatrix}^T \tag{2.28}$$

Stacking up in columns each wheel $\boldsymbol{h}_i$ in a single matrix $\boldsymbol{H}(\phi_v) \in \mathbb{R}^{4\times3}$, whose inverse embeds the information about the contribution of each wheel to the vehicle velocity, we get [7]

$$\boldsymbol{H}(\phi_v) = \frac{1}{r} \begin{bmatrix} cos(\phi_v) + sin(\phi_v) & -cos(\phi_v) + sin(\phi_v) & -l - w \\ cos(\phi_v) - sin(\phi_v) & cos(\phi_v) + sin(\phi_v) & l + w \\ cos(\phi_v) + sin(\phi_v) & -cos(\phi_v) + sin(\phi_v) & l + w \\ cos(\phi_v) - sin(\phi_v) & cos(\phi_v) + sin(\phi_v) & -l - w \end{bmatrix} \tag{2.29}$$

where it has been considered that each wheel is placed at a distance $|x_{w,i}| = l$ and $|y_{w,i}| = w$ from the geometrical center of the vehicle.

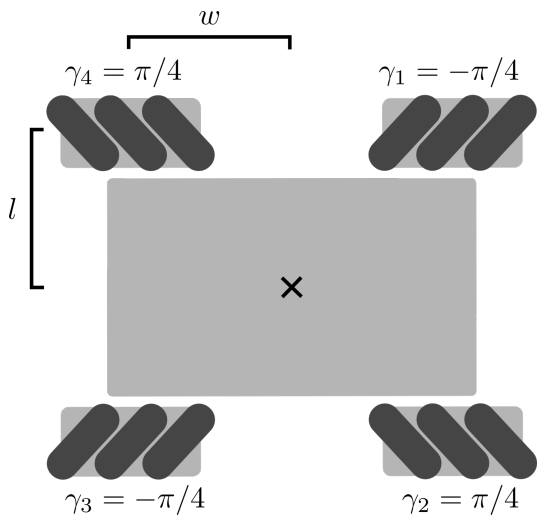


Figure 2.4: Wheel position representation. Adapted from [7]

Writing the same model w.r.t. the body frame V instead of S, $\boldsymbol{H}$ becomes independent from ϕ_v [7]:

$$\boldsymbol{\omega} = \boldsymbol{H}(0)^{V}\mathcal{V}_v = \begin{bmatrix} 1 & -1 & -l-w \\ 1 & 1 & l+w \\ 1 & -1 & l+w \\ 1 & 1 & -l-w \end{bmatrix} \begin{bmatrix} v_{vx} \\ v_{vy} \\ \omega_{vz} \end{bmatrix} \tag{2.30}$$

From this expression, it is very intuitive to state that

- a movement in $\hat{x}_b$ direction is achieved by actuating all the wheels at the same speed.
- a movement in $\hat{y}_b$ direction is achieved by actuating wheels 1 and 3 in the backward direction and wheels 2 and 4 in the forward direction (at the same speed).
- a rotation is achieved by moving wheels 1 and 4 in the same direction and wheels 3 and 2 in the opposite direction.

Now, similarly to what has already been done for the arm, in order to fully describe the state of the vehicle, a vector $\boldsymbol{q}_v$ is introduced. We refer to this vector as the set of *vehicle's configuration variables*,

$$\boldsymbol{q}_v = [x_v \quad y_v \quad \phi_v \quad \theta_{w,1} \dots \theta_{w,4}] \in \mathbb{R}^{78} \tag{2.31}$$

Where x_v, y_v, ϕ_v describe its 2D pose, while $\theta_{w,1} \dots \theta_{w,4}$ represent the rotation angles of the wheels.

Recalling the predefined frame $\{S\}$, that from here on will be referenced as *space frame* it is possible to describe the orientation and the position of the vehicle by means of a transformation matrix

$${}^{S}_{V}\boldsymbol{T}(\boldsymbol{q}_{v_{1:3}}) = \begin{bmatrix} \cos\phi_v & -\sin\phi_v & 0 & x_v \\ \sin\phi_v & \cos\phi_v & 0 & y_v \\ 0 & 0 & 1 & h \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.32}$$

Jacobian of the vehicle To derive the expression describing the Jacobian of the vehicle, the first 3 elements of $\boldsymbol{q}_v$ will be initially neglected. Equation 2.33 already describes the connection between wheels rate of change and the velocity of the vehicle. However this has been done considering only planar quantities. In order to extend the formulation to the whole 3D space, the twist and $\boldsymbol{H}^{\dagger}(0)$ will be rewritten as ${}^{V}\mathcal{V}_{v,6} = [v_{vx} \quad v_{vy} \quad 0 \quad 0 \quad 0 \quad \omega_{vz}]^T$ and $\boldsymbol{H}^{\dagger}_6(0) = [\boldsymbol{0}_4 \quad \boldsymbol{0}_4 \quad \boldsymbol{H}^{\dagger}(0) \quad \boldsymbol{0}_4]^T \in \mathbb{R}^{6\times 4}$. Premultiplying the body twist for $[Ad_{{}^{S}_{V}\boldsymbol{T}}]$ the space twist is obtained [7]:

$${}^{S}\mathcal{V}_{v,6} = [Ad_{{}^{S}_{V}\boldsymbol{T}}]{}^{V}\mathcal{V}_{v,6} = [Ad_{{}^{S}_{V}\boldsymbol{T}}]\boldsymbol{H}^{\dagger}_6(0)\dot{\boldsymbol{q}}_{v_{4:7}} \tag{2.33}$$

where

[8] $\boldsymbol{q}_v \in \mathbb{R}^7$, however, the degrees of freedom of the vehicle remain 3 due to the direct relationship between wheel angular motion and vehicle motion. The state of the wheels has been introduced for the sake of completeness and it will be also useful for modelling the dynamics

$$[Ad_{{}^S_V\boldsymbol{T}}] = \begin{bmatrix} {}^S_V\boldsymbol{R} & [\boldsymbol{p}_{SV}]^S_V\boldsymbol{R} \\ \boldsymbol{0}_{3\times3} & {}^S_V\boldsymbol{R} \end{bmatrix}, \quad {}^S_V\boldsymbol{R} = \begin{bmatrix} \cos\phi_v & -\sin\phi_v & 0 \\ \sin\phi_v & \cos\phi_v & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{p}_{SV} = \begin{bmatrix} x_v \\ y_v \\ h \end{bmatrix}$$

The matrix linking the vehicle twist ${}^S\mathcal{V}_6$ to the velocity of the wheels $\dot{\boldsymbol{\phi}}$ is itself a Jacobian. This Jacobian will be referred as:

$${}^S\boldsymbol{J}_{v,w}(\boldsymbol{q}_{v_{1:3}}) = [Ad_{{}^S_V\boldsymbol{T}}]\boldsymbol{H}_6^\dagger(0) \tag{2.34}$$

However, in many cases, the low-level controllers of the wheels are masked to the user, therefore it is not needed to address for the wheels' angular motion and the Jacobian of the vehicle becomes a trivial matrix:

$${}^S\boldsymbol{J}_v = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.35}$$

and

$${}^S\mathcal{V}_{v,6} = {}^S\boldsymbol{J}_v\boldsymbol{q}_{v_{1:3}} \tag{2.36}$$

2.2.3 Kinematics of a mobile manipulator

Thanks to *macro-mini manipulators theory* introduced by Khatib in [2], integrating together the manipulator and vehicle's model is easy.

This theory describes the effects of serially interconnecting a bulky *macro-manipulator*, attached to the ground, and a lighter *micro-manipulator*. Since the mobile base can be considered as a 3 DOF manipulator composed of two prismatic joints and one revolute joint, and that the connection of the mobile base to the arm leads to a serial manipulator, this theory applies also to the mobile manipulators' case.

As a consequence, the mobile base is reconducted to the macro-manipulator case, while the arm constitutes the micro-manipulator. If the vehicle and the arm configuration spaces are described, respectively, by two sets of configuration variables $\boldsymbol{q}_v$ and $\boldsymbol{q}_a$, then the set of configuration variables belonging to their union is:

$$\boldsymbol{q} = \begin{bmatrix} \boldsymbol{q}_v & \boldsymbol{q}_a \end{bmatrix} \tag{2.37}$$

And the transformation matrix that relates a frame of reference $\{E\}$ attached to the end-effector and the fixed space frame becomes:

$$ {}^{S}_{E}\boldsymbol{T}(\boldsymbol{q}) = {}^{S}_{V}\boldsymbol{T}(\boldsymbol{q}_v){}^{V}_{A}\boldsymbol{T}{}^{A}_{E}\boldsymbol{T}(\boldsymbol{q}_a) \tag{2.38} $$

where $\{A\}$ is a frame of reference attached to the first link of the arm (in 2.2.1 it was referred to as $\{0\}$) and $\{V\}$ is the aforementioned reference frame attached to the vehicle geometrical center.

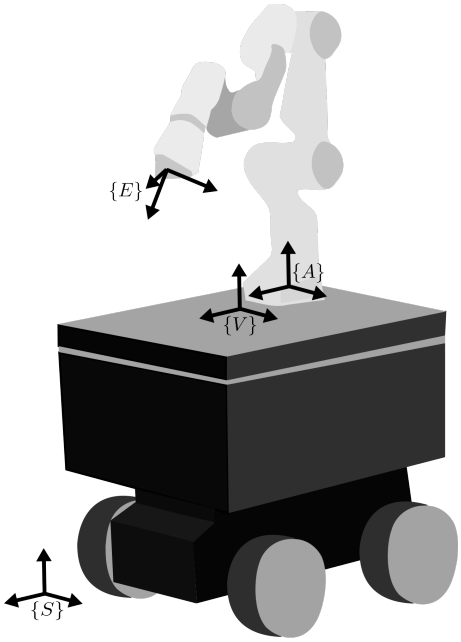


Figure 2.5: Mobile manipulator frames of reference

Jacobian of a mobile manipulator Similarly to what happens for manipulators, also in the case of mobile manipulators the motion that is of most interest is the one of the end-effector. It is clear that both the base and the arm contribute to this motion, thus, one may think to simply stack the vehicle and the manipulator Jacobians together analogously to what has been done for the joints Jacobians in the manipulator case. Even though this is a good approach, the base Jacobian relates the base configuration velocities to the base task velocities and not to the end-effector task velocities, furthermore, the end-effector frame $\{E\}$ will be rotated in space by the rotation of the mobile base.

Therefore, to retrieve the contribution of the base to the end-effector motion is beneficial to consider the velocity of a point (e.g. $\{E\}$'s origin) w.r.t. a moving frame (e.g. $\{V\}$) [2].

For the linear part we have [9]:

$$ {}^{S}\boldsymbol{v}_e = {}^{S}\boldsymbol{v}_{e,a} + {}^{S}\boldsymbol{v}_v + {}^{S}\boldsymbol{\omega}_v \times {}^{S}\boldsymbol{p}_{ve} \tag{2.39} $$

where $\boldsymbol{p}_{ve}$ can be evaluated by taking into account the first 3 components of the fourth column of ${}^{V}_{E}\boldsymbol{T} = {}^{B}_{0}\boldsymbol{T}{}^{0}_{E}\boldsymbol{T}(\boldsymbol{q}_a)$ and ${}^{S}\boldsymbol{v}_{e,a} = {}^{S}_{A}\boldsymbol{R}{}^{A}\boldsymbol{v}_{e,a}$.

[9] here the subscript $_{e,a}$ has been used to stress that this velocity is caused by the arm joints only, thus distinguishing it from the total velocity of the end-effector

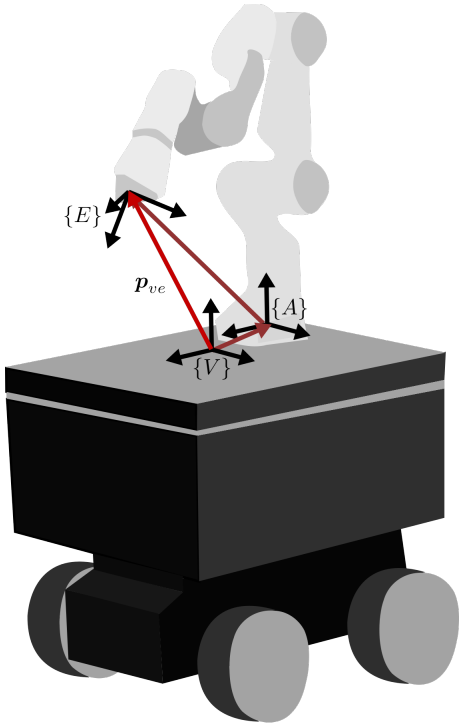


Figure 2.6: Displacement vector p_{ve}

For the angular part, instead,

$$^{S}\boldsymbol{\omega}_e = {}^{S}\boldsymbol{\omega}_{e,a} + {}^{S}\boldsymbol{\omega}_v \tag{2.40}$$

where ${}^{S}\boldsymbol{\omega}_{e,a} = {}^{S}_{A}\boldsymbol{R}\,{}^{A}\boldsymbol{\omega}_{e,a}$

The two relations together lead to,

$$\begin{bmatrix} {}^{S}\boldsymbol{v}_e \\ {}^{S}\boldsymbol{\omega}_e \end{bmatrix} = \begin{bmatrix} \boldsymbol{I}_{3\times3} & -[{}^{S}\boldsymbol{p}_{ve}] \\ \boldsymbol{0}_{3\times3} & \boldsymbol{I}_{3\times3} \end{bmatrix} \begin{bmatrix} {}^{S}\boldsymbol{v}_v \\ {}^{S}\boldsymbol{\omega}_v \end{bmatrix} + \begin{bmatrix} {}^{S}_{0}\boldsymbol{R} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & {}^{S}_{0}\boldsymbol{R} \end{bmatrix} \begin{bmatrix} {}^{S}\boldsymbol{v}_{e,a} \\ {}^{S}\boldsymbol{\omega}_{e,a} \end{bmatrix} = \begin{bmatrix} \boldsymbol{V}_v(\boldsymbol{q}_a)^{S}\boldsymbol{J}_v & \boldsymbol{V}_a(\boldsymbol{q}_v)^{S}\boldsymbol{J}_a(\boldsymbol{q}_a) \end{bmatrix} \begin{bmatrix} \boldsymbol{q}_{v_{1:3}} \\ \boldsymbol{q}_a \end{bmatrix} \tag{2.41}$$

where $\boldsymbol{V}_v = \begin{bmatrix} \boldsymbol{I}_{3\times3} & -[{}^{S}\boldsymbol{p}_{BE}] \\ \boldsymbol{0}_{3\times3} & \boldsymbol{I}_{3\times3} \end{bmatrix}$, $\boldsymbol{V}_a(\boldsymbol{q}_v) = \begin{bmatrix} {}^{S}_{0}\boldsymbol{R} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & {}^{S}_{0}\boldsymbol{R} \end{bmatrix}$

and the anticommutative property of cross product has be exploited to swap ${}^{S}\boldsymbol{p}_{ve}$ and ${}^{S}\boldsymbol{\omega}_v$. The wheels have been neglected for simplicity.

To conclude, the *whole-body geometric Jacobian* is formulated as,

$${}^{S}\boldsymbol{J} = \begin{bmatrix} \boldsymbol{V}_v(\boldsymbol{q}_a)^{S}\boldsymbol{J}_v & \boldsymbol{V}_a(\boldsymbol{q}_v)^{S}\boldsymbol{J}_a(\boldsymbol{q}_a) \end{bmatrix} \tag{2.42}$$

2.3 Dynamics of omnidirectional mobile manipulator

Dynamics is the branch of mechanics that studies forces as the causes of a motion.

Understanding the dynamics, and modelling it, is crucial for developing control techniques such as impedance control, as it will be better explained in the next section.

The most used techniques to derive the dynamics of a robot are the *Lagrange formulation* and the *Newton-Euler formulation*, their detailed explanation is depicted in [1].

In this section, the resulting dynamic models of a manipulator and of a mobile base will be presented and then joined together to describe the mobile manipulator's dynamics.

Differently from what seems to be a trend today, at least in mobile manipulation [8] [9] where the dynamic couplings between arm and base are neglected under the assumption of quasi-static conditions of the base during the manipulation task, here, they will be considered since this hypothesis is in very contrast with the objective of this thesis.

2.3.1 Dynamics in configuration space

Configuration space dynamics puts in relation the torques produced by the joints with their motions and constitutes the starting point for deriving dynamical models since this is the direct outcome of the Lagrangian and Newton-Euler approaches.

Dynamics of a manipulator

As written in many references over the literature, the dynamics of a manipulator can be modeled as [10]:

$$\boldsymbol{M}_a(\boldsymbol{q}_a)\ddot{\boldsymbol{q}}_a + \boldsymbol{C}_a(\boldsymbol{q}_a, \dot{\boldsymbol{q}}_a) + \boldsymbol{g}_a(\boldsymbol{q}_a) = \boldsymbol{\tau}_a + \boldsymbol{\tau}_a^{ext}, \tag{2.43}$$

Where $\boldsymbol{q}_a \in \mathbb{R}^{n_a}$ is the vector of the joint coordinates, $\boldsymbol{M}_a \in \mathbb{R}^{n_a \times n_a} > 0$ is the inertial matrix of the arm, $\boldsymbol{C}_a \in \mathbb{R}^{n_a}$ is deputed to describe the Coriolis and centrifugal force, $\boldsymbol{g}_a \in \mathbb{R}^{n_a}$ is the gravity vector, $\boldsymbol{\tau}_a$ and $\boldsymbol{\tau}_a^{ext} \in \mathbb{R}^{n_a}$ are the input and external torque vectors.

Dynamics of a omnidirectional vehicle

In the same way, applying the Lagrangian formulation to a vehicle leads to the subsequent expression [10]:

$$\boldsymbol{M}_v(\boldsymbol{q}_v)\ddot{\boldsymbol{q}}_v + \boldsymbol{C}_v(\boldsymbol{q}_v, \dot{\boldsymbol{q}}_v) = \boldsymbol{E}_v(\boldsymbol{q}_v)\boldsymbol{\tau}_v - \boldsymbol{A}_v^T(\boldsymbol{q}_v)\boldsymbol{\lambda}, \tag{2.44}$$

Where $\boldsymbol{q}_v \in \mathbb{R}^{n_v}$ is the vector describing the pose of the vehicle ($n_v =$ number of the DoF of the mobile platform), $\boldsymbol{M}_v \in \mathbb{R}^{n_v \times n_v} > 0$ is the inertial matrix of the arm, $\boldsymbol{C}_v \in \mathbb{R}^{n_v}$ is deputed to describe the Coriolis and centrifugal force, $\boldsymbol{E}_v \in \mathbb{R}^{n_v \times n_v} > 0$ is the input transformation

matrix [10], $\boldsymbol{\tau}_v \in \mathbb{R}^{n_v}$ is the input torque vector, $\boldsymbol{A}_v \in \mathbb{R}^{n_v \times n_v}$ is the constraint matrix and $\boldsymbol{\lambda} \in \mathbb{R}^{n_v}$ is the Lagrangian multiplayer which denotes the constraint force vector.

In the case of an omnidirectional wheeled vehicle $\boldsymbol{A}_v = \boldsymbol{0}_{n_v \times n_v}$ since no constraint is applied.

Dynamics of a mobile manipulator

When two bodies interact with each other, it is needed to extend their models by considering the motion contributes that each part's dynamics causes on the other. These contributes are called *dynamical couplings* and lead to a reformulation of 2.43 and 2.44 [2][12][10]:

$$\begin{aligned} \boldsymbol{M}_a(\boldsymbol{q}_a)\ddot{\boldsymbol{q}}_a + \boldsymbol{C}_a(\boldsymbol{q}_a, \dot{\boldsymbol{q}}_a) + \boldsymbol{C}_{av}(\boldsymbol{q}_a, \dot{\boldsymbol{q}}_a, \dot{\boldsymbol{q}}_v) + +\boldsymbol{g}_a(\boldsymbol{q}_a) \\ = \boldsymbol{\tau}_a + \boldsymbol{\tau}_a^{ext} - \boldsymbol{M}_{av}(\boldsymbol{q}_a, \boldsymbol{q}_v)\ddot{\boldsymbol{q}}_v. \end{aligned} \tag{2.45}$$

where $\boldsymbol{C}_{av} \in \mathbb{R}^{n_a}$ represents Coriolis and centrifugal terms caused by angular motion of the mobile platform, $\boldsymbol{M}_{va} \in \mathbb{R}^{n_a \times n_v}$ is the inertial matrix which represents the effect of the mobile platform dynamics on the manipulator.

$$\begin{aligned} \boldsymbol{M}_v(\boldsymbol{q}_v)\ddot{\boldsymbol{q}}_v + \boldsymbol{C}_v(\boldsymbol{q}_v, \dot{\boldsymbol{q}}_v) + \boldsymbol{C}_{va}(\boldsymbol{q}_v, \boldsymbol{q}_a, \dot{\boldsymbol{q}}_v, \dot{\boldsymbol{q}}_a) \\ = \boldsymbol{E}_v(\boldsymbol{q}_v)\boldsymbol{\tau_v} - \boldsymbol{A}_v^T(\boldsymbol{q}_v)\boldsymbol{\lambda} - \boldsymbol{M}_{a,up}(\boldsymbol{q}_v, \boldsymbol{q}_a)\ddot{\boldsymbol{q}}_v \\ - \boldsymbol{M}_{va}(\boldsymbol{q}_v, \boldsymbol{q}_a)\ddot{\boldsymbol{q}}_a, \end{aligned} \tag{2.46}$$

where $\boldsymbol{M}_{va} \in \mathbb{R}^{n_v \times n_v}$ and $\boldsymbol{C}_{va} \in \mathbb{R}^{n_v}$ denote the inertial term and Coriolis and centrifugal terms due to the presence of the manipulator, $\boldsymbol{M}_{av} = \boldsymbol{M}_{va}^T \in \mathbb{R}^{n_v \times n_a}$ is the inertial matrix which reflects the dynamic effect of the manipulator motion on the mobile platform.

In the end the complete model of the whole body considering both the arm and the base can be written in a more compact form:

$$\begin{aligned} \begin{bmatrix} \boldsymbol{M}_v + \boldsymbol{M}_{a,up} & \boldsymbol{M}_{va} \\ \boldsymbol{M}_{va}^T & \boldsymbol{M}_a \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}}_v \\ \ddot{\boldsymbol{q}}_a \end{bmatrix} + \begin{bmatrix} \boldsymbol{C}_v & \boldsymbol{C}_{va} \\ \boldsymbol{C}_{av} & \boldsymbol{C}_a \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_v \\ \dot{\boldsymbol{q}}_a \end{bmatrix} + \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{g}_a \end{bmatrix} \\ = \begin{bmatrix} \boldsymbol{E}_v(\boldsymbol{q}_v)\boldsymbol{\tau}_v \\ \boldsymbol{\tau}_a \end{bmatrix} + \begin{bmatrix} \boldsymbol{\tau}_v^{ext} \\ \boldsymbol{\tau}_a^{ext} \end{bmatrix} - \boldsymbol{A}_v \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{\lambda} \end{bmatrix} \end{aligned} \tag{2.47}$$

It's worth noticing that there is no $\boldsymbol{M}_{v,up}$ matrix to be added to the $\boldsymbol{M}_a$ matrix because the arm is built on top of the mobile base, hence it doesn't add any mass to the arm motions.

As mentioned before, for omnidirectional wheeled vehicles the last term involving $\boldsymbol{A}_v$ can be

[10] e.g. for a two wheels mobile robot, having one actuator per wheel the $\boldsymbol{E}_v$ matrix will be $\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$ [11]

deleted from the equation. Thanks to the omnidirectional wheels there are no constraints to be taken into account.

This equation can be also written in a decoupled fashion by moving all the terms that involve both the bodies at the second member. This is particularly useful to understand which is the actual torque exerted by the arm or by the base.

$$\begin{bmatrix} \boldsymbol{M}_v & \mathbf{0} \\ \mathbf{0} & \boldsymbol{M}_a \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}}_v \\ \ddot{\boldsymbol{q}}_a \end{bmatrix} + \begin{bmatrix} \boldsymbol{C}_v & \mathbf{0} \\ \mathbf{0} & \boldsymbol{C}_a \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_v \\ \dot{\boldsymbol{q}}_a \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \boldsymbol{g}_a \end{bmatrix} = \begin{bmatrix} \boldsymbol{\tau}_v \\ \boldsymbol{\tau}_a \end{bmatrix} + \begin{bmatrix} \boldsymbol{\tau}_v^{ext} \\ \boldsymbol{\tau}_a^{ext} \end{bmatrix} - \begin{bmatrix} \boldsymbol{M}_{a,up} & \boldsymbol{M}_{va} \\ \boldsymbol{M}_{va}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}}_v \\ \ddot{\boldsymbol{q}}_a \end{bmatrix} - \begin{bmatrix} \mathbf{0} & \boldsymbol{C}_{va} \\ \boldsymbol{C}_{av} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_v \\ \dot{\boldsymbol{q}}_a \end{bmatrix} \tag{2.48}$$

From this view, it becomes clear that the couplings will be acting as disturbances for what regards most of the tasks.

Dynamics of a mobile manipulator considering the wheels To include the wheels in the model dynamics, $\boldsymbol{q}_v$ is extended as follows

$\boldsymbol{q}_v = [x_v \quad y_v \quad \phi_v \quad \theta_{w,1} \dots \theta_{w,n_w}]$, therefore $n_v = 3 + n_w$.

Then, similarly to what has already been done for the arm and the chassis, the couplings are added into the inertia and Coriolis matrices:

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{M}_v + \boldsymbol{M}_{a,up} & \boldsymbol{M}_{vw} & \boldsymbol{M}_{va} \\ \boldsymbol{M}_{vw}^T & \boldsymbol{M}_w & \mathbf{0}_{n_w \times n_a} \\ \boldsymbol{M}_{va}^T & \mathbf{0}_{n_a \times n_w} & \boldsymbol{M}_a \end{bmatrix} \tag{2.49}$$

$$\boldsymbol{C} = \begin{bmatrix} \boldsymbol{C}_v & \boldsymbol{C}_{vw} & \boldsymbol{C}_{va} \\ \boldsymbol{C}_{wv} & \boldsymbol{C}_w & \mathbf{0}_{n_w \times n_a} \\ \boldsymbol{C}_{va}^T & \mathbf{0}_{n_a \times n_w} & \boldsymbol{C}_a \end{bmatrix} \tag{2.50}$$

where $\boldsymbol{M}_{vw} = M_{wv}^T \in \mathbb{R}^{n_v \times n_w}$, $\boldsymbol{C}_{vw} = C_{wv}^T \in \mathbb{R}^{n_v \times n_w}$ describe the couplings between the wheels and the chassis, $\boldsymbol{M}_w$, $\boldsymbol{C}_w$ are the mass and coriolis matrix related to the wheels and the 0 matrices are included because the arm and the wheels have do not influence each other directly.

In the end the whole body dinamical model is,

$$\begin{bmatrix} \boldsymbol{M}_v + \boldsymbol{M}_{a,up} & \boldsymbol{M}_{vw} & \boldsymbol{M}_{va} \\ \boldsymbol{M}_{vw}^T & \boldsymbol{M}_w & \boldsymbol{0}_{n_w \times n_a} \\ \boldsymbol{M}_{va}^T & \boldsymbol{0}_{n_a \times n_w} & \boldsymbol{M}_a \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}}_v \\ \ddot{\boldsymbol{q}}_a \end{bmatrix} + \begin{bmatrix} \boldsymbol{C}_v & \boldsymbol{C}_{vw} & \boldsymbol{C}_{va} \\ \boldsymbol{C}_{wv} & \boldsymbol{C}_w & \boldsymbol{0}_{n_w \times n_a} \\ \boldsymbol{C}_{va}^T & \boldsymbol{0}_{n_a \times n_w} & \boldsymbol{C}_a \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_v \\ \dot{\boldsymbol{q}}_a \end{bmatrix} + \begin{bmatrix} \boldsymbol{0}_{n_v \times 1} \\ \boldsymbol{g}_a \end{bmatrix} = \begin{bmatrix} \boldsymbol{\tau}_v \\ \boldsymbol{\tau}_a \end{bmatrix} + \begin{bmatrix} \boldsymbol{\tau}_v^{ext} \\ \boldsymbol{\tau}_a^{ext} \end{bmatrix} \tag{2.51}$$

2.3.2 Dynamics in task space

Most of the times robots are required to execute motions that are defined in task space (also called *operational space*). By exploiting the link between torques and forces,

$$\boldsymbol{\tau} = \boldsymbol{J}(\boldsymbol{q})^T \boldsymbol{F} \tag{2.52}$$

it is possible to extend the just-derived models to task space.

As introduced by Khatib in [13], applying equations 2.5 and 2.52 to 2.47 the dynamics in task space can be written as,

$$\boldsymbol{\Lambda}(\boldsymbol{x})\ddot{\boldsymbol{x}} + \boldsymbol{\mu}(\boldsymbol{x}, \dot{\boldsymbol{x}})\dot{\boldsymbol{x}} + \overline{\boldsymbol{J}}(\boldsymbol{q})^T \boldsymbol{g}(\boldsymbol{q}) = \overline{\boldsymbol{J}}(\boldsymbol{q})^T \boldsymbol{\tau} + \boldsymbol{F}_{ext}, \tag{2.53}$$

where

a) $\boldsymbol{\Lambda}(\boldsymbol{x}) = (\boldsymbol{J}(\boldsymbol{q})\boldsymbol{M}^{-1}(\boldsymbol{q})\boldsymbol{J}^T(\boldsymbol{q}))^{-1}$

b) $\overline{\boldsymbol{J}}(\boldsymbol{q}) = \boldsymbol{M}^{-1}(\boldsymbol{q})\boldsymbol{J}^T(\boldsymbol{q})(\boldsymbol{J}(\boldsymbol{q})\boldsymbol{M}^{-1}(\boldsymbol{q})\boldsymbol{J}^T(\boldsymbol{q}))^{-1}$

c) $\boldsymbol{\mu}(\boldsymbol{x}, \dot{\boldsymbol{x}}) = \overline{\boldsymbol{J}}(\boldsymbol{q})^T(\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \boldsymbol{M}(\boldsymbol{q})\overline{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{J}}(\boldsymbol{q}))\overline{\boldsymbol{J}}(\boldsymbol{q})$

$\boldsymbol{\Lambda}(\boldsymbol{x})$ and $\boldsymbol{\mu}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ are the inertia matrix and the Coriolis/centrifugal matrix in task space coordinates $\boldsymbol{x}$ and $\overline{\boldsymbol{J}}(\boldsymbol{q})$ is the *dynamically consistent pseudo-inverse*.

To understand the meaning of *dynamically consistent pseudo-inverse* it is necessary to introduce some theoretical background. The inverse kinematics problem has an infinite number of solutions, in fact, since $\boldsymbol{J}$ is rectangular, there is not a unique $\dot{\boldsymbol{q}}$ that satisfies the relation $\boldsymbol{J}\dot{\boldsymbol{q}} = \dot{\boldsymbol{x}}$. As a consequence there is not a unique inverse of the Jacobian $\boldsymbol{J}^{-1}$ that maps $\dot{\boldsymbol{x}}$ into $\dot{\boldsymbol{q}}$. A possible workaround is the *Moore-Penrose inverse* which is obtained by exploiting the least square method to find the matrix $\overline{\boldsymbol{J}}$ that minimizes the norm $||\boldsymbol{J}\dot{\boldsymbol{q}}^* - \boldsymbol{x}||$ and results in the solution $\dot{\boldsymbol{q}}^*$ with minimum norm [14].

There is also a weighted version of the Moore-Penrose inverse, that in this case, will result in the pseudo-inverse corresponding to the minimal solution $\dot{\boldsymbol{q}}^{*T}\boldsymbol{W}^T\boldsymbol{W}\dot{\boldsymbol{q}}^*$ [15], where $\boldsymbol{W}$ is the chosen weight matrix. Therefore, setting $\boldsymbol{W} = \boldsymbol{M}$ we ensure that $\overline{\boldsymbol{J}}(\boldsymbol{q})$ is the inverse corresponding to the solution that minimizes the kinetic energy of the robot (namely $T = \frac{1}{2}\dot{\boldsymbol{q}}\boldsymbol{M}\dot{\boldsymbol{q}}$).

Dynamic consistency property will also be beneficial in the redundancy resolution as explained in section 2.5.3.

2.3.3 Remarkable properties of dynamic model

The approach adopted to model the dynamics is well-known in robotics and carries with itself some properties that are useful also for control, in particular, [1]:

- $\boldsymbol{M}(\boldsymbol{q})$ is symmetric positive definite
- $\dot{\boldsymbol{M}} - 2\boldsymbol{C}$ is a skew-symmetric matrix

For the second property, it is worth mentioning its link with the energy balance of the system, in fact, it can be proven that it is a direct consequence of the *principle of conservation of energy* [1], making it useful to do considerations on passivity and stability.

These two properties can be extended also to the task space model, a proof of this can be found in [4].

2.4 Center of mass and total mass identification

Using model-based approaches to control a system, it is fundamental to have a good estimation of its parameters. While most of the parameters of the hardware that this thesis takes as reference for testing its outcomes (see section 3.1.1) were provided by the manufacturers, this was not the case for the dynamic ones (e.g. body inertia, center of mass, total mass) of the mobile base. Taking in consideration that no specific instrumentation was available and that this is not the main focus of the thesis, the identification esperiments were chosen accordingly to the best compromise among accuracy, cost efficiency and simplicity.

In this section an experimental way of determining the total mass and the center of mass of vehicle is advanced. The estimation of the inertia has been avoided since it requires a more complex and time-consuming approach, such as using system identification techniques as explained in [1]. In its place, the data provided from the manufacturer for a similar platform has been adopted.

Center of mass identification The idea that has been followed and that is also sustained by [16] is based on the fact that the robot in static conditions satisfies the relations:

$$\sum_i \boldsymbol{F}_i = 0 \tag{2.54}$$

$$\sum_i \boldsymbol{M}_i = 0 \tag{2.55}$$

where $\boldsymbol{F}_i$ and $\boldsymbol{M}_i$ are, respectively, the forces and the moments acting on the robot.

Considering:

- a frame of reference O placed in the geometrical center of the chassis
- the robot being parallel to the pavement
- the contact forces $\boldsymbol{F}_i$ between each wheel and their contact point
- the total weight of the chassis $\boldsymbol{W}$ applied in the center of mass
- the distances $\boldsymbol{d}_i$ and $\boldsymbol{d}_{COM}$ between each of the aforementioned forces point of application and the origin on O

By applying 2.55 the x and y coordinate of the center of mass are given:

$$\begin{bmatrix} d_{COM_x} \\ d_{COM_y} \end{bmatrix} = \frac{1}{W} \sum_i F_{i_z} \begin{bmatrix} d_{i_x} \\ d_{i_y} \end{bmatrix} \tag{2.56}$$

Only F_{i_z} component has been considered because the robot is placed parallel to the terrain.

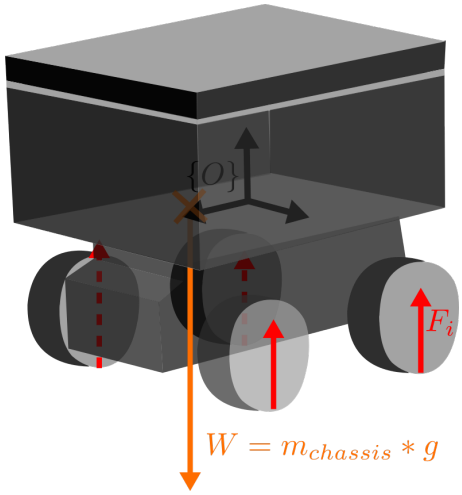


Figure 2.7

In order to get also the z component of the center of mass another experiment with the robot in a different configuration must be carried out [16] and this is a direct consequence of the fact that we are measuring forces that are all parallel among each other. The ideal way to do so would be to flip the robot and repeat the experiment, however this is not practically advisable since no guarantee is provided on the resistance of the structure in that configuration. Therefore, instead of completely flip it, the robot can been tilted by placing a wedge underneath two of the four wheels.

This leads to a further equation that is:

$$d_{COM_z} = \frac{1}{W_x}(\sum_i F_{i_z} d_{i_x} - \sum_i F_{i_x} d_{i_z}) + \frac{W_z}{W_x} d_{COM_x} \tag{2.57}$$

where $F_{i_z} = F_i cos(\alpha)$ and $F_{i_x} = F_i sin(\alpha)$, an analogous reasoning can be done for the total weight components W_x and W_z.

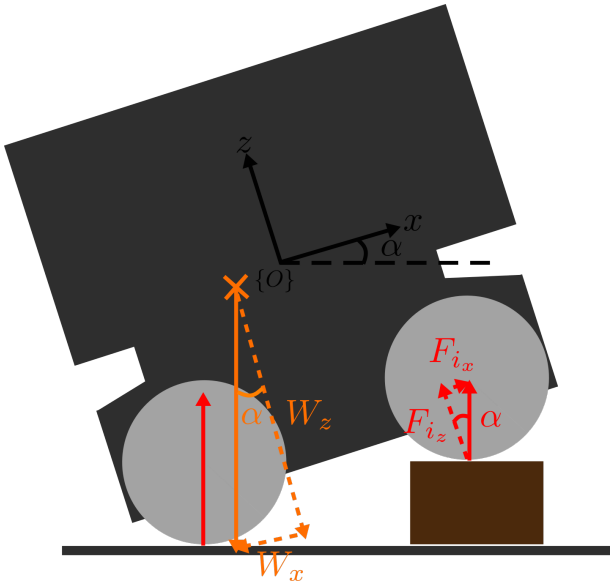


Figure 2.8

Total mass identification The identification of the total mass comes easily by measuring the weight force exerted on each contact point of the mobile base, that, in this case, are simply the points in which the wheels touch the terrain.

$$m_{chassis} = \sum_i F_{i_z} * g \tag{2.58}$$

2.5 Whole-body control of mobile manipulator

Controlling a mobile manipulator is not an easy task, the whole platform has a high amount of degrees of freedom, the base and the arm are subject to dynamic couplings and their actuators work at different rates [11]. These are the main reasons why, currently, in most applications, the base and the arm are controlled separately, in a sequential way, thus limiting the actual potential of the hardware and reducing productivity, this is observable both in industry and in research (e.g. [8]). Some have tried a more holistic approach such as [17] [18], but have focused mainly on kinematics, thus making their solution undesirable for practices that require compliance with the environment. Definitely, much more can be done in this sense.

The problem of developing a whole-body controller, which is also compliant and that doesn't neglect the arm-base couplings is not a heavily explored topic, only a few studies have been found, such as [9], which unluckily was discovered only in the final stages of this thesis, but provides a good confirmation that a model based whole-body impedance control constitutes a viable option.

In the discussion about whole-body control, any reference to optimization techniques will be avoided since human-populated environments require a reliable and known behaviour

[11] Typically the base has slower dynamics

and by employing a model-based approach it is possible to achieve a more complete understanding of the problem.

When a robot comes into contact with the environment its motion becomes constrained by the surroundings. Because it is unusual to own an accurate model of the space around the robot, it is fundamental to tackle compliant movements in order to automatically react to the force constraints exerted by the environment and avoid any possible source of instability. To better explain, complying means to interact with the objects placed in the vicinity, in a non-rigid manner, at the cost of moving slightly away from the desired trajectory [1]. Compliance control, thus, aims to ensure a desired disturbance response to external forces while minimizing the deviation from the desired path.

A way of dealing with compliance is called impedance control (also referred to as force-based impedance control), whose key concept is to make the robot follow a desired dynamic behaviour. A sufficient mathematical representation of this desired dynamic behaviour is usually constituted by the mass-damper-spring model that is written concerning the pose error $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x}_d$ between the end-effector pose $\boldsymbol{x}$ and the virtual equilibrium point $\boldsymbol{x}_d$[12] [4]:

$$\boldsymbol{\Lambda}_d \ddot{\tilde{\boldsymbol{x}}} + \boldsymbol{D}_d \dot{\tilde{\boldsymbol{x}}} + \boldsymbol{K}_d \tilde{\boldsymbol{x}} = \boldsymbol{F}_{ext} \tag{2.59}$$

where $\boldsymbol{K}_d > 0$, $\boldsymbol{D}_d > 0$ and $\boldsymbol{\Lambda}_d > 0$ are the matrices of desired stiffness, damping and inertia, while $\boldsymbol{F}_{ext} = \boldsymbol{J}^{-T} \boldsymbol{\tau}_{ext}$ is the vector of the *generalized forces* that the environment exerts on the robot.

Being a simple second-order system with positive-definite matrices, in the free-motion case $\boldsymbol{F}_{ext} = 0$, ($\boldsymbol{x}$ will tend asymptotically to $\boldsymbol{x}_d$ and the closed-loop system will be asymptotically stable.

It is clear that by tuning:

- $\boldsymbol{K}_d$ - the robot will act more or less rigidly, which means that it will allow smaller or bigger changes in the tracking of the desired motion
- $\boldsymbol{D}_d$ - the robot will adapt to changes in a faster or slower way and with more or fewer oscillations
- $\boldsymbol{\Lambda}_d$ - the robot will keep strongly or weakly its previous state of motion

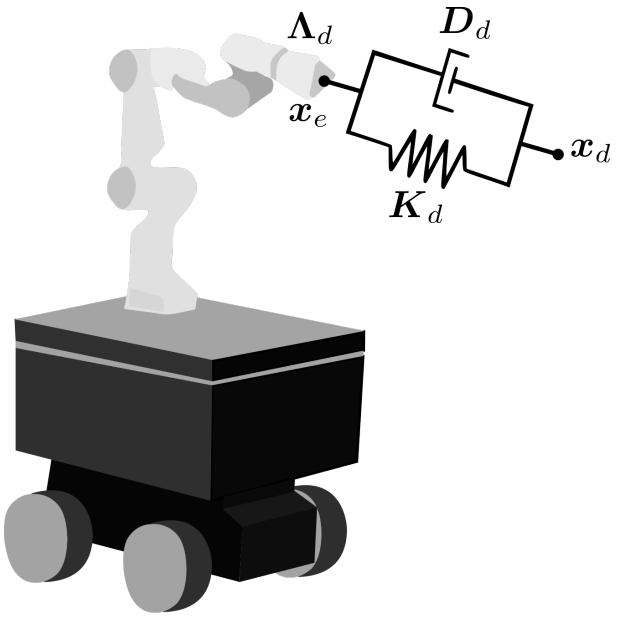
Now, from 2.3 derives that the motion equations of the robot are non-linear. The classical impedance controller introduced by Hogan in [19] acts on these non-linearities by directly compensating them:

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q}) \boldsymbol{\tau}_{imp} + \boldsymbol{\tau}_{comp} \tag{2.60}$$

where $\boldsymbol{\tau}_{comp} = \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q})$ and $\boldsymbol{\tau}_{imp}$ is employed to enforce the desired impedance behaviour.

More specifically, this control law in task space formulation coincides with [4],

[12] In the impedance control literature, the desired setpoint $\boldsymbol{x}_d$ usually is called a *virtual desired setpoint* (or *virtual equilibrium point*), since it can only be reached in case of free motion, i.e. when no external forces act on the robot.



$$\boldsymbol{\tau} = \boldsymbol{J}(\boldsymbol{q})^T \boldsymbol{F}_\tau =$$
$$\boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{J}(\boldsymbol{q})^T \left(\boldsymbol{\Lambda}(\boldsymbol{x})\ddot{\boldsymbol{x}}_d + \boldsymbol{\mu}(\boldsymbol{x}, \dot{\boldsymbol{x}})\dot{\boldsymbol{x}}\right) - \boldsymbol{J}(\boldsymbol{q})^T \boldsymbol{\Lambda}(\boldsymbol{x})\boldsymbol{\Lambda}_d^{-1} \left(\boldsymbol{K}_d\tilde{\boldsymbol{x}} + \boldsymbol{D}_d\dot{\tilde{\boldsymbol{x}}}\right) + \boldsymbol{J}(\boldsymbol{q})^T \left(\boldsymbol{\Lambda}(\boldsymbol{x})\boldsymbol{\Lambda}_d^{-1} - \boldsymbol{I}\right) \boldsymbol{F}_{ext}. \tag{2.61}$$

It can be seen that plugging this torque in 2.53, 3.10 is obtained.

Since this technique linearizes the system, many choose, $\boldsymbol{K}_d$, $\boldsymbol{D}_d$ and $\boldsymbol{\Lambda}_d$ as diagonal positive matrices to keep the system linear and decoupled. In this way, if $\boldsymbol{F}_{ext} < \infty$, the global asymptotic stability of the closed-loop system, can be also proven for the forced case, since it is characterized by a collection of 2nd-order linear differential equations with positive coefficients. Ideally, a wide range of finite, non-null, impedance parameters can be implemented without losing this property.

Nonetheless, sometimes, this approach is avoided because it involves the sensing of the external forces and complicates the problem by including the additional specification of a desired inertia matrix which might not be needed for many applications. This problem is especially important for the mobile manipulator case in human-populated environments because it would imply the placement of several force sensors all over the platform[13].

By setting $\boldsymbol{\Lambda}_d(\boldsymbol{x})$ equal to the inertia of the manipulator $\boldsymbol{\Lambda}(\boldsymbol{x})$, namely performing an *inertia shaping avoidance* [4], force sensors are no more needed and the new input torque can be written as:

$$\boldsymbol{\tau} = \boldsymbol{J}(\boldsymbol{q})^T \boldsymbol{F}_\tau = \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{J}(\boldsymbol{q})^T \left(\boldsymbol{\Lambda}(\boldsymbol{x})\ddot{\boldsymbol{x}}_d + \boldsymbol{\mu}(\boldsymbol{x}, \dot{\boldsymbol{x}})\dot{\boldsymbol{x}}_d - \boldsymbol{K}_d\tilde{\boldsymbol{x}} - \boldsymbol{D}_d(\boldsymbol{x})\dot{\tilde{\boldsymbol{x}}}\right) \tag{2.62}$$

[13] In the case of industrial manipulators, instead, one force sensor at the end-effector might be sufficient

It should be noted that in this way the closed loop dynamics is not linearized and assumes the form,

$$\boldsymbol{\Lambda}(\boldsymbol{x})\ddot{\tilde{\boldsymbol{x}}} + (\boldsymbol{D}_d(\boldsymbol{x}) + \boldsymbol{\mu}(\boldsymbol{x}, \dot{\boldsymbol{x}}))\dot{\tilde{\boldsymbol{x}}} + \boldsymbol{K}_d\tilde{\boldsymbol{x}} = \boldsymbol{F}_{ext} \tag{2.63}$$

In principle, one could try to also compensate for $\boldsymbol{\mu}$ but since $\boldsymbol{\Lambda}$ varies with $(\boldsymbol{x})$, by including $\boldsymbol{\mu}$ we respect the skew-simmetricity of $\dot{\boldsymbol{M}} - 2\boldsymbol{C}$ thus ensuring asymptotic tracking of the desired pose $\boldsymbol{x}_d$ in free motion ($\boldsymbol{F}_{ext} = \boldsymbol{0}$).

2.5.1 Stiffness and Damping choice

The choice of $\boldsymbol{K}_d$ strongly depends on the application and many are basing their tuning on experiments [4]. In general, a static choice of $\boldsymbol{D}_d$ may lead to poor performances due to the time-varying nature of $\boldsymbol{\Lambda}(\boldsymbol{q})$ [20]. Certainly, this is not the case when inertia shaping is performed ($\boldsymbol{\Lambda}_d \neq \boldsymbol{\Lambda}$) thus resulting in a fixed $\boldsymbol{\Lambda}$.

In any case, $\boldsymbol{D}_d$, can be derived from $\boldsymbol{K}_d$ and $\boldsymbol{\Lambda}$ so that the eigenvalues of the impedance dynamics are real, thus enforcing a critically damped behaviour. This result is achieved by means of the *Double diagonalization method* that consists in solving the generalized eigenvalue problem[21] [4]:

Given a symmetric positive definite matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ and a symmetric matrix $\boldsymbol{K}_d \in \mathbb{R}^{n\times n}$, a nonsingular matrix $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ can be found, such that $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{Q}^T$ and $\boldsymbol{K}_d = \boldsymbol{Q}\boldsymbol{K}_{d0}\boldsymbol{Q}^T$ for some diagonal matrix $\boldsymbol{K}_{d0} = diag(\lambda^{\boldsymbol{\Lambda}}_{\boldsymbol{K}_d,i})$ where $\lambda^{\boldsymbol{\Lambda}}_{\boldsymbol{K}_d,i}$ is the i^{th} generalized eigenvalue of $\boldsymbol{K}_d$ with respect to $\boldsymbol{\Lambda}$.

Assuming that $\boldsymbol{\Lambda}$ varies slowly, this allows choosing $\boldsymbol{D}_d$ as

$$\boldsymbol{D}_d = 2\boldsymbol{Q}(\boldsymbol{x})^T diag(\xi_i\sqrt{\lambda^{\boldsymbol{\Lambda}}_{\boldsymbol{K}_d,i}})\boldsymbol{Q}(\boldsymbol{x}) \tag{2.64}$$

with $\xi_i \in [0, 1]$ as damping factor to enforce the desired behaviour.

In this way, in fact, the system will behave as a second-order system of the form

$$\ddot{\boldsymbol{z}} + 2diag(\xi_i\sqrt{\lambda^{\boldsymbol{\Lambda}}_{\boldsymbol{K}_d,i}})\dot{\boldsymbol{z}} + diag(\lambda^{\boldsymbol{\Lambda}}_{\boldsymbol{K}_d,i})\boldsymbol{z} = \boldsymbol{Q}(\boldsymbol{x})^{-T}\boldsymbol{F}_{ext} \tag{2.65}$$

where $\boldsymbol{z} = \boldsymbol{Q}(\boldsymbol{x})\tilde{\boldsymbol{x}}$

While this technique is formally correct, solving a generalized eigenvalue problem might be difficult, especially in the proximity of singularities. To be more robust, a *factorization design* can be adopted [21]:

Let's assume to be in the free motion case, 3.10 is a system of homogeneous second-order differential equations. Now, taking one of these equations in consideration, the associated characteristic equation leads to two equal and real solutions (critically damped system) if it coincides with the square of a binomial $(a^2 + 2ab + b^2)$, extending this reasoning to the whole system, in order to make it critically damped, $\boldsymbol{D}_d$ should be chosen as:

$$\boldsymbol{D}_d = \boldsymbol{\Lambda}_{d,\frac{1}{2}} \boldsymbol{K}_{d,\frac{1}{2}} + \boldsymbol{K}_{d,\frac{1}{2}} \boldsymbol{\Lambda}_{d,\frac{1}{2}} \tag{2.66}$$

Where $\Lambda_d = \Lambda_{d,\frac{1}{2}} \Lambda_{d,\frac{1}{2}}$ and $\boldsymbol{K}_d = \boldsymbol{K}_{d,\frac{1}{2}} \boldsymbol{K}_{d,\frac{1}{2}}$.

For obtaining a different behaviour, instead $\boldsymbol{D}_d$ can be modified to [21],

$$\boldsymbol{D}_d = \boldsymbol{\Lambda}_{d,\frac{1}{2}} \boldsymbol{D}_\xi \boldsymbol{K}_{d,\frac{1}{2}} + \boldsymbol{K}_{d,\frac{1}{2}} \boldsymbol{D}_\xi \boldsymbol{\Lambda}_{d,\frac{1}{2}} \tag{2.67}$$

with $\boldsymbol{D}_\xi = diag(\xi_i)$ and $\xi_i \in [0, 1]$ where $\xi_i = 0$ corresponds to an undamped system while $\xi_i = 1$ reconnects to 2.66.

Practical insights: Large values of stiffness, which may be needed for certain tasks, lead to amplification of the noise and, thus, instability. On the other hand, a higher gain, implies a rejection of model uncertainties, making the system more robust [22]. Furthermore, it is also important to consider that high friction systems (in which friction is not compensated) and systems with poor backdrivabilty performance will end up in bad position accuracy if controlled through impedance control [22].

Control law changes due to task space representation

As Ott states in [4] the choice of coordinates affects the overall stiffness behaviour. Stiffness matrices are symmetric positive definite matrices of the shape:

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_t & \boldsymbol{K}_c \\ \boldsymbol{K}_c^T & \boldsymbol{K}_r \end{bmatrix} \tag{2.68}$$

where t and r stand for, respectively, translational and rotational stiffness elements while c describes the couplings between the translational and rotational behaviours.

It is important to remember the physical and geometrical interpretation of this matrix, in fact, the stiffness is a measure of the resistance to deformation and, also, it can be viewed as a mapping between force and displacement.

Now, decomposing $\boldsymbol{K}$ through eigendecomposition [23]:

$$\boldsymbol{K} = \boldsymbol{U} \boldsymbol{\Gamma} \boldsymbol{U}^T \tag{2.69}$$

where $\boldsymbol{U} = diag(\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3)$ is the eigenvector matrix and $\boldsymbol{\Gamma} = diag(\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \boldsymbol{\gamma}_3)$ is the eigenvalue matrix, it becomes clear that along direction $\boldsymbol{u}_i$ a displacement $\tilde{\boldsymbol{x}}$ is mapped into a force $\boldsymbol{F} = \boldsymbol{\gamma}_i \tilde{\boldsymbol{x}}$.

The vectors $(\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3)$ are also called *principal axes* of $\boldsymbol{K}$ and they remain constant in the system of reference in which the displacement is defined [4]. Therefore, the representation of the displacement error $\tilde{\boldsymbol{x}}$ directly affects in which reference frame the principal axes will be constant.

In many applications having the principal axes constant in the tool frame or in the desired

frame can ease the choice of stiffness parameters since it is known in which direction it is desirable to have a stiff motion of the end-effector and in which not.

Changing the frame of reference in which $\tilde{x}$ is defined, implies that in its time derivatives also the variations in time of the new frame have to be considered [4]:

$$\dot{\tilde{\boldsymbol{x}}} = \frac{\partial \tilde{\boldsymbol{x}}(\boldsymbol{q},t)}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} + \frac{\partial \tilde{\boldsymbol{x}}(\boldsymbol{q},t)}{\partial t} \tag{2.70}$$

$$\ddot{\tilde{\boldsymbol{x}}} = \frac{\partial \tilde{\boldsymbol{x}}(\boldsymbol{q},t)}{\partial \boldsymbol{q}}\ddot{\boldsymbol{q}} + \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \tilde{\boldsymbol{x}}(\boldsymbol{q},t)}{\partial \boldsymbol{q}}\right)\dot{\boldsymbol{q}} + \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial \tilde{\boldsymbol{x}}(\boldsymbol{q},t)}{\partial t} \tag{2.71}$$

Luckily, by calling

$$\boldsymbol{J}_x \triangleq \frac{\partial \tilde{\boldsymbol{x}}(\boldsymbol{q},t)}{\partial \boldsymbol{q}} \tag{2.72}$$

$$\boldsymbol{v}_t \triangleq -\frac{\partial \tilde{\boldsymbol{x}}(\boldsymbol{q},t)}{\partial t} \tag{2.73}$$

It is possible to infer a direct correspondence between the new formulation and 2.62 which maintains all the properties of the impedance control law introduced by Ott [4]:

$$\begin{aligned} \boldsymbol{J}(\boldsymbol{q}) &\rightarrow \boldsymbol{J}_x(\boldsymbol{q},t) \\ \dot{\boldsymbol{x}}_d &\rightarrow -\boldsymbol{v}_t(\boldsymbol{q},t) \\ \ddot{\boldsymbol{x}}_d &\rightarrow -\dot{\boldsymbol{v}}_t(\boldsymbol{q},t) \end{aligned}$$

While the equations of motion can be rewritten as:

$$\boldsymbol{\Lambda}(\boldsymbol{q},t)\left(\ddot{\tilde{\boldsymbol{x}}} + \dot{\boldsymbol{v}}_t(\boldsymbol{q},t)\right) + \boldsymbol{\mu}(\boldsymbol{q},\dot{\boldsymbol{q}},t)\left(\dot{\tilde{\boldsymbol{x}}} + \boldsymbol{v}_t(\boldsymbol{q},t)\right) + \boldsymbol{J}_x(\boldsymbol{q},t)^{-T}\boldsymbol{g}(\boldsymbol{q}) - \boldsymbol{J}_x(\boldsymbol{q},t)^{-T}\left(\boldsymbol{\tau} + \boldsymbol{\tau}_{ext}\right) = \boldsymbol{0} \tag{2.74}$$

A note on the influence of orientation representation on impedance behaviour The choice of quaternions has proven to be particularly beneficial also for attaining a meaningful impedance behaviour. As shown by Caccavale et al. in [23], in fact, by employing Euler Angles, the impedance behaviour depends not only on the inertia, damping and stiffness matrices but it is also influenced by the orientation of the desired frame. This is made evident by the closed-loop system resulting from the use of Euler angles:

$$\boldsymbol{\Lambda}_d\ddot{\tilde{\boldsymbol{\phi}}} + \boldsymbol{D}_d\dot{\tilde{\boldsymbol{\phi}}} + \boldsymbol{K}_d\tilde{\boldsymbol{\phi}} = \boldsymbol{T}^T(\boldsymbol{\phi})\,\boldsymbol{\mu}_{ext} \tag{2.75}$$

where $\boldsymbol{T}(\boldsymbol{\phi})$ is the transformation matrix that maps the Euler angles derivative to the angular velocity $\boldsymbol{\omega} = \boldsymbol{T}(\phi)\dot{\boldsymbol{\phi}}$ and $\boldsymbol{\mu}_{ext}$ is the external moment applied to the robot.

This could seem to be a negligible issue, however, $\boldsymbol{T}(\boldsymbol{\phi})$ is also suffering from representation

singularities leading to the exchange of huge moments with the environment [23].

2.5.2 Artificial potential fields

As introduced in 2.3, the dynamics of a manipulator can be modeled through the *Lagrangian equations*, namely,

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\boldsymbol{x}}}\right) - \frac{\partial L}{\partial \boldsymbol{x}} = \boldsymbol{F} \tag{2.76}$$

where the Lagrangian $L(\boldsymbol{x}, \dot{\boldsymbol{x}}) = T(\boldsymbol{x}, \dot{\boldsymbol{x}}) - U(\boldsymbol{x})$ is a measure of the energy balance of the system, being the difference between the total kinetic energy T and the potential energy V and $\boldsymbol{F}$ is a force acting on the system.

Here will be explained how potential fields can be easily exploited to enforce a desired motion to the robot. This theory was first introduced by Khatib in [24].

Notably, a force corresponds to the gradient of a potential field: $\boldsymbol{F} = -\nabla V$, furthermore, it is well-known that every passive system will try to place itself in a minimum energy configuration, therefore, we can try to apply a force corresponding to a potential field that has its minimum in the goal pose that our robot has to achieve, e.g. [24]

$$\boldsymbol{V}_{goal} = \frac{1}{2} k_p (\boldsymbol{x} - \boldsymbol{x}_d)^T (\boldsymbol{x} - \boldsymbol{x}_d) \tag{2.77}$$

Including a compensation of the gravity the Lagrangian equations assume the form:

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{\boldsymbol{x}}}\right) - \frac{\partial (T - V_{goal})}{\partial \boldsymbol{x}} = 0 \tag{2.78}$$

which is a conservative system stably oscillating around $\boldsymbol{x}_d$.

In order to damp oscillations, a dissipation has to be introduced, this corresponds to adding the term $\boldsymbol{F}_{diss} = -k_v \dot{\boldsymbol{x}}$ to the plugged force.

This approach can be extended to many applications (e.g. obstacle avoidance) and is particularly useful for redundancy resolution.

2.5.3 Redundancy resolution

Mobile manipulators have the characteristics of possessing a huge number of degrees of freedom, usually $n_{DOF} > 6$.

As a consequence, the system is redundant with respect to the Cartesian space, meaning that there are several ways to perform the same end-effector motion.

While this certainly opens the door to a great number of possibilities, it also introduces some problems such as the non-invertibility of the Jacobian and a more challenging treatment of

kinematic singularities [14]

In order to exploit this potential and overcome these problems, the null space of the Jacobian can be analyzed and manipulated.

In the following, the same topic will be examined from a dynamic perspective, thus leveraging the null space of $\boldsymbol{J}^{-T}$ instead of the one belonging to $\boldsymbol{J}$.

By definition, the null space of $\boldsymbol{J}^{-T}$ is the linear subspace of its domain that is mapped to zero [25]. Namely,

$$null(\boldsymbol{J}^{-T}) = \{\boldsymbol{\tau} \in \mathbb{R}^n : \boldsymbol{J}^{-T}\boldsymbol{\tau} = 0) \tag{2.79}$$

A practical interpretation of this definition coincides with saying that there is an infinite number of torques that result in no force generated at the end-effector.

A widely used method to act on the null space of the robot is by means of the *null space projection* [2]

$$\boldsymbol{N}(\boldsymbol{q}) = \boldsymbol{I} - \boldsymbol{J}^T(\boldsymbol{q})\overline{\boldsymbol{J}}^T(\boldsymbol{q}) \tag{2.80}$$

In fact, any torque $\boldsymbol{\tau}_0$ is projected in the null space of $\boldsymbol{J}^{-T}$ by simply premultiplying it by $\boldsymbol{N}(\boldsymbol{q})$, as a result, we can act on $\boldsymbol{\tau}_0$ to enforce desire useful internal motions that do not affect the end-effector [15].

Eventually, the general torque of a redundant manipulator is the sum of two contributes, one that affects the forces acting at the end-effector and one that is only responsible for internal motions [2]:

$$\boldsymbol{\tau} = \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{F} + [I - \boldsymbol{J}^T(\boldsymbol{q})\overline{\boldsymbol{J}}^T(\boldsymbol{q})]\boldsymbol{\tau}_0 \tag{2.81}$$

Dynamically consistent null space It is important to consider a dynamically consistent inverse in the computation of the projection, otherwise, force couplings will be generated at the end-effector level even by projected torques [2].

2.6 Safety in robotics

A core aspect of this thesis is the safe interaction between robots and humans. All the choices for developing the control architecture have been made by keeping this clearly in mind, this section focuses on what is considered a safe interaction and how safety can be tackled in the control.

[14] Kinematic singularities are configurations in which the Jacobian loses rank [1], meaning that 2 or more joint movements result in the same displacement of the end-effector. In non-redundant robots, these can be easily avoided through accurate trajectory planning

[15] As a matter of fact $\boldsymbol{F} = \overline{\boldsymbol{J}}^T(\boldsymbol{I} - \boldsymbol{J}^T(\boldsymbol{q})\overline{\boldsymbol{J}}^T(\boldsymbol{q}))\boldsymbol{\tau}_0 = (\boldsymbol{I} - \boldsymbol{I})\boldsymbol{\tau}_0 = 0$

2.6.1 Understanding what is a safe behaviour

Everyone has their own idea of what is safe and what is not, however, robotic applications require a clear definition (quantitative) on which to base control decisions. Particularly useful, in this sense, is [26] where the safe behaviour is defined starting from which injury a motion may cause. This, of course, depends on many factors that can be lessened to just 3 that embed all the interesting information:

1. m mass
2. v speed
3. shape of the hitting object

Several experiments have been conducted on pig skin to derive the relationship between the couple (m, v) and the injury caused for different shapes. The injury is classified following the AO classification [27] and, in particular, the IC2 level of damage (which means *contusion without skin opening*) is considered the worst allowed scenario for the human-robot interaction.

On the basis of the collected data and on the decided reasonable injury level, a safety curve (m, v) is derived through data fitting.

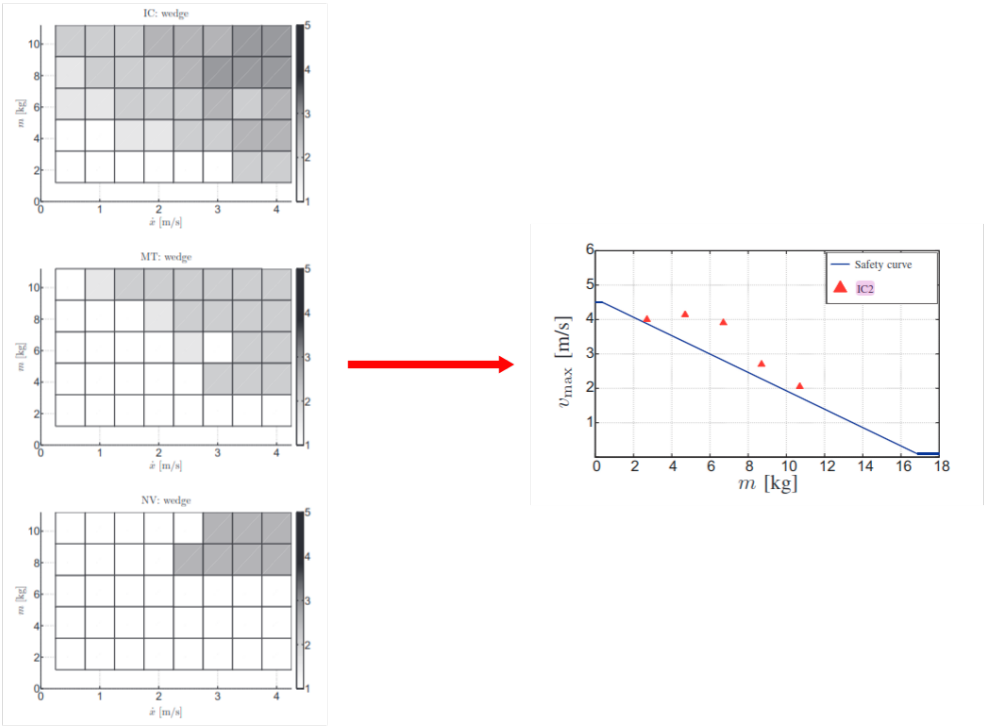


Figure 2.9: On the left: data collected related to a wedge-shaped tool, each graph represents different tissue damage: IC = skin damage, MT = muscle and tendon injury, NV = nerve and vessel injury. A darker square means higher damage.
On the right: Corresponding safe (m, v) curve for a wedge-shaped tool. Adapted from [26]

It is then sufficient to stay below this curve in order to avoid dangerous practices.

2.6.2 Attaining a safe behaviour: generalized Safe Motion Unit

In [26] is provided a simple yet effective algorithm for keeping the robot under the safety curve thus preventing any possible injury above IC2 level (under the name *Safe Motion Unit*), the algorithm was developed for a manipulator and then extended in [28] by Hamad et al. to tree-like robot structures (under the name *generalized Safe Motion Unit*).

Key concepts behind the algorithm The charts are derived using a point mass and a velocity in a specific direction, however, robots are far more complex systems, therefore their velocity has to be projected into a specific direction $\boldsymbol{u}$ and their mass matrix has to be reduced to a scalar by evaluating its contribute on $\boldsymbol{u}$, only in this way it is possible to get a meaningful comparison.

While for the velocity a simple scalar product with $\boldsymbol{u}$ is sufficient for obtaining the projection,

$$v_{\boldsymbol{u}} = \boldsymbol{v}^T \boldsymbol{u} \tag{2.82}$$

for the mass matrix, the concept of reflected mass has to be introduced

$$m_{\boldsymbol{u}}(\boldsymbol{q}) = \frac{1}{\boldsymbol{u}^T \boldsymbol{\Lambda}_{tr}^{-1}(\boldsymbol{q}) \boldsymbol{u}} \tag{2.83}$$

where the subscript tr stands for *translation*, because only the linear velocity is taken into account.

Now, considering that the robot is composed of multiple rigid bodies, it is worth evaluating not just one velocity but multiple velocities in order to be sure that any part of the robot could be considered safe. This is done by picking several points of interest (POIs) on the robot structure that represent particularly dangerous areas (e.g. edges, corners, tools...).

The location of the POI defines which dynamical and kinematic parameters should be used [28]. For the case of a mobile manipulator, the robot can be subdivided into 2 parts: arm and base.

For a POI located on the base, since the arm is on top of it, also the inertia coming from the arm should be considered, therefore

$$\boldsymbol{M}(\boldsymbol{q}) = \boldsymbol{M}_v(\boldsymbol{q}_v) + \boldsymbol{M}_{a,up}(\boldsymbol{q}_a)$$

Then, the translational part of the associated inertia matrix in Cartesian space is given by

$$\boldsymbol{\Lambda}(\boldsymbol{x})_{tr}^{-1} = \boldsymbol{J}_{POI,tr} \boldsymbol{M}^{-1} \boldsymbol{J}_{POI,tr}^T$$

where $\boldsymbol{J}_{POI} = \begin{bmatrix} \boldsymbol{I}_{3\times3} & -[{}^S\boldsymbol{p}_{POI}] \\ \boldsymbol{0}_{3\times3} & \boldsymbol{I}_{3\times3} \end{bmatrix} \boldsymbol{J}_v$ is the Jacobian evaluated at the position of the POI.

For a POI located on the arm, similar reasoning can be done, however, it should be taken into account that:

- The base is beneath the arm, meaning that if the base is not in motion only $\boldsymbol{M}_a$ should be considered in the computation of the inertia matrix
- If the arm is in motion while the base is fixed, $\boldsymbol{J}_{POI}$ is computed through $\boldsymbol{J}_a$ instead of $\boldsymbol{J}_v$.
- If both the arm and the base are in motion, $\boldsymbol{J}_{POI}$ is computed through the whole-body Jacobian $\boldsymbol{J}_b$ instead of $\boldsymbol{J}_v$.

By employing similar reasoning any possible combination can be taken into account. The table below summarizes them for the case of single-arm mobile manipulators.

POI location	Body in motion	$\boldsymbol{M}$	$\boldsymbol{J}_{POI}$
vehicle	vehicle	$\boldsymbol{M}_v + \boldsymbol{M}_{a,up}$	$\boldsymbol{J}_v$
vehicle	arm	No SMU	No SMU
vehicle	arm + vehicle	$\boldsymbol{M}_v + \boldsymbol{M}_{a,up}$	$\boldsymbol{J}_v$
arm	arm	$\boldsymbol{M}_a$	$\boldsymbol{J}_a$
arm	vehicle	$\boldsymbol{M}_a + \boldsymbol{M}_{a,up}$	$\boldsymbol{J}_v$
arm	arm + vehicle	$\boldsymbol{M}_a + \boldsymbol{M}_{a,up}$	$\boldsymbol{J}_b$

Table 2.1: Kinematics and Dynamics to be considered for attaining a safe velocity at POI location catalogued for any possible scenario on single-arm mobile manipulator

After having defined a list of POIs and their associated dynamics, every time instant the reflected mass at each POI is used to retrieve the corresponding safe velocity $\boldsymbol{v}_{safe,i}$ using the safety curves. The reflected mass is computed along $\boldsymbol{u}_i$ which is the direction in which a hit could cause more damage to the human (e.g. if a POI has been chosen on an edge, $\boldsymbol{u}_i$ is along the edge direction).

The smallest $\boldsymbol{v}_{safe,i}$ is taken as the most conservative.

This velocity, namely $\boldsymbol{v}_{safe}$, is then compared to the requested one at each POI, if the requested one is bigger, the joints have to be commanded to slow down until the velocity at the most conservative POI gets below $\boldsymbol{v}_{safe}$, otherwise the desired velocity remains unchanged.

A detailed description of the algorithm as it is implemented in this thesis can be found in 3.4.

3 Materials and methods

This chapter has the purpose to show the design implications maturated from the theory discussed in the state of the art and to provide interesting information about the hardware employed as an example of application. The discussion is generally valid for any omnidirectional wheeled manipulator.

3.1 RB-KAIROS+ dynamics and kinematics mode

Throughout this thesis, a custom version of the platform RB-KAIROS+ has been considered as possible hardware on which to test and verify its outcomes. In this section, it will be briefly described and then exploited as an example of how to derive the kinematic and dynamical model of a mobile manipulator.

3.1.1 Hardware overview

RB-KAIROS+ is a mobile manipulator provided by Robotnik and designed for collaborative manipulation in industrial environments. The unit at my disposal was a custom one that replaces the Universal Robots cobot with a Franka Emika Panda and matches the system in use in the European project DARKO (`https://darko-project.eu/`).

Figure 3.1: A render of the RB-Kairos+ mobile manipulator

The mobile base is equipped with four mechanum wheels that allow movements in any direction, a 3D lidar, a 3D camera and two 2D laser sensors for stopping the robot in the proximity of an obstacle.

The arm is a 7DOF lightweight cobot provided with torque sensors at each joint (all joints are revolute), specifically developed for research.

A central CPU is deputed of running all of these systems together by means of the Robot Operating System (ROS).

3.1.2 Model of the system

Leveraging the consideration made in sections 2.2, 2.3, and considering that most of the time, in practice, the low-level controllers of the wheels are masked to the user, the platform dynamics has been modelled as:

$$\begin{aligned}\begin{bmatrix} \boldsymbol{M}_v + \boldsymbol{M}_{a,up} & \boldsymbol{M}_{va} \\ \boldsymbol{M}_{va}^T & \boldsymbol{M}_a \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}}_v \\ \ddot{\boldsymbol{q}}_a \end{bmatrix} + \begin{bmatrix} \boldsymbol{C}_v & \boldsymbol{C}_{va} \\ \boldsymbol{C}_{va}^T & \boldsymbol{C}_a \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_v \\ \dot{\boldsymbol{q}}_a \end{bmatrix} + \begin{bmatrix} \boldsymbol{0}_{n_v \times 1} \\ \boldsymbol{g}_a \end{bmatrix} \\ = \begin{bmatrix} \boldsymbol{\tau}_v \\ \boldsymbol{\tau}_a \end{bmatrix} + \begin{bmatrix} \boldsymbol{\tau}_v^{ext} \\ \boldsymbol{\tau}_a^{ext} \end{bmatrix}\end{aligned} \tag{3.1}$$

where it can be noticed that the wheels' dynamics has been neglected.

The direct kinematics problem is solved through

$$ {}^S_E\boldsymbol{T}(\boldsymbol{q}) = {}^S_V\boldsymbol{T}(\boldsymbol{q}_v) {}^V_A\boldsymbol{T} {}^A_E\boldsymbol{T}(\boldsymbol{q}_a) \tag{3.2}$$

where

- $\{S\}$ is the fixed space frame.
- $\{V\}$ is the frame attached to the vehicle geometrical center
- $\{A\}$ is the frame at the base of the arm
- $\{E\}$ is the frame at the end-effector

While its whole-body geometric Jacobian is

$$ {}^S\boldsymbol{J} = \begin{bmatrix} \boldsymbol{V}_v(\boldsymbol{q}_a) {}^S\boldsymbol{J}_v & \boldsymbol{V}_a(\boldsymbol{q}_v) {}^S\boldsymbol{J}_a(\boldsymbol{q}_a) \end{bmatrix} \tag{3.3}$$

with $\boldsymbol{q}_v = [x_v \quad y_v \quad \phi_v] \in \mathbb{R}^3$ representing the two-dimensional pose of the vehicle and $\boldsymbol{q}_a \in \mathbb{R}^7$ representing the angular displacement of each joint belonging to the arm.

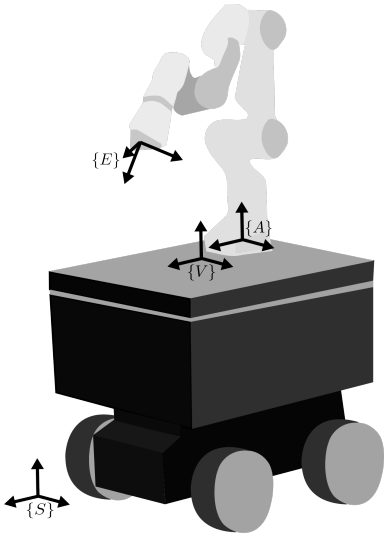


Figure 3.2: frame of references' definition

3.2 System Identification

To assign a value to each parameter of the system a proprietary software of TUM has been used. This software is based on the mathematical formulations introduced in sections 2.2, 2.3 and takes as input:

- The inertia tensors of the vehicle's chassis, the wheels and the arm joints.
- The centers of mass of the vehicle's chassis, the wheels and the arm joints.
- The mass of the vehicle's chassis, the wheels and the arm joints.
- The information about the geometry of the vehicle's chassis, the wheels and the arm (modified DH parameters).

and returns:

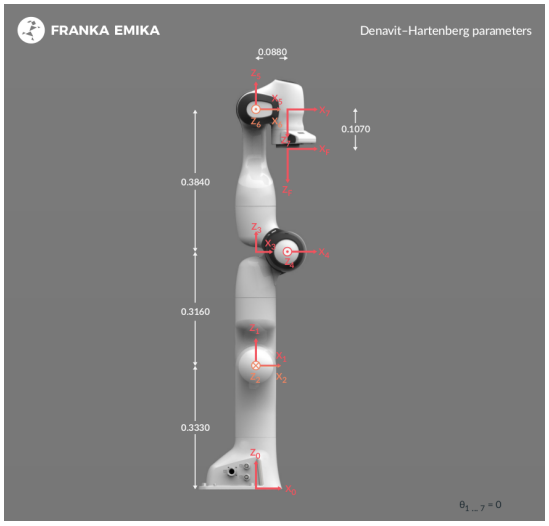
- The vehicle and arm's mass matrix.
- The vehicle and arm's Coriolis matrix.
- The arm's gravity vector
- the vehicle and arm's body Jacobian

The information about the geometry of the arm and the base has been taken from the manufacturer, here below two exemplification pictures are reported

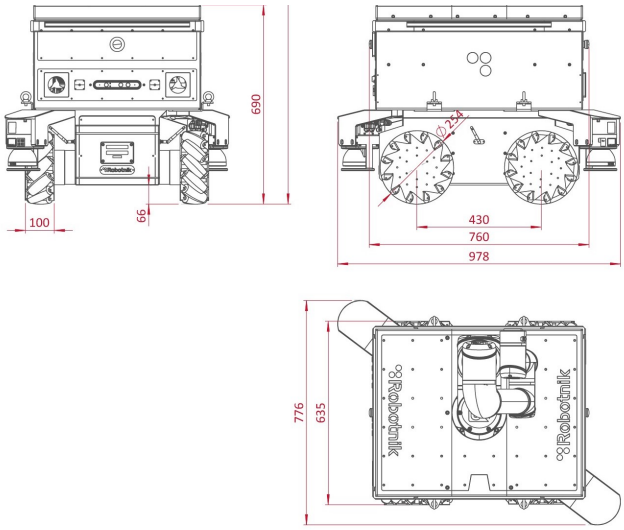
For the joints' inertia tensors, centers of mass and masses, a previous identification performed through the open-source software *FLOating BAse RObot dynamical IDentification* [1] has been used.

For the base's center of mass and total mass an experiment on account was performed.

[1] `https://github.com/kjyv/FloBaRoID`



(a) Picture of the arm (Franka Emika PANDA) used for defining the D-H parameters following Craig's convention



(b) RB-KAIROS+ mobile base geometry

3.2.1 Total mass and center of mass identification experimental procedure

Exploiting the theory leveraged in section 2.4 an experiment was performed using the poor instrumentation available in the laboratory. First, the arm was dismounted, then the experimental procedure was conducted in 2 phases, the first one allowed to determine the x and y coordinates of the center of mass (in this case the vehicle was placed parallel to the ground), the second one, instead, led to the finding of the z coordinate by inclining the robot.

The two phases are described in detail here below:

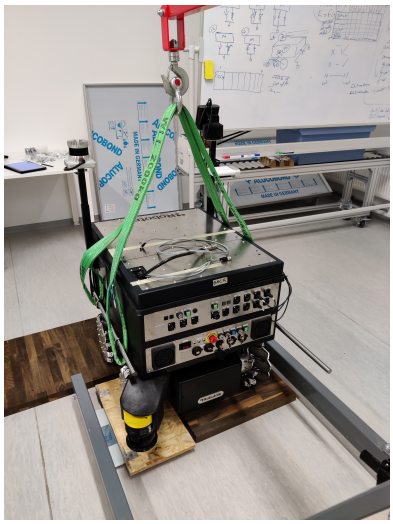
1. **Chassis parallel to the ground** - a single scale was placed below the first wheel, a measurement of the force exerted by the wheel has been taken, then the robot has been lifted and slowly released paying attention to reach the same configuration attained before. The experiment has been repeated in the same way for each of the 4 wheels considering that the elevation caused by the scale was matched on the other wheels by using some wedge made from wood.

   2 measurements for each wheel have been taken.

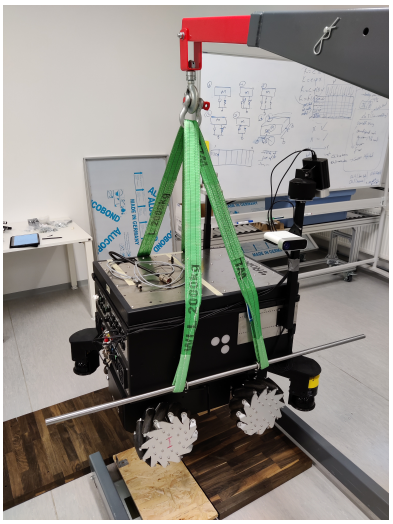
2. **Wheels inclined w.r.t the ground** - The first experimental phase has been repeated with the chassis subject to an inclination. Also in this case, the elevation has been matched and the robot has been maintained as fixed as possible in the same initial position. This was crucial for keeping always the same contact point on the scale.

   2 measurements for each wheel have been taken.

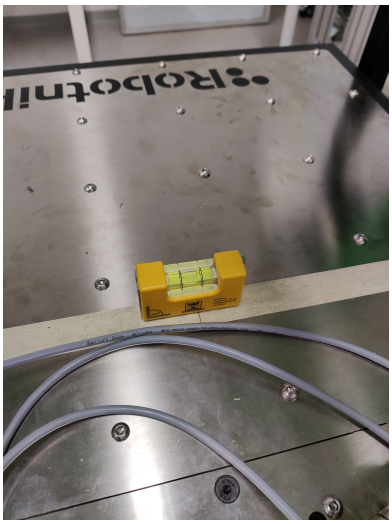
   It is worth noticing that this angle has a huge impact on the estimation of the z coordinate. A steeper inclination makes the experiment to bemore robust.



(a) Setting of the experiment, measurement of the force exerted by wheel 4

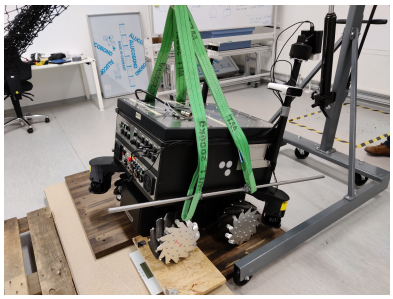


(b) Part in which the chassis has been elevated to in order to retake the measurement



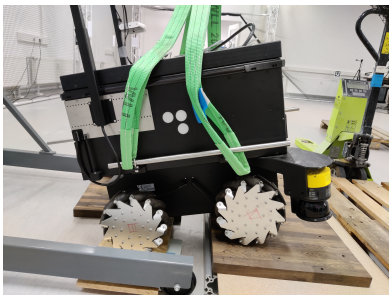
(c) Level placed on top of the chassis to ensure that it is parallel to the ground

Figure 3.4: Phase 1 (chassis parallel to the ground) pictures



(a) Measurement of the force exerted by wheel 1

(b) Measurement of the force exerted by wheel 2



(c) Side view, here is evident that the chassis is not touching the lift

Figure 3.5: Phase 2 (chassis inclined) pictures

Prior information and acquired data Here are briefly reported the available information on the wheel contact point position, the measurements of the contact forces captured during the experiment. For measuring weight forces, a scale has been employed, thus, the measurement unit reported below will be "Kg". This may seem in contrast with the theory depicted in the previous chapter, but it's not because weight and mass differ only by the gravity constant g.

wheel	d_{i_x} (m)	d_{i_y} (m)	d_{i_z} (m)
1	-0.215	-0.2675	-0.345
2	0.215	-0.2675	-0.345
3	0.215	0.2675	-0.345
4	-0.215	0.2675	-0.345

Table 3.1: Wheel contact points defined in the reference frame originating in the geometrical center of the chassis (see figure 2.7)

Measurements regarding the chassis parallel to the ground:

wheel	F_{i_1} (Kg)	F_{i_2} (Kg)
1	54.6	54.4
2	51.4	51.8
3	58.6	59.2
4	41.7	40.8

Table 3.2: Wheel parallels to the ground: measured weights. The indices 1 and 2 are used to distinguish between the first and second measurement

Measurements regarding the chassis inclined of α:

wheel	F_{i_1} (Kg)	F_{i_2} (Kg)
1	41.9	42.4
2	67.2	66.8
3	64.5	64.5
4	29.2	30

Table 3.3: Wheel parallels to the ground: measured weights. The indices 1 and 2 are used to distinguish between first and second measurement

In the end, the inclination angle was obtained as: $\alpha = sin^{-1}(\frac{90.8mm}{430mm})$ = 0.21276 rad = 12.1905°. where 90.8 mm is the elevation of the front wheels, measured with a vernier calibre, and 430mm is the distance between the contact points of the front and back wheels,

bar 1 weight	3.2 Kg
bar 2 weight	1.4 Kg

Table 3.4: Weights of metal bars used to move and hang the robot

provided by the manufacturer. Such a small height makes this procedure prone to very big uncertainties, which can't be estimated since they depend on the quality of the height measurement. However, this is the best that could have been done without making the chassis' bottom touch the wedge.

3.3 Whole-body controller

Here a whole-body control framework for safe mobile manipulation is presented. The main intent of the framework is to make the robot follow a trajectory while attaining a compliant and safe behaviour. Safety, is handled at trajectory planning level: at any moment, if a human is detected in the vicinity, the control input is recomputed to tackle a slower motion.

The robot is given two tasks, which will be called *primary task* and *secondary task*. The primary task involves the accomplishment of the desired motion at the end-effector while showing a desired impedance, the secondary task, instead, exploits internal motions to perform coordinated arm base motions. The way in which the secondary task is formulated also allows to avoid singularities, joint limits and excessive extensions of the arm.

The torques generated from the two tasks are then added together and sent to the robot actuators, the null-space projection applied in $\boldsymbol{tau}_n$ ensures that the two tasks do not overlap in any sense.

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{imp} + \boldsymbol{\tau}_n \tag{3.4}$$

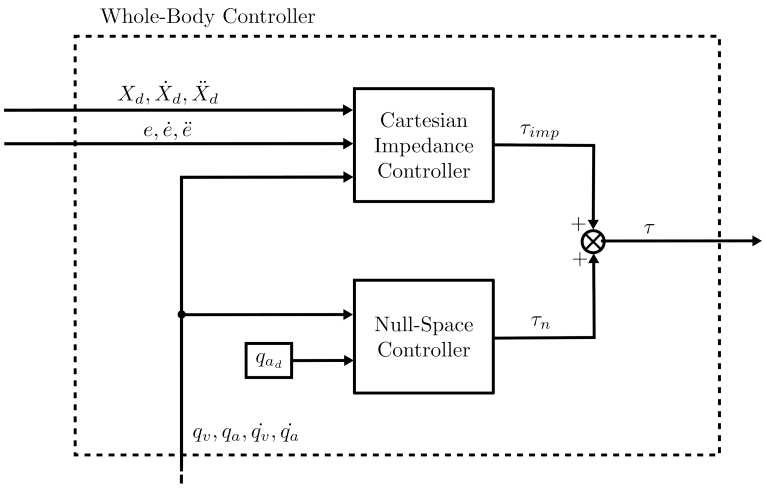


Figure 3.6: Whole-body control schematic

While the arm directly supports a torque signal, the mobile base is usually controlled in velocity, therefore the torque elements corresponding to the base are firstly passed to an admittance interface that generates a velocity coherent with the impedance to be attained.

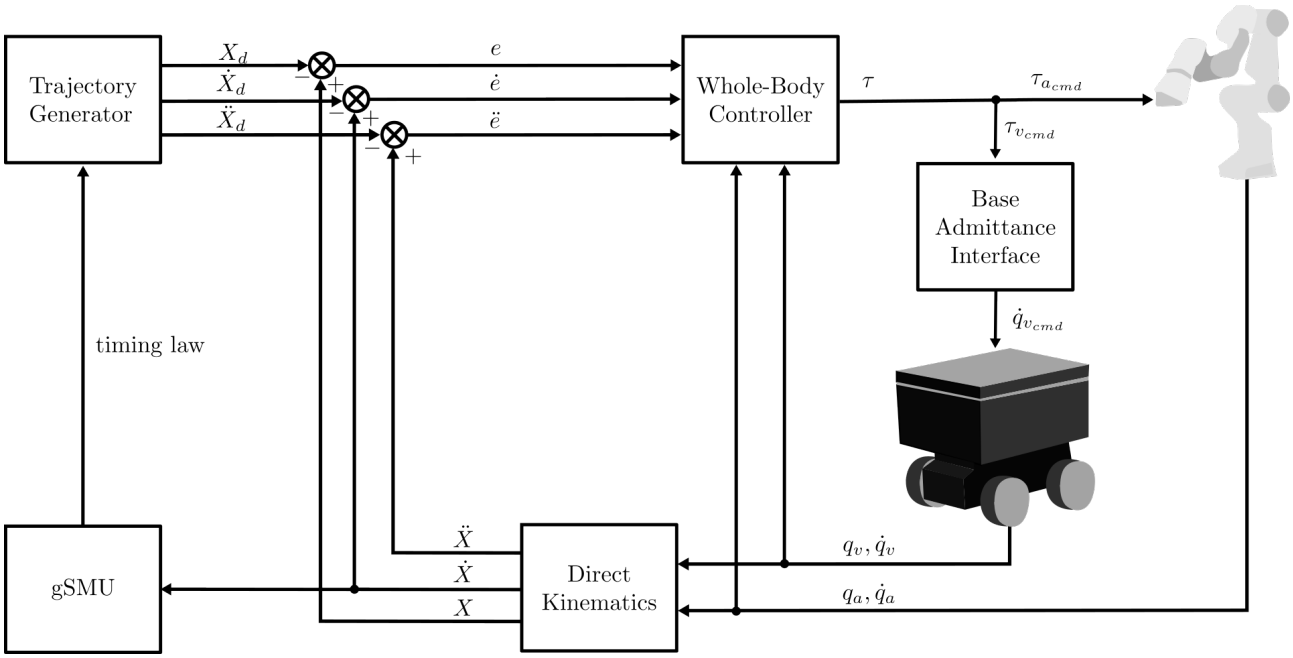


Figure 3.7: Control framework schematic

3.3.1 Primary task

To follow a desired trajectory while attaining a specified impedance a classical Cartesian space impedance controller with gravity compensation has been adopted, but before introducing the control law, it is necessary to specify how the pose is represented and how the control error is defined.

For the advantages explained in 2.1.1 and due to their property of non-affecting the dynamic behaviour of the rotational part [23], quaternions have been favoured over the other representations for the orientation of the end-effector. Therefore, the pose has the same shape as 2.15.

Leveraging the result given by 2.1.1 and choosing the desired frame D as the frame of reference in which the principal axes of the rotational stiffness remain constant (see 2.5.1) the control error has been defined as,

$$\tilde{\boldsymbol{x}} = \begin{bmatrix} \boldsymbol{e}_t \\ \boldsymbol{e}_o \end{bmatrix} = \begin{bmatrix} {}^S_D\boldsymbol{R}^T({}^S\boldsymbol{p}_E - {}^S\boldsymbol{p}_D) \\ 2(\eta_{ds} * \boldsymbol{\epsilon}_{es} - \eta_{es} * \boldsymbol{\epsilon}_{ds} - \boldsymbol{\epsilon}_{ds} \times \boldsymbol{\epsilon}_{es})) \end{bmatrix} = \begin{bmatrix} {}^D\boldsymbol{p}_{ed} \\ 2\boldsymbol{\epsilon}_{ed} \end{bmatrix} \tag{3.5}$$

Differentiating ${}^D\boldsymbol{p}_{ed}$ with respect to time [2] and approximating the quaternion derivative with

[2] it is useful to recall the derivative of the rotation matrix: ${}^S_D\dot{\boldsymbol{R}}^T = \boldsymbol{S}({}^D\boldsymbol{\omega}_D){}^S_D\boldsymbol{R}^T$ and the property $S(\boldsymbol{R}\boldsymbol{\omega}) = \boldsymbol{R}\boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{R}^T$[1]

the angular velocity (as explained in 2.1.1), the velocity error $\dot{\tilde{\boldsymbol{x}}}$ is obtained

$$\dot{\tilde{\boldsymbol{x}}} = \begin{bmatrix} {}^{S}_{D}\boldsymbol{R}^{T}(\dot{\boldsymbol{p}}_E - \dot{\boldsymbol{p}}_D) - {}^{S}_{D}\boldsymbol{R}^{T}\boldsymbol{S}(\boldsymbol{\omega}_D)(\boldsymbol{p}_E - \boldsymbol{p}_D) \\ {}^{S}_{D}\boldsymbol{R}^{T}(\boldsymbol{\omega}_E - \boldsymbol{\omega}_D) \end{bmatrix} \tag{3.6}$$

Looking at these two errors and considering the discussion made in 2.5.1, it is evident that

$$\boldsymbol{J}_x\dot{\boldsymbol{q}} = \begin{bmatrix} {}^{S}_{D}\boldsymbol{R}^{T} & \boldsymbol{0}_{3x3} \\ \boldsymbol{0}_{3x3} & {}^{S}_{D}\boldsymbol{R}^{T} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{p}}_E \\ \boldsymbol{\omega}_E \end{bmatrix} = \begin{bmatrix} {}^{S}_{D}\boldsymbol{R}^{T} & \boldsymbol{0}_{3x3} \\ \boldsymbol{0}_{3x3} & {}^{S}_{D}\boldsymbol{R}^{T} \end{bmatrix} \begin{bmatrix} \boldsymbol{J}_t \\ \boldsymbol{J}_o \end{bmatrix} \dot{\boldsymbol{q}} \tag{3.7}$$

$$\boldsymbol{v}_t = \begin{bmatrix} {}^{S}_{D}\boldsymbol{R}^{T} & \boldsymbol{0}_{3x3} \\ \boldsymbol{0}_{3x3} & {}^{S}_{D}\boldsymbol{R}^{T} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{p}}_D + \boldsymbol{S}(\boldsymbol{\omega}_D)(\boldsymbol{p}_E - \boldsymbol{p}_D) \\ \boldsymbol{\omega}_D \end{bmatrix} \tag{3.8}$$

$$\dot{\boldsymbol{v}}_t = \begin{bmatrix} {}^{S}_{D}\boldsymbol{R}^{T} & \boldsymbol{0}_{3x3} \\ \boldsymbol{0}_{3x3} & {}^{S}_{D}\boldsymbol{R}^{T} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{p}}_D + \boldsymbol{S}(\boldsymbol{\alpha}_D)(\boldsymbol{p}_E - \boldsymbol{p}_D) + \boldsymbol{S}(\boldsymbol{\omega}_D)(\dot{\boldsymbol{p}}_E - \dot{\boldsymbol{p}}_D) - \boldsymbol{S}(\boldsymbol{\omega}_D)\boldsymbol{a} \\ \boldsymbol{\alpha}_D - \boldsymbol{S}(\boldsymbol{\omega}_D)\boldsymbol{\omega}_D \end{bmatrix} \tag{3.9}$$

where $\boldsymbol{J}_t$ and $\boldsymbol{J}_o$ are, respectively, the translational and the rotational part of the body Jacobian and $\boldsymbol{a} = \dot{\boldsymbol{p}}_D + \boldsymbol{S}(\boldsymbol{\omega}_D)(\boldsymbol{p}_E - \boldsymbol{p}_D)$.

In conclusion, the classical impedance controller is reformulated as,

$$\boldsymbol{\tau}_{imp} = \boldsymbol{J}_x(\boldsymbol{q},t)^T\boldsymbol{F}_{imp} = \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{J}_x(\boldsymbol{q},t)^T\left(-\boldsymbol{\Lambda}(\boldsymbol{q},t)\dot{\boldsymbol{v}}_t(\boldsymbol{q},t) - \boldsymbol{\mu}(\boldsymbol{q},\dot{\boldsymbol{q}},t)\boldsymbol{v}_t(\boldsymbol{q},t) - \boldsymbol{K}_d\tilde{\boldsymbol{x}} - \boldsymbol{D}_d(\boldsymbol{q},t)\dot{\tilde{\boldsymbol{x}}}\right)$$

by means of this torque, the closed-loop system assumes the form

$$\boldsymbol{\Lambda}(\boldsymbol{q},t)\ddot{\tilde{\boldsymbol{x}}} + (\boldsymbol{D}_d(\boldsymbol{q},t) + \boldsymbol{\mu}(\boldsymbol{q},t))\dot{\tilde{\boldsymbol{x}}} + \boldsymbol{K}_d\tilde{\boldsymbol{x}} = \boldsymbol{F}_{ext} \tag{3.10}$$

The lack of subscript in $\boldsymbol{\Lambda}$ indicates that no inertia-shaping has been applied as a consequence of the absence of a force sensor at the end-effector level. $\boldsymbol{D}_d$, is determined by means of the factorization design technique because practically more robust. We recall, here, that $\boldsymbol{\mu}$ has not been compensated to ensure that passivity property of the system dynamics is respected. Since the only compensation applied is the one for gravity, the resulting system is non-linear.

A note on the effect of antipodal quaternions on control error Quaternions are well-known for their 2-to-1 covering of the 3D rotation group $SO(3)$. This means that $(\eta, \boldsymbol{\epsilon}) = (cos(\frac{\theta}{2}), \boldsymbol{u}sin(\frac{\theta}{2}))$ and $(-\eta, -\boldsymbol{\epsilon}) = (cos(\frac{\theta+2\pi}{2}), \boldsymbol{u}sin(\frac{\theta+2\pi}{2})) = -(cos(\frac{\theta}{2}), -\boldsymbol{u}sin(\frac{\theta}{2}))$ represent the same orientation [3]. While this is certainly true, the two quaternions result in different rotations. In particular, considering the quaternion error, a negative sign in the scalar part results in a rotation that is π radiants longer, thus compromising the performances of the controller. For this reason, in the control implementation, a check on the sign of η is performed to ensure that the shorter possible rotation is always followed.

3.3.2 Secondary task

To act on internal motion, inspiration has been taken from [29]. Firstly a potential $\boldsymbol{V}$ to be minimized is defined, then its gradient is computed and the resulting torque is projected in the null space of the mobile manipulator to decouple it from the primary task:

$$\boldsymbol{\tau}_n = (\boldsymbol{I} - \boldsymbol{J}^T\overline{\boldsymbol{J}}^T)\boldsymbol{\tau}_0 = -(\boldsymbol{I} - \boldsymbol{J}^T\overline{\boldsymbol{J}}^T)\boldsymbol{M}(\boldsymbol{q})\nabla\boldsymbol{V} \tag{3.11}$$

where the mass matrix $\boldsymbol{M}(\boldsymbol{q})$ has been included to provided a dynamically consistent torque.

Russakow and Khatib provide an interesting artificial potential field for coordinated motion between an arm and a floating base (namely a drone). Since a floating base and an omnidirectional vehicle share a lot of similarities, this approach can be directly applied to our case.

The developed $\boldsymbol{V}$ exploits the redundant degrees of freedom of the base to keep a certain configuration of the arm.

Its gradient coincides with

$$\nabla\boldsymbol{V} = \begin{bmatrix} k_{v,x}\dot{x}_v \\ k_{v,y}\dot{y}_v \\ k_{v,\theta}\dot{\theta}_v \\ k_{p,a1}(q_{a,1} - q_{a,1d}) + k_{v,a1}(\dot{q}_{a,1}) \\ k_{p,a2}(q_{a,2} - q_{a,2d}) + k_{v,a2}(\dot{q}_{a,2}) \\ k_{p,a3}(q_{a,3} - q_{a,3d}) + k_{v,a3}(\dot{q}_{a,3}) \\ k_{p,a4}(q_{a,4} - q_{a,4d}) + k_{v,a4}(\dot{q}_{a,4}) \\ k_{p,a5}(q_{a,5} - q_{a,5d}) + k_{v,a5}(\dot{q}_{a,5}) \\ k_{p,a6}(q_{a,6} - q_{a,6d}) + k_{v,a6}(\dot{q}_{a,6}) \\ k_{p,a7}(q_{a,7} - q_{a,7d}) + k_{v,a7}(\dot{q}_{a,7}) \end{bmatrix} \tag{3.12}$$

where $\boldsymbol{q}_{a,d}$ is the arm configuration to be maintained, $k_{p,ai}$ are the gains for the configuration error and the terms multiplied by $k_{v,...}$ have been introduced to enforce stability via dissipation.

Thanks to this potential, the base will be activated to minimize the difference between the actual and the desired configuration. An accurate choice of $\boldsymbol{q}_{a,d}$ makes the robot try to stay away from singularities, joint limits and can be also used to increase its manipulability.

3.3.3 Admittance interface

Compliance control require to act on the system dynamics and this is accomplished by sending torques to the robot's joints. Nevertheless, most of the times, mobile bases are not provided of a torque interface, instead, they dispose of a velocity/position interface. As a consequence, the impedance control law can't be directly sent to the base actuators and has to be transformed in an equivalent velocity input signal that the wheels have to follow. The proposed admittance interface will make use of the dynamical model without changing it, in order to keep the fidelity with the hardware dynamics. Others, such as [9], when specifying the admittance interface, prefer to impose a different dynamics which is linear, decoupled and of magnitude similar to the other subsystems of the robot, however this leads to addressing the new parameters in a specific way and to compensating the non-linearities in order to avoid instabilities.

Since no force sensor is available on the hardware the external forces acting on the robot will be treated as disturbances, the high gain of the velocity controller that acts on the wheels is supposed to perform a perfect rejection.

Recalling 3.13, the velocity input can be obtained as the integral of

$$\ddot{\boldsymbol{q}}_{v,cmd}(t) = (\boldsymbol{M}_v(t) + \boldsymbol{M}_{a,up}(t))^{-1}(\boldsymbol{\tau}_v(t) - \boldsymbol{C}_v(t)\dot{\boldsymbol{q}}_v(t)) \tag{3.13}$$

Here the time was made explicit to stress that the acceleration command is obtained through terms belonging to the current time instant and it corresponds to the acceleration induced by $\boldsymbol{\tau}_v$ in free motion.

3.4 Safety implementation

Having in mind the preliminary knowledge discussed in 2.6.2, it is possible to present the algorithm:

Algorithm 1 generalized Safe Motion Unit (*Adapted from [28]*)

Inputs: Curvatures, inertial data, desired base/arm EE velocity
Resources: INJURY database with encoded safety curve

1: Define a set of POIs (points of interest) $\{POI_1 \dots POI_n\}$: By specifying their position $\boldsymbol{p}_{POI_i}$ and the direction $\boldsymbol{u}_i$ in which they are more harmful.
2: Compute $\boldsymbol{v}_{POI_i}$: Obtain the desired velocity of each POI from the end-effector desired velocity by applying the adjoint representation $[\boldsymbol{Ad}_T]$ corresponding to the transformation $\boldsymbol{T} = (\boldsymbol{I}, \boldsymbol{p}_{POI_i})$.
3: Evaluate the inverse of the operational space kinetic energy matrix for the coupled mobile manipulator system at the base/arm POIs using the POI Jacobian, see 2.6.2.
4: Calculate the reflected mass of the mobile manipulator at each POI in the most dangerous direction $\boldsymbol{u}_i$ through 2.83.
5: Evaluate the maximum permissible velocity that is safe $v_{max_i}^{safe}$ for the corresponding reflected mass of each POI ($m_{u_{POI_i}}$) and its curvature primitive from the corresponding safety curve after querying the INJURY database.
6: select the most conservative v_{max}^{safe} (lowest $v_{max_i}^{safe}$ among all the POIs).
7: **if** $|\boldsymbol{v}_{POI} * \boldsymbol{u}| > v_{POI}^{safe}$ **then**
8: Scale v_{max}^{safe} with the projection of POI velocity in $\boldsymbol{u}$ direction to get the new human-safe POI velocity v_{POI}^{safe} (if $\boldsymbol{v}_{POI}$ along $\boldsymbol{u}$ is lower than v_{POI}^{safe}, v_{POI}^{safe} = $\boldsymbol{v}_{POI}$ and no change is applied to the end-effector velocity).
9: Calculate the new, safe velocity of the mobile manipulator's end-effector v_e^{safe} from v_{POI}^{safe} through $[\boldsymbol{Ad}_T]$ with $\boldsymbol{T} = (\boldsymbol{I}, -\boldsymbol{p}_{POI_i})$.
10: **else**
11: the end-effector desired velocity remains unchanged
12: **end if**

This algorithm is executed at each sample time and then, if needed, a trajectory planner which is proprietary to TUM is used to slow down the robot until it achieves the new desired velocity.

Motion planning is not a topic addressed by this thesis but to have a general idea, the planner in use is able to generate trajectories to tackle point-to-point motion via a trapezoidal velocity profile. The peculiarity of the planner in use is its capability of relaxing the time law thus leading to a slower motion while maintaining the same desired path.

3.5 Software architecture

The control framework has been developed in MATLAB and then ported to ROS (Robotic Operating System). C++ has been chosen for the porting since it is compatible with ROS, it is computationally efficient and it is object-oriented.

3.5.1 Kinematics and dynamics

Each matrix that is involved in the kinematics and dynamics of the robot has been transposed into a function, by means of a string it is possible to select the matrix corresponding to the desired part of the robot (mobile base, manipulator, whole-body). Since the information about the dynamics and kinematics of the robot has to be accessible in different time instants and by different nodes, the model has been designed as a library.

3.5.2 Safety

Safety is implemented as a ROS service. The server can be interrogated with a request, specifying:

- the body associated with the POI
- the position of the POI in $\{E\}$
- the most dangerous direction $\boldsymbol{u}$
- the configuration variables attained by the arm and the vehicle at the moment of the request
- the desired velocity to be attained by the end effector

and responds with the velocity of the end-effector which ensures a safe behaviour. The human position is retrieved from a topic that makes use of the *SPENCER Multi-Modal People Detection & Tracking Framework*[3] and consider it in the computation of the safe velocity.

Currently, the safety has been developed for only one POI, but can be easily extended to a list of POIs by adapting the request format and adding a loop in the server that computes the safe velocity for each POI and considers the smallest one for the comparison with the current velocity of the robot.

[3] see `https://github.com/spencer-project/spencer_people_tracking`

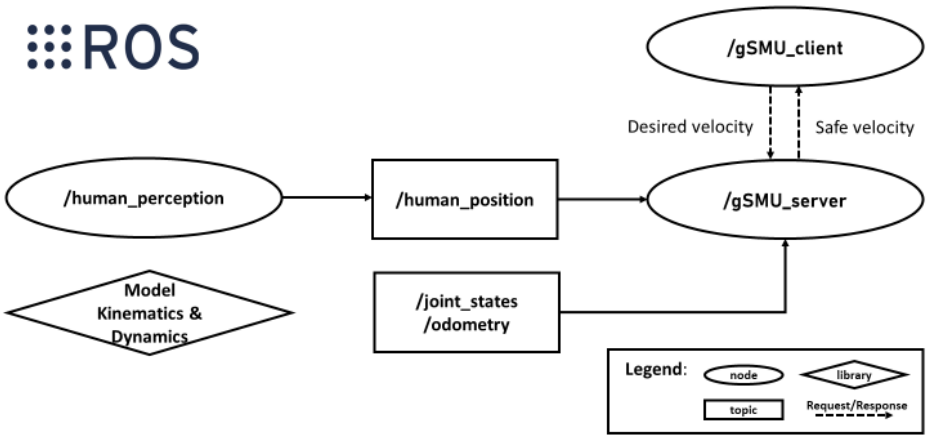


Figure 3.8: Safety implementation in ROS

3.5.3 Whole-body controller

The whole-body controller node embeds the planning and the control law. A class has been developed to read the data coming from the sensors and elaborate on them, compute the current dynamics and kinematics, specify the control parameters and finally determine the control inputs. The class is instantiated inside the node and by calling its constructor it is possible to specify the body to be controlled (mobile base, manipulator, whole-body). At the moment this feature is useful only for testing, but in future can be useful if a hierarchical control technique is adopted.

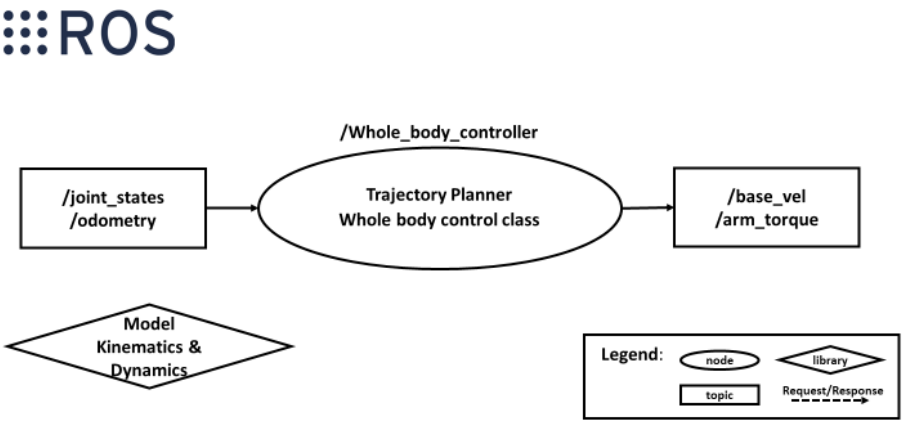


Figure 3.9: Whole body control implementation in ROS

4 Simulations

The control framework and the robot model have been tested on MATLAB/Simulink and subsequently in ROS/Gazebo. Several simulations have been conducted to understand the behaviour of the closed-loop system.

Firstly the impedance control has been simplified and tested on the base only in Simulink (in this case the redundancy resolution is not needed and eliminated). Secondly the same has been done for the arm and, in the end, the two systems have been tested together through the whole-body controller. A visual animation has also been provided to better understand the motions entailed by the robot.

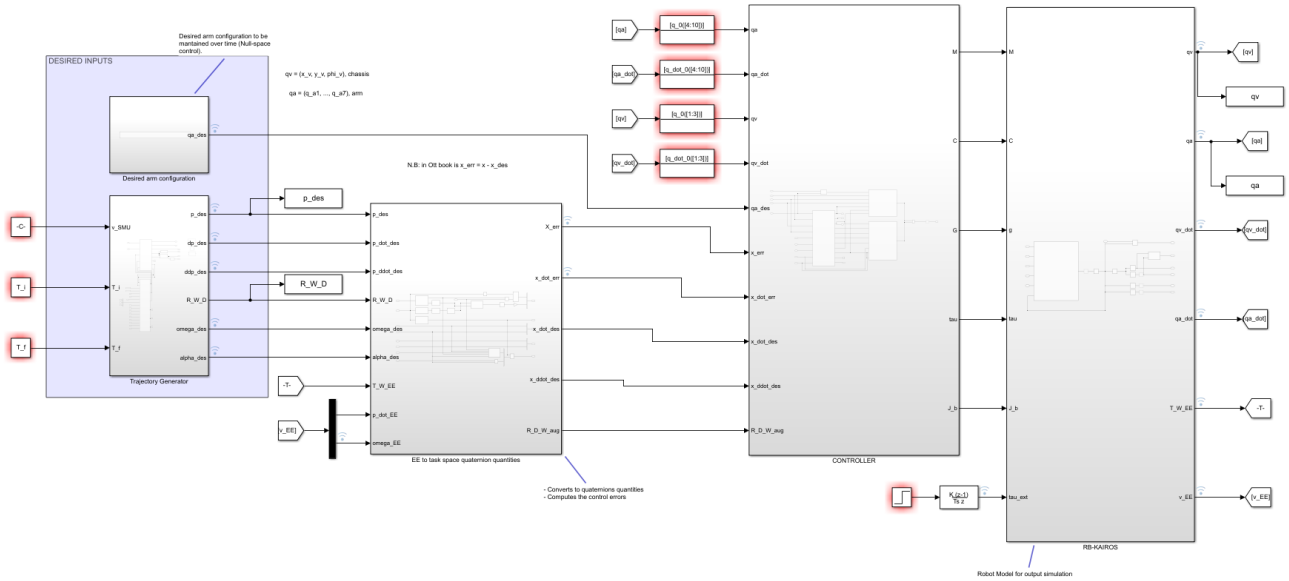


Figure 4.1: A screenshot of the whole-body control framework in Simulink

After having evaluated the proper functioning of the control framework in MATLAB a huge amount of time has been invested in the ROS/Gazebo simulation because it represents the closest thing to the software implementation on the real hardware. Gazebo is an open-source simulator also suitable for simulating dynamics that can be easily interconnected with ROS. A simulation for the RB-KAIROS+ robot was already available and provided by DARKO project, however, it was designed just for observing the motions of the robot and not to simulate its dynamics, which is very undesirable for testing an impedance controller. Therefore, the urdf models have been revised to match the ones derived during this thesis. Furthermore, the digital twin of the robot was subject to small and fast oscillations, resulting in very disturbed measurements of the joints' configurations and of the base odometry. A cause of this behaviour has been found in the low-level controllers that ROS implements to simulate the actuators' dynamics. By changing the PID parameters related to these controllers and

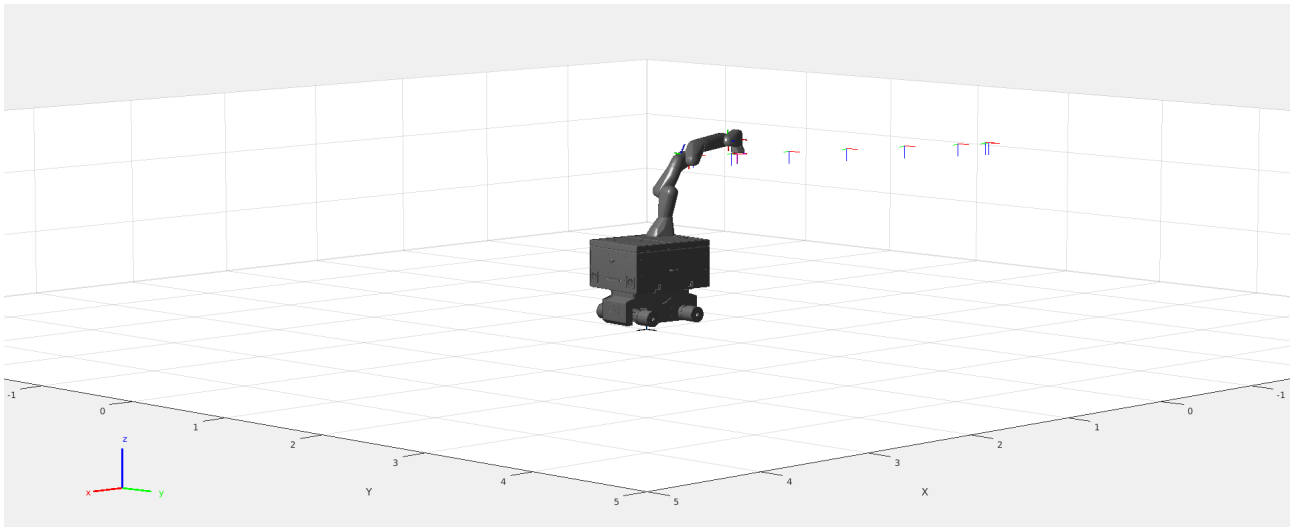


Figure 4.2: Robot's motions visualization in MATLAB

tuning the parameters that characterize the physics engine a more stable simulation has been obtained. Further complications were caused by the fact that Gazebo runs in a Docker container to match the ROS version available in RB-KAIROS+ and by the complexity of the overall simulation that involves several packages from DARKO and the manufacturers. Nevertheless, eventually, it has been possible to test the C++ code also in Gazebo, the results will be discussed in the next section.

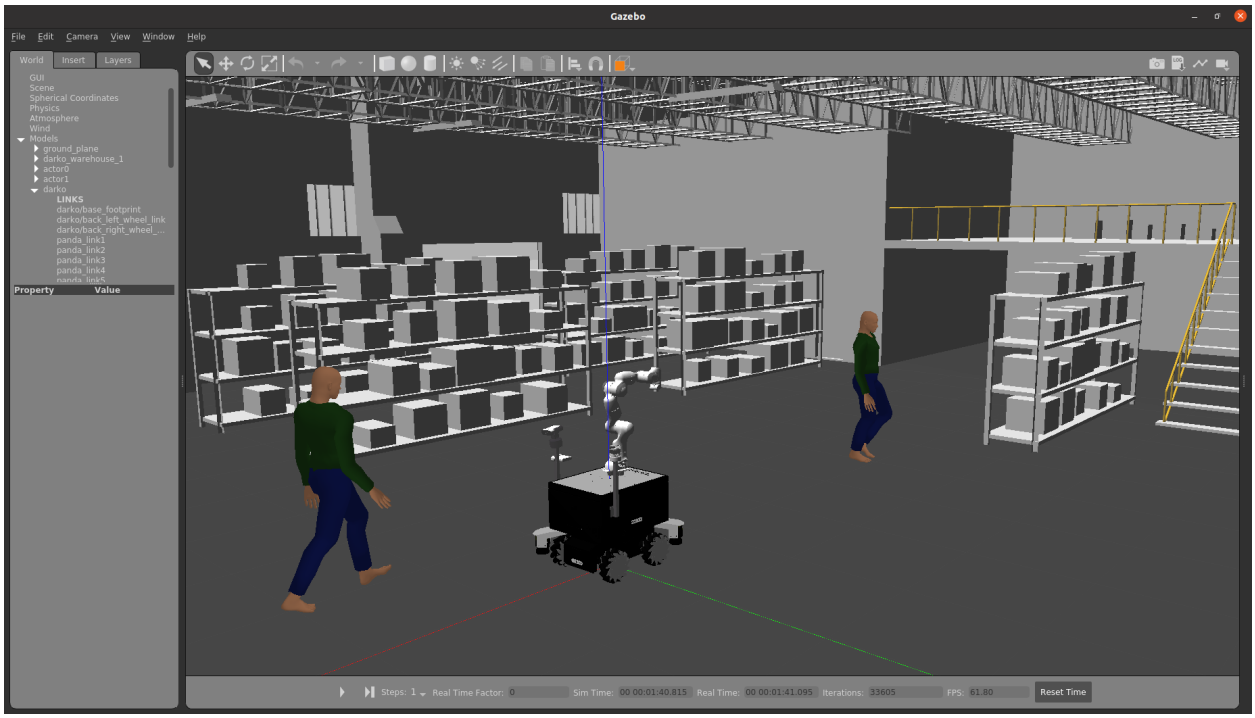


Figure 4.3: Gazebo simulation environment

5 Results and Discussions

5.1 Center of mass and total mass identification experiment

The measurements collected during the experiment are elaborated here and presented as results.

5.1.1 Partial results

The subsequent values are obtained directly applying equations 2.56, 2.57 and 2.58 on the data coming from the experiment, without subtracting the metal bars center of mass and do not include uncertainty.

$\hat{d}_{COM_x}$	-0.01537576 m
$\hat{d}_{COM_y}$	0.007716970 m
$\hat{d}_{COM_z}$	0.08113930 m
$\hat{m}_{chassis}$	206.25 Kg

Table 5.1: Center of mass coordinates with respect to the reference frame defined in the geometrical center and total mass of the chassis.

5.1.2 Error Analysis

For a complete understanding of the quality of the experiment, it is necessary to perform an error analysis. Here are reported all the sources of disturbances that lead to quantitative uncertainty in the measurements.

- The scale is a bathroom scale (Ideen Welt PT-738). There is no data provided about its precision but testing it several times always gives the same results, also employing different weights. For what comes to accuracy, using known weights shows values coherent with them, the difference is only due to the sensitivity of the instrument which is 0.01 Kg. So, the data coming from the scale will be considered correct up to a delta of ±0.01 kg.

- Weight measurements have shown a maximum shift w.r.t. the expected value of ±0.45Kg. To be conservative, this measure will be considered for the overall error calculation.
- The elevation has been measured with a calibre which has a resolution of 0.05 mm. Another source of error, since mechanum wheels have not a perfectly rounded surface, comes from a possible wheel displacement that would change the distance between contact points. However, this error is considered negligible because the motors are particularly stiff and notable attention has been put to avoid undesired rotations.
- The two bars necessary to hang the robot introduce a deviation to the results. Thus their weight must be subtracted from the total weight measurement and their centers of masses should be included in the overall center of mass calculus. For what regards their weight, it is subject only to the resolution error of the scale.

These considerations have then to be collocated inside the formulas that were used to determine the center of mass location in order to have an estimation of the overall uncertainty.

Firstly, we analyse the estimation of the total weight of the chassis. The source of error in this case comes from the scale resolution and from the deviation from the expected value. According to equation 2.58 the calculus involves 4 sums, other than this, the mass of the bars should be subtracted, therefore, in total 6 sums are involved. As it is well-know [30], the overall error corresponds to the sum of each error which is:

$$\Delta_{m_{chassis}} = 4 * 0.45 + 4 * 0.1 + 2 * 0.1Kg = 1.86Kg \tag{5.1}$$

Then, coming to the x and y coordinates of the Center of Mass, equation 2.56 shows that it involves $\frac{1}{W}$, whose error is $\Delta_W = \Delta_{m_{chassis}} - 2 * 0.1$ [1] and the sum of products between F_{i_z} and d_{i_x} or d_{i_y}.

The error of a product is summarized as [30] $\frac{\Delta_c}{c} = \frac{\Delta_a}{a} + \frac{\Delta_b}{b}$, where b and a are the quantities involved in the product, Δ_b and Δ_a are their respective errors and $c = a * b$.

Thus, the partial error on the coordinates x and y is:

$$\frac{\Delta_{prod_x}}{prod_x} = \sum_i \frac{0.45 + 0.01}{F_{i_z}} + \sum_i \frac{0}{d_{i_x}} = 0.008921 \tag{5.2}$$

$$\frac{\Delta_{prod_y}}{prod_y} = \sum_i \frac{0.45 + 0.01}{F_{i_z}} + \sum_i \frac{0}{d_{i_y}} = 0.008921 \tag{5.3}$$

This result combined with the error coming from $\frac{1}{W}$ gives:

[1] because in the calculus of the COM the bars need to be initially included, then their COMs will be subtracted to get the true COM value of the chassis

$$\frac{\Delta_{d_{COM_x}}}{d_{COM_x}} = \frac{\Delta_{prod_x}}{prod_x} + \frac{\Delta_W}{W} = 0.008921 + 0.008921 = 1.7842\% \tag{5.4}$$

$$\frac{\Delta_{d_{COM_y}}}{d_{COM_x}} = \frac{\Delta_{prod_y}}{prod_y} + \frac{\Delta_W}{W} = 0.008921 + 0.008921 = 1.7842\% \tag{5.5}$$

Finally, it remains to compute the uncertainty on d_{COM_z}. This quantity is described by 2.57 and its indirect dependency on α requires to introduce the error propagation in sine and cosine functions. In [30] is stated that the error deriving from applying a arcsine to an uncertain angle is given by: $\Delta_\alpha = \Delta_{asin(x)} = \frac{1}{\sqrt{1-x^2}} * \Delta_x$

The measurement on x is subject to an error of 0.05e-3 m, therefore;

$$\Delta_\alpha = \frac{1}{\sqrt{1 - (\frac{90.8}{430})^2}} * \Delta_x = 5.1153e-05 \quad rad \tag{5.6}$$

Considered the small value obtained, the error on α will be neglected.

The other terms involved, are already known, in particular, there are 3 terms involving $\frac{\Delta_W}{W}$, 2 terms of the kind $\frac{\Delta_{prod}}{prod}$ and one term $\frac{\Delta_{d_{COM_x}}}{d_{COM_x}}$.

$$\frac{\Delta_{d_{COM_z}}}{d_{COM_z}} = 3*\frac{\Delta_W}{W} + 2*\frac{\Delta_{prod}}{prod} + \frac{\Delta_{d_{COM_x}}}{d_{COM_x}} = 4*0.8921\% + 2*0.8921\% + 1.7842\% = 7.1368\% \tag{5.7}$$

5.1.3 Final results

The last thing to be done for obtaining a complete description of the center of mass of the chassis is to remove the metal bars' contribute.

$$d_{COM_x} = \frac{\hat{d}_{COM_x} * \hat{m}_{chassis} - d_{COM_x1} * m_{bar1} - d_{COM_x2} * m_{bar2}}{\hat{m}_{chassis}} = -0.0249272751m \tag{5.8}$$

$$d_{COM_y} = \frac{\hat{d}_{COM_y} * \hat{m}_{chassis} - d_{COM_y1} * m_{bar1} - d_{COM_y2} * m_{bar2}}{\hat{m}_{chassis}} = 0.007716970m \tag{5.9}$$

$$d_{COM_z} = \frac{\hat{d}_{COM_z} * \hat{m}_{chassis} - d_{COM_z1} * m_{bar1} - d_{COM_z2} * m_{bar2}}{\hat{m}_{chassis}} = 0.081139300m \tag{5.10}$$

$$m_{chassis} = \hat{m}_{chassis} - (m_{bar1} + m_{bar2}) = 201.65Kg \tag{5.11}$$

where the subscripts 1,2 and indicate the center of mass data relative to bars 1 and 2.

In the end, the final results of the identification process are summarized in the subsequent table

d_{COM_x}	-0.0249272751 ± 1.7842% m
d_{COM_y}	0.007716970 ± 1.7842% m
d_{COM_z}	0.081139300 ± 7.1368% m
$m_{chassis}$	201.65 ± 0.9223% Kg

Table 5.2: Center of mass coordinates with respect to the reference frame defined in the geometrical center and total mass of the chassis.

5.2 Whole-body controller

Here are presented the results of the control law introduced in 3.3. Initially, these are shown isolating the control inputs to only one of the two systems at the time, then, the control is activated for both. Tests are conducted in Simulink and Gazebo and a comparison between the two is provided. In this section is included also a discussion about the models of the system and the consequences that they implicate on the closed-loop behaviour.

5.2.1 Separated systems: results in Simulink

To test the correct functioning of the control law the mobile manipulator has been divided into its two subsystems to reduce the overall complexity and have an easier debugging of the problems. This is made necessary by the high amount of degrees of freedom and the consequent high number of tunable parameters.

Mobile base impedance control The mobile base possesses only 3 DOF, therefore it constitutes an underactuated system for the 3-Dimensional space. Null space control is not possible for such a system, hence it has been eliminated, but everything else has been kept as described in 3.3.

Both in Simulink and Gazebo, the base has proven to be stable for a wide range of stiffness parameters. In the free motion case, the desired pose is asymptotically reached as expected.

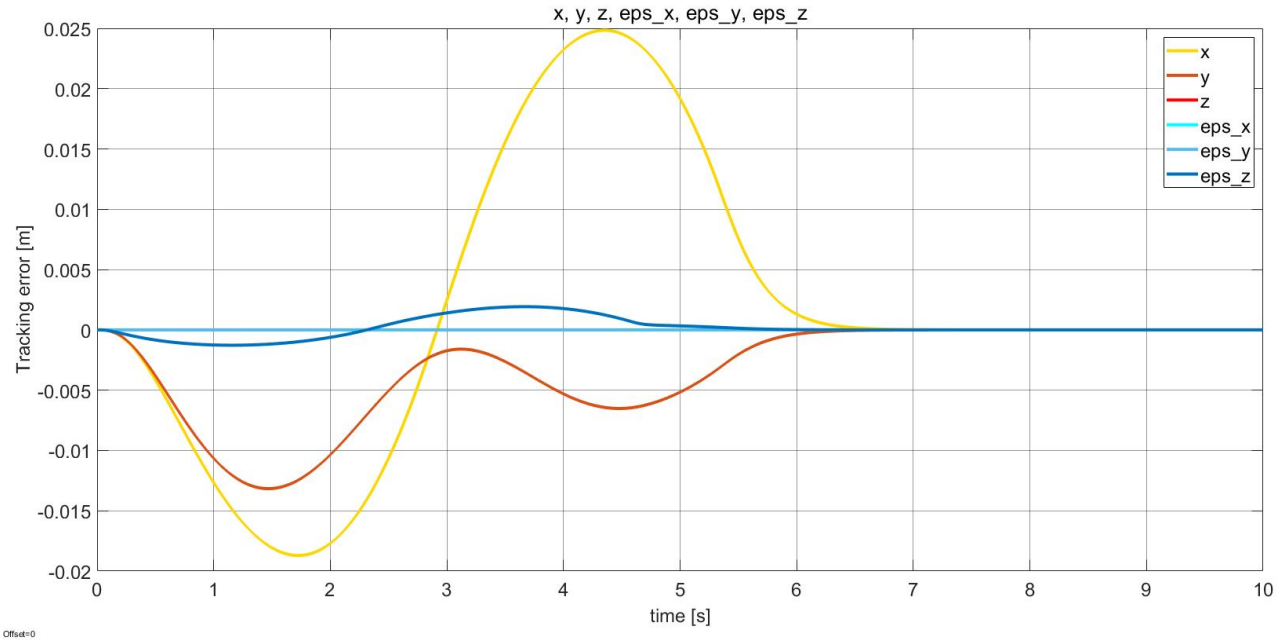
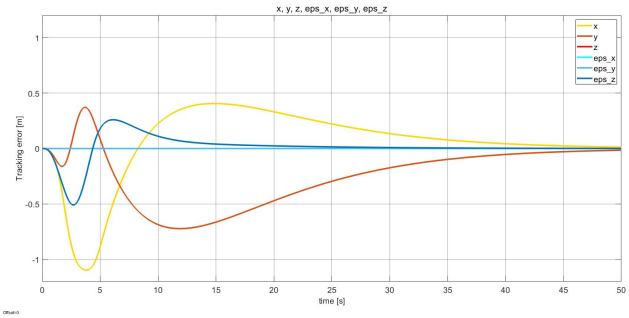
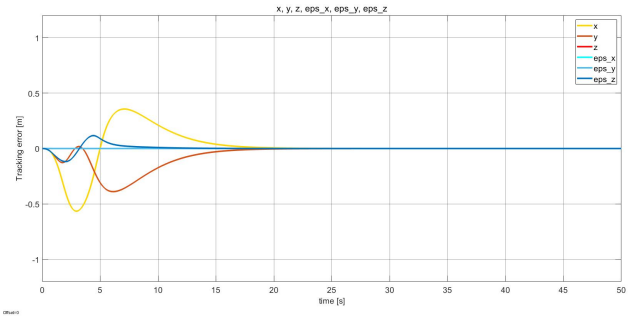


Figure 5.1: MATLAB: Mobile base only - Tracking error for $\boldsymbol{K}_d = 5000$, damping factor = 1. Requested task: motion along x and y directions of 1 meter, rotation around z of $\frac{\pi}{3}$

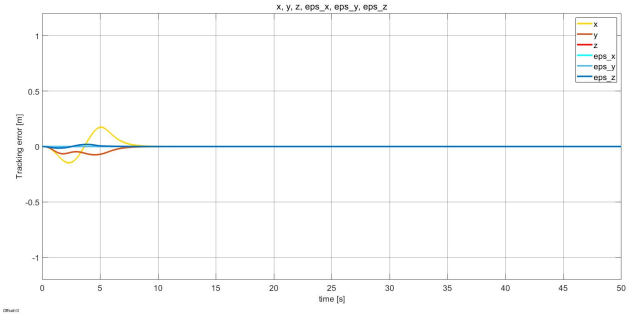
Increasing the stiffness it is possible to increase accuracy and reduce the time needed to reach the requested destination and a lower damping factor leads to oscillations confirming that the impedance control law has been implemented properly.



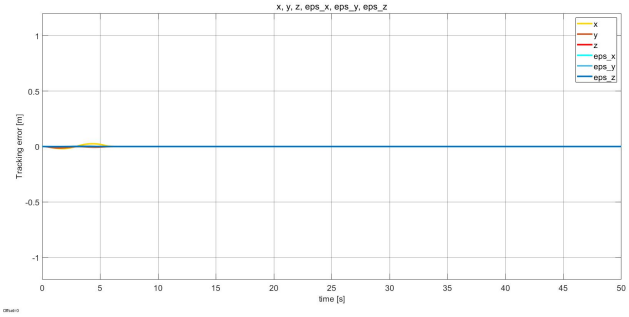
(a) $\boldsymbol{K}_d = diag(5)$



(b) $\boldsymbol{K}_d = diag(50)$

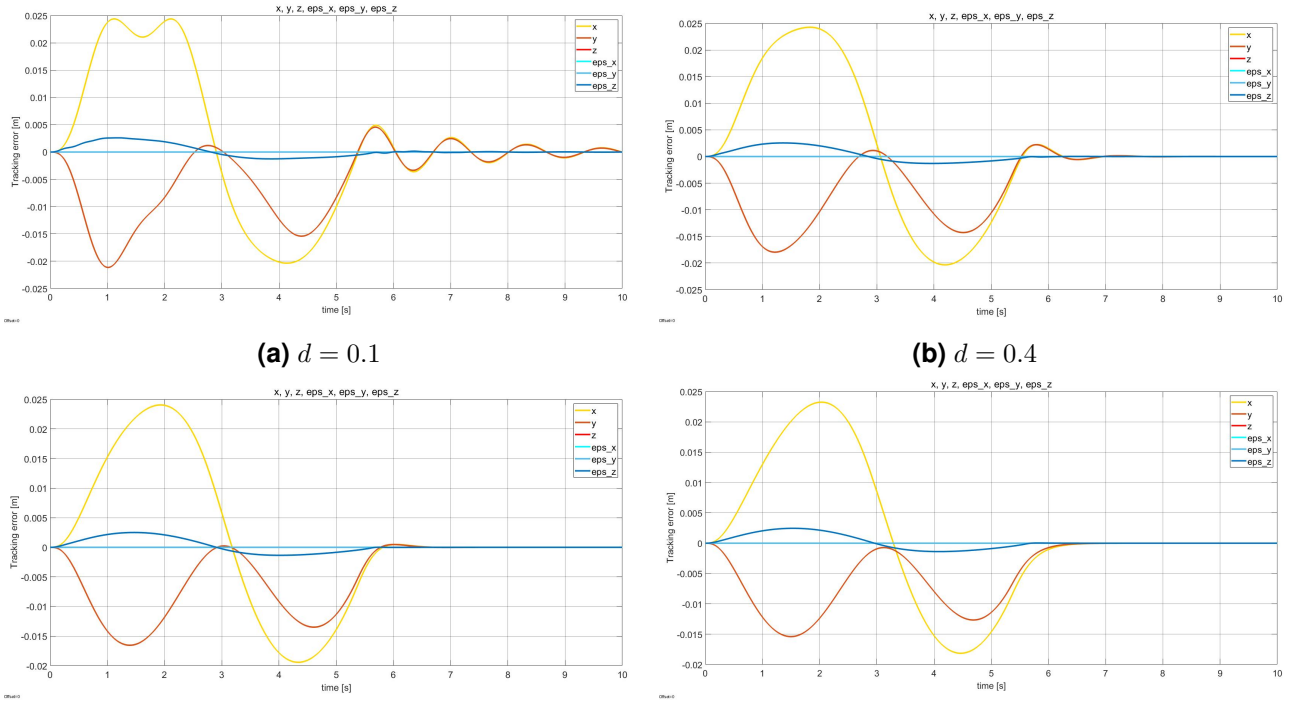


(c) $\boldsymbol{K}_d = diag(500)$



(d) $\boldsymbol{K}_d = diag(5000)$

Figure 5.2: MATLAB: Mobile base only - Tracking error comparison for different values of stiffness. The damping factor was kept constant at 1. Requested task: motion along x and y directions of 1 meter, rotation around z of $\frac{\pi}{3}$

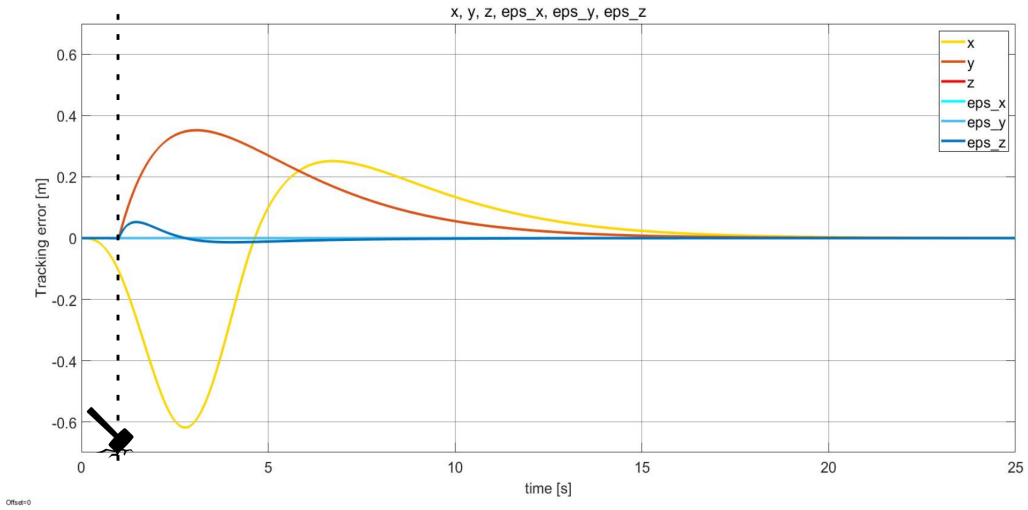


(a) $d = 0.1$ **(b)** $d = 0.4$

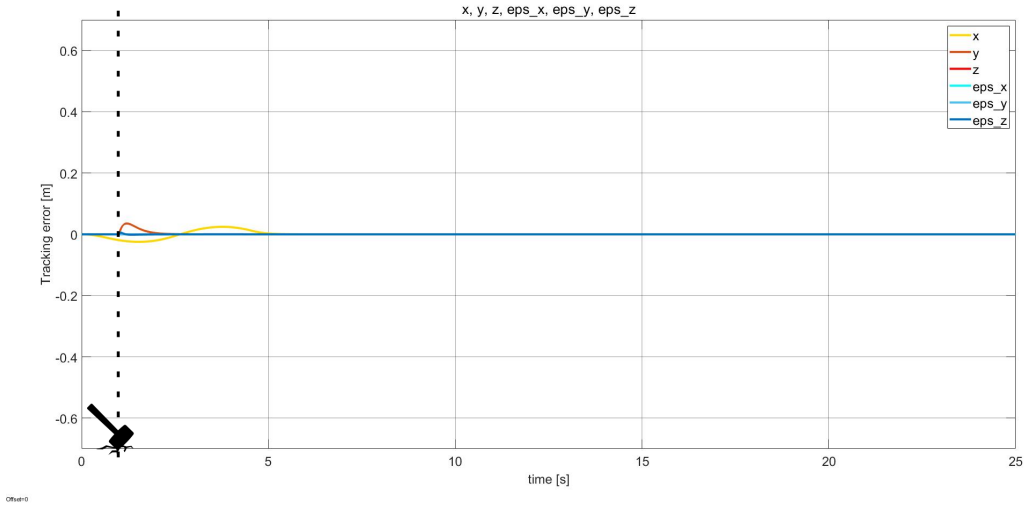
(c) $d = 0.7$ **(d)** $d = 1$

Figure 5.3: MATLAB: Mobile base only - Tracking error comparison for different values of damping factor. The stiffness was kept constant at $\boldsymbol{K}_d = diag(5000)$. Requested task: motion along x of -1 m, along y of 1 m, rotation around z of $-\frac{\pi}{2}$

Also during the interaction with the environment, the closed-loop system behaves as expected, showing a more or less compliant behaviour depending on the stiffness that has been imposed.



(a) Compliant behaviour for $\boldsymbol{K}_d = diag(50)$



(b) Compliant behaviour for $\boldsymbol{K}_d = diag(5000)$

Figure 5.4: MATLAB: Mobile base only - Tracking error for the requested task: motion along x. This figure compares the response to an external impulsive force applied along the y direction for different values of stiffness

Effect of uncompensated couplings on the tracking error Being a simpler system, the base is perfect to show some insights on the effects of non-linearities and on the consequence of designing the control law so that the stiffness principal axes are constant in the desired frame.

Consider, for example, a simple translation on the x-axis as the desired trajectory to be tracked, suppose that the system is linear [2] and that its starting orientation coincides with the one of the space frame $\{S\}$. The expected behaviour is an error along the x-axis that asymptotically goes to 0 while on the y-axis and on the orientation around the z-axis there is no error, a consequence of the fact that no motion has been requested.

And in fact, this is what can be observed:

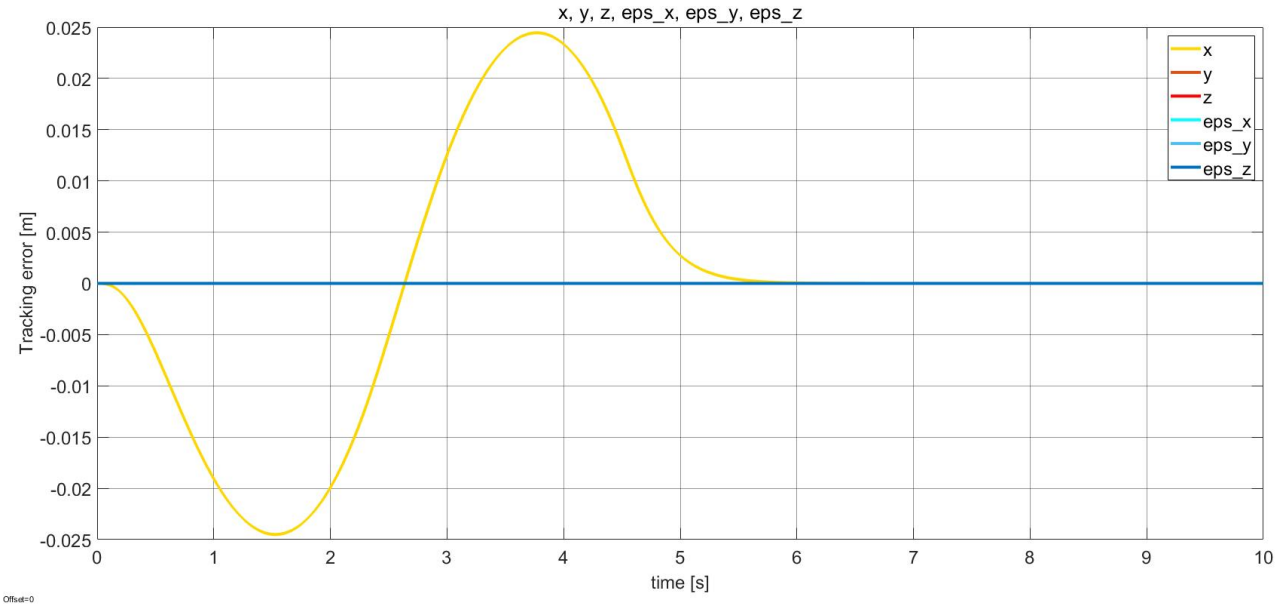


Figure 5.5: MATLAB: Mobile base only - Tracking error for $\boldsymbol{K}_d = diag(5000)$, damping factor = 1. Requested task: motion along x, starting orientation: ${}^{S}(0,0,0)$

Now, suppose that the robot is orientated with its x-axis rotated of 90° with respect to the x-axis of $\{S\}$. In this case it is important to remember that the error is represented in D frame whose orientation (since no rotation has been requested) coincides in every instant with the starting orientation of the mobile base. Therefore the error is appreciated on the y-axis rather than on the x-axis:

This is evident also looking at the Jacobian which assumes the shape:

$$\boldsymbol{J}_x = {}^{D}\boldsymbol{J}_v = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

[2] to simulate this kind of system, the non-diagonal terms have been eliminated from the simulation

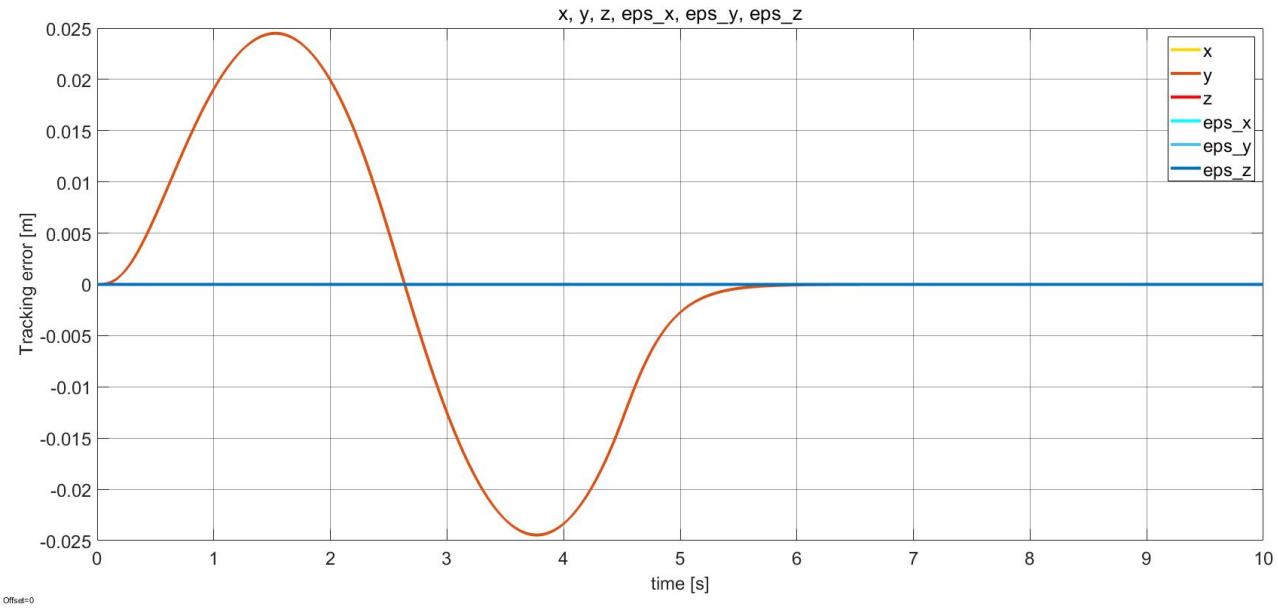


Figure 5.6: MATLAB: Mobile base only - Tracking error for $\boldsymbol{K}_d = diag(5000)$, damping factor = 1. Requested task: motion along x. Starting orientation: ${}^{S}(0, 0, \pi/2)$

Eventually, it is interesting to bring back the couplings and to analyse their contributions. The inertia matrix of the mobile base was found to be:

$$M_v = \begin{bmatrix} 201.65 & 0 & 4.38sin(\phi_z) - 1.35cos(\phi_z) \\ 0 & 201.65 & -4.38cos(\phi_z) - 1.35sin(\phi_z) \\ 4.38sin(\phi_z) - 1.35cos(\phi_z) & -4.38cos(\phi_z) - 1.35sin(\phi_z) & 201.65 \end{bmatrix}$$

Reflecting on $\boldsymbol{\tau} = \boldsymbol{M}_v \ddot{\boldsymbol{q}}_v + ...$ It is evident that an acceleration along x, which in a linear system only triggers a force in the x direction, in this case, also triggers a torque around the z-axis. Indeed this is what should happen employing 2.62 in which the non-linearities are left untouched, as a matter of fact the simulation confirms this theoretical analysis:

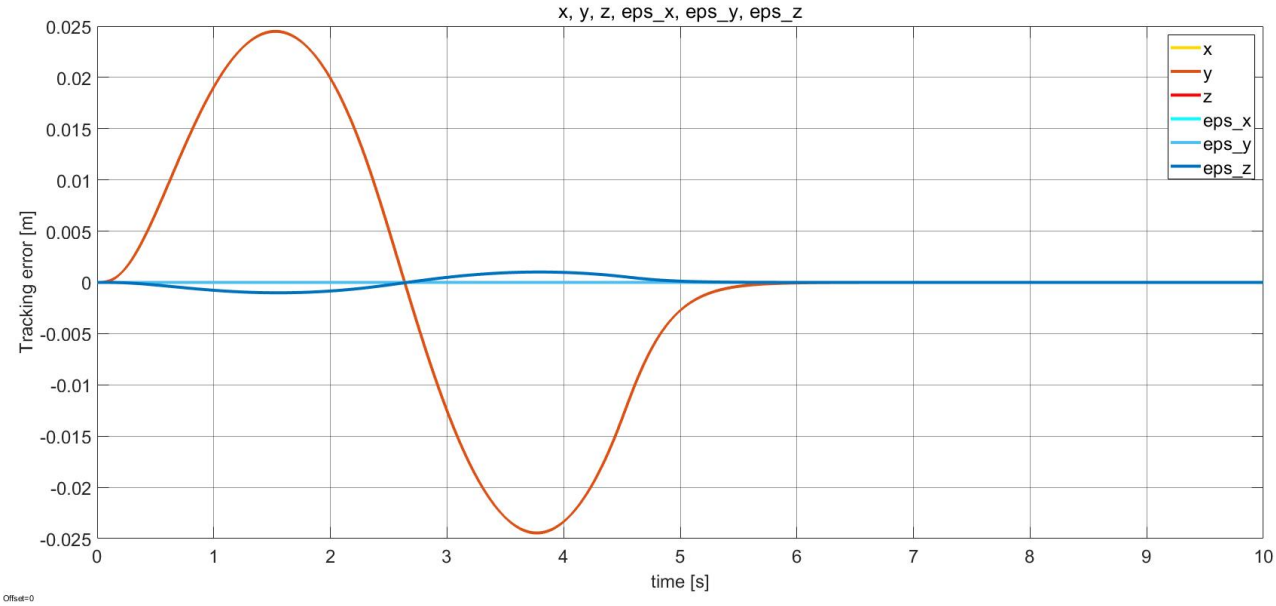


Figure 5.7: MATLAB: Mobile base only - Tracking error for $\boldsymbol{K}_d = diag(5000)$, damping factor = 1. Requested task: motion along x. Starting orientation: ${}^{S}(0, 0, \pi/2)$. Non-linearities considered

Arm impedance control and redundancy resolution The control of the arm has been tested in a very similar way to what has already been done for the base. Since it has 7 DOF, a simpler version of 3.12 has been employed, namely,

$$\nabla \boldsymbol{V} = \begin{bmatrix} k_{p,a1}(q_{a,1} - q_{a,1d}) + k_{v,a1}(\dot{q}_{a,1}) \\ k_{p,a2}(q_{a,2} - q_{a,2d}) + k_{v,a2}(\dot{q}_{a,2}) \\ k_{p,a3}(q_{a,3} - q_{a,3d}) + k_{v,a3}(\dot{q}_{a,3}) \\ k_{p,a4}(q_{a,4} - q_{a,4d}) + k_{v,a4}(\dot{q}_{a,4}) \\ k_{p,a5}(q_{a,5} - q_{a,5d}) + k_{v,a5}(\dot{q}_{a,5}) \\ k_{p,a6}(q_{a,6} - q_{a,6d}) + k_{v,a6}(\dot{q}_{a,6}) \\ k_{p,a7}(q_{a,7} - q_{a,7d}) + k_{v,a7}(\dot{q}_{a,7}) \end{bmatrix}$$

In this way, the internal motions are used to keep the arm the nearest possible to a certain configuration.

If the internal motions are not specifically controlled, even though the end-effector has reached the specified pose, the joints keep moving because there is no unique configuration for such a pose. Eventually, a kinematical singularity can be approached triggering an oscillating behaviour.

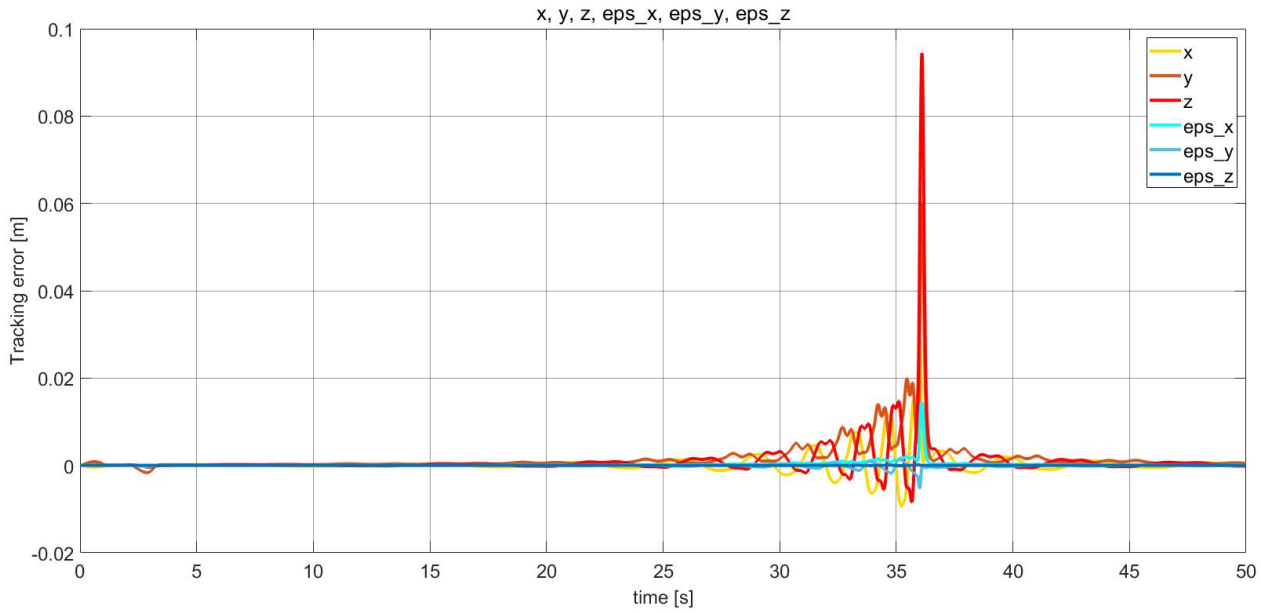


Figure 5.8: MATLAB: Arm only - Tracking error for $\boldsymbol{K}_d = diag(5000)$, damping factor = 1. Requested task: motion along x and y directions of 0.3 meters, rotation around z of $\frac{\pi}{6}$. Null-space control not implemented

By simply imposing $\boldsymbol{K}_{p,a} = diag([0, 0, 0, 40, 0, 0, 0])$ and $\boldsymbol{K}_{v,a} = diag([30, 30, 30, 30, 30, 30, 30])$, with $q_{a,4} = \frac{3\pi}{4}$ which corresponds to the request of trying to not extend the elbow, the system demonstrates to be asymptotically stable.

Non-linearities, tracking performance variations according to different values of stiffness and damping have been already addressed for the base and won't be repeated here but the same reasonings are valid for the arm case.

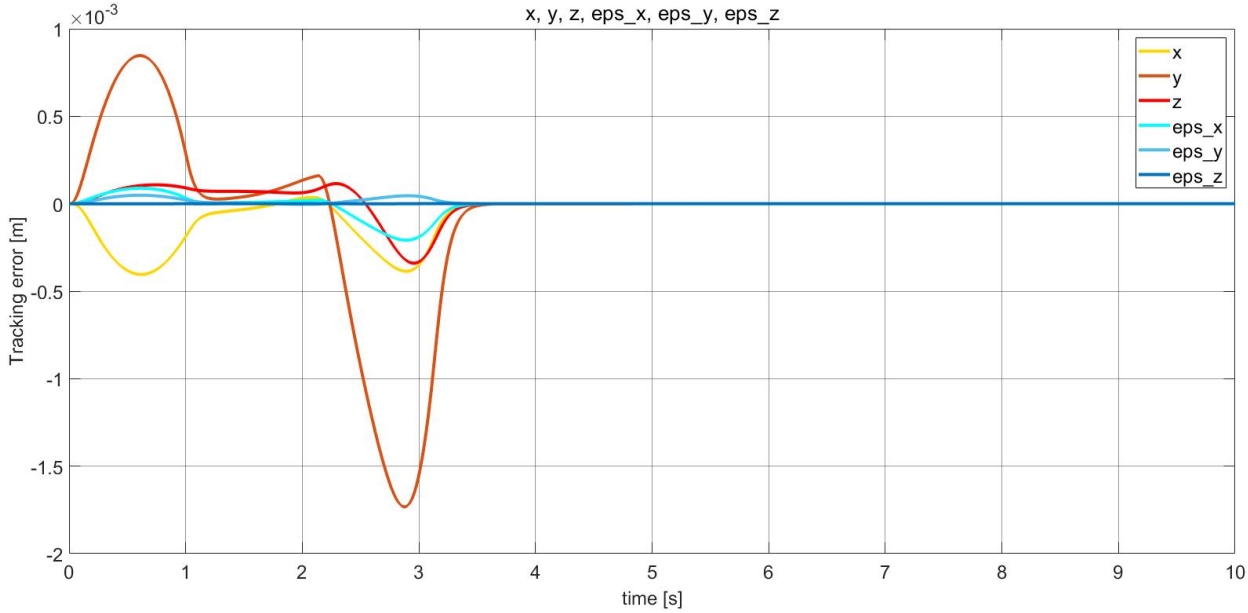


Figure 5.9: MATLAB: Arm only - Tracking error for $\boldsymbol{K}_d = diag(5000)$, damping factor = 1. Requested task: motion along x and y directions of 1 meter, rotation around z of $\frac{\pi}{6}$

5.2.2 Arm and base coordinated control: results in Simulink

When the arm and the base are subject to coordinated control, things get more complex. The difficulties seem to be related to the null-space control which, in this case, is bigger implying that more parameters have to be tuned. Their choice has proven to be critical for stability and tracking.

The error is kept low, especially for high values of stiffness in a very similar way to what has already been observed for the isolated cases. The subsequent graph summarizes the performances attained for

- $\boldsymbol{K}_{p,a} = diag([800, 800, 800, 800, 0, 0, 0])$
- $\boldsymbol{K}_v = diag([10, 10, 100, 300, 300, 300, 300, 300, 300, 300])$

which seem to be the best parameters for the secondary task.

It is interesting to analyse the effect of the first 2 elements of $\boldsymbol{K}_v$, in fact, it can be seen that higher values lead to smaller (or null) oscillations for the velocity $\dot{\boldsymbol{q}}_v$ but at the same time, they increase the base activation at the beginning of the motion, leading to bigger accelerations and worse tracking. However, the difference in tracking performances is of the order of 10^{-3}, therefore, choosing a higher $\boldsymbol{K}_v$ for the base might be the best if it is desired to reach a point with the base stopping as fast as possible.

Eventually, an external force has been applied to verify that the robot is capable of being compliant without incurring instabilities. Here is a demonstration of the reaction to an external force along y-direction, acting on the end-effector of magnitude 100N. As expected, high stiffness makes the robot react rigidly, low stiffness shows a compliant response.

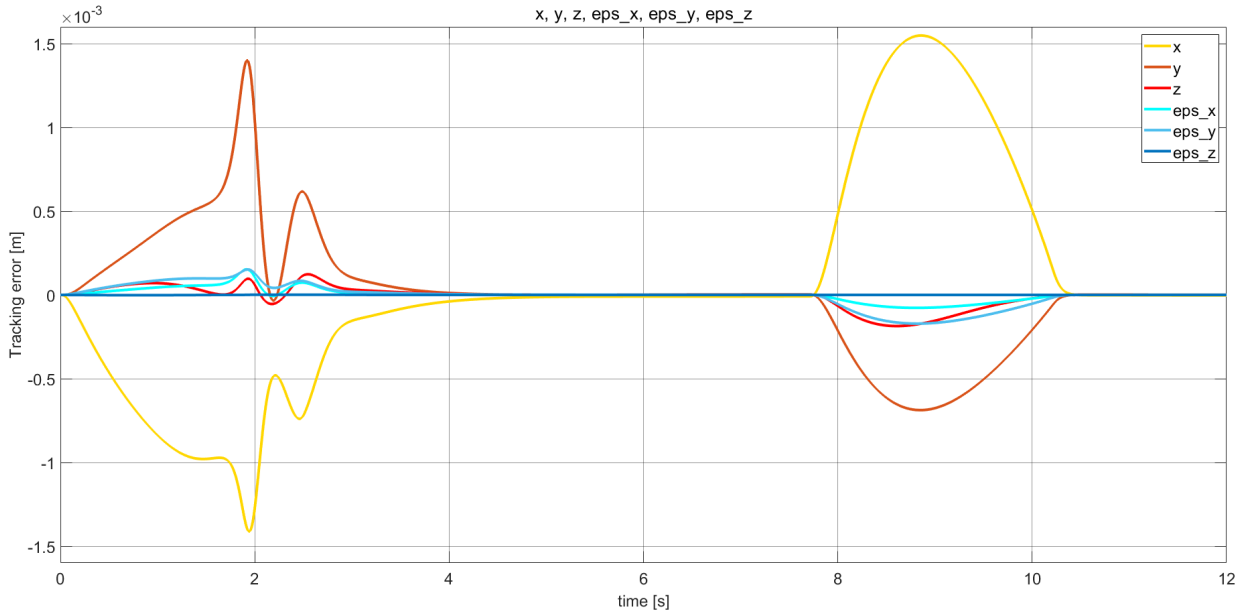
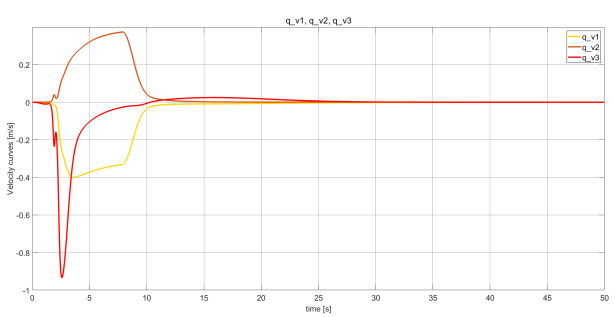
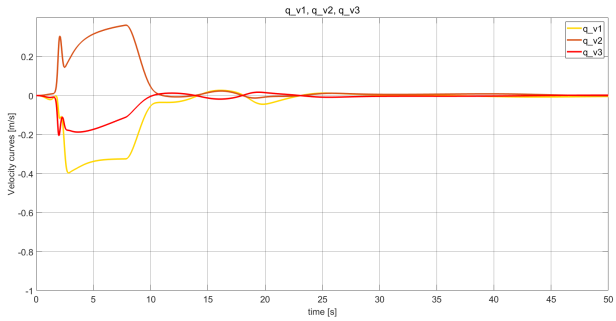


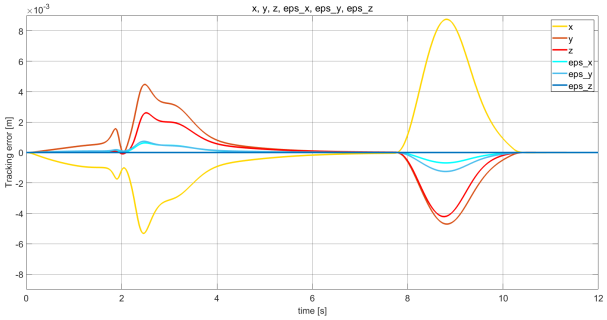
Figure 5.10: MATLAB: Whole-body control - Tracking error for $\boldsymbol{K}_d = diag(5000)$, damping factor = 1. Requested task: motion along x of 3 meters and along y direction of 1 meter, rotation around z of $\frac{\pi}{6}$.



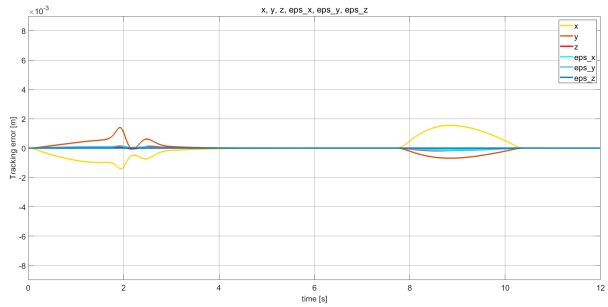
(a) $\dot{\boldsymbol{q}}_v$ curves for $\boldsymbol{K}_v = diag([\mathbf{100}, \mathbf{100}, 100, ...])$



(b) $\dot{\boldsymbol{q}}_v$ curves for $\boldsymbol{K}_v = diag([\mathbf{10}, \mathbf{10}, 100, ...])$

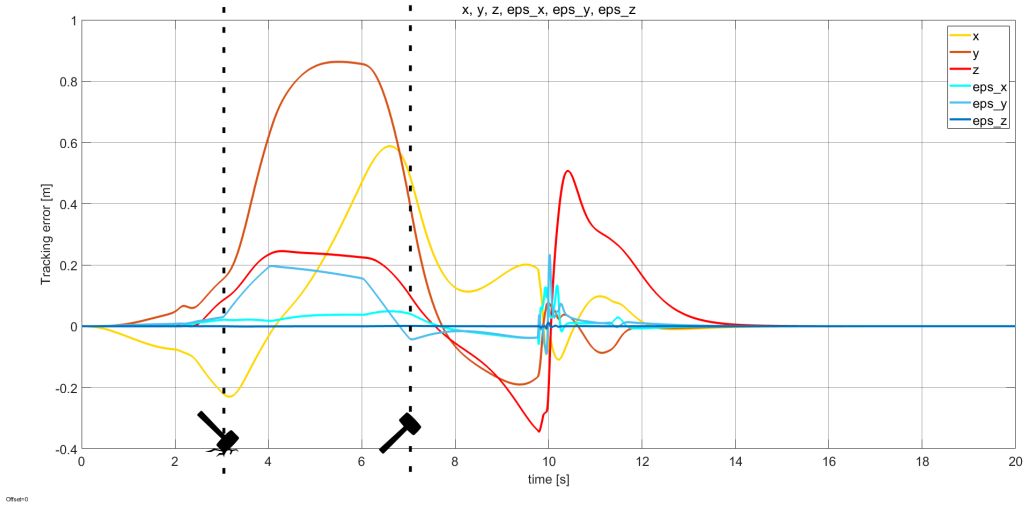


(c) Tracking error for $\boldsymbol{K}_v = diag([\mathbf{100}, \mathbf{100}, 100, ...])$

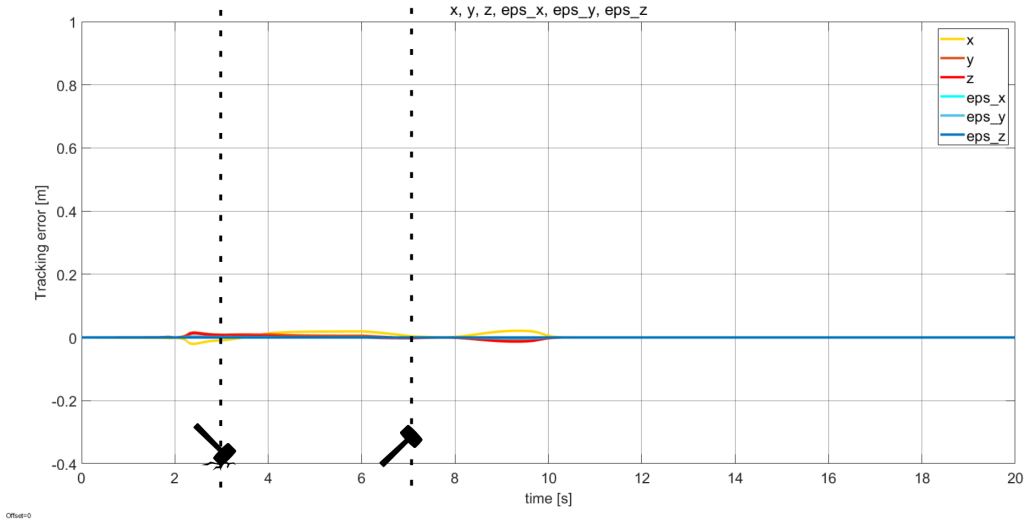


(d) Tracking error for $\boldsymbol{K}_v = diag([\mathbf{10}, \mathbf{10}, 100, ...])$

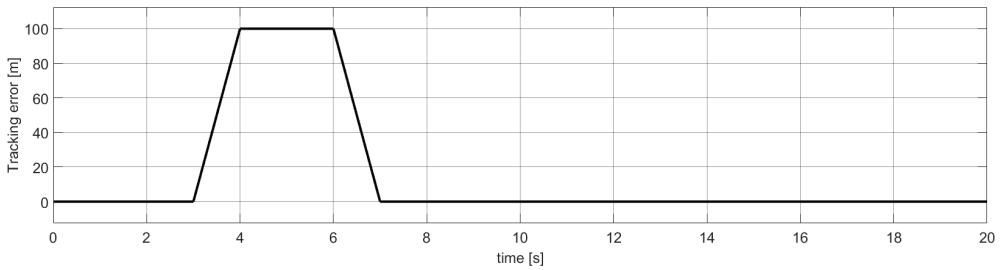
Figure 5.11: MATLAB: Whole-body control - Comparison between different values of $\boldsymbol{K}_v$. Impedance parameters: $\boldsymbol{K}_d = diag(5000)$, damping factor = 1.
Requested task: motion along x of 3 meters and along y direction of 1 meter, rotation around z of $\frac{\pi}{6}$.



(a) $\boldsymbol{K}_d = diag(50)$



(b) $\boldsymbol{K}_d = diag(5000)$



(c) Force profile applied on y-axis

Figure 5.12: MATLAB: Whole-body control - Tracking error for different values of stiffness. Requested task: motion along x of 3 meters and along y direction of 1 meter, rotation around z of $\frac{\pi}{6}$. An external force, of magnitude equal to 100N and shaped as a trapezoidal signal has been applied along the y-direction between seconds 3 and 7

The simulations show that this kind of approach to whole-body control is viable and leads to robust reliable results. However, this is just a first attempt and many enhancements can be applied on top of it as it will be discussed in 6.1. Especially important is the study of the internal motions that seem to play a big role in overall performance and stability.

5.2.3 Separated systems: results in Gazebo

Mobile base impedance control The mobile base has been tested with the same parameters adopted in Simulink and showed to be stable for almost all of them (too small values for stiffness and damping, e.g $\boldsymbol{K}_d = diag(10), d < 0.5$, lead to unstable oscillation. Nonetheless, tracking performances are not quite the same: the final destination is successfully reached with 0 error but during the motion, the tracking is not as good as in Matlab simulations.

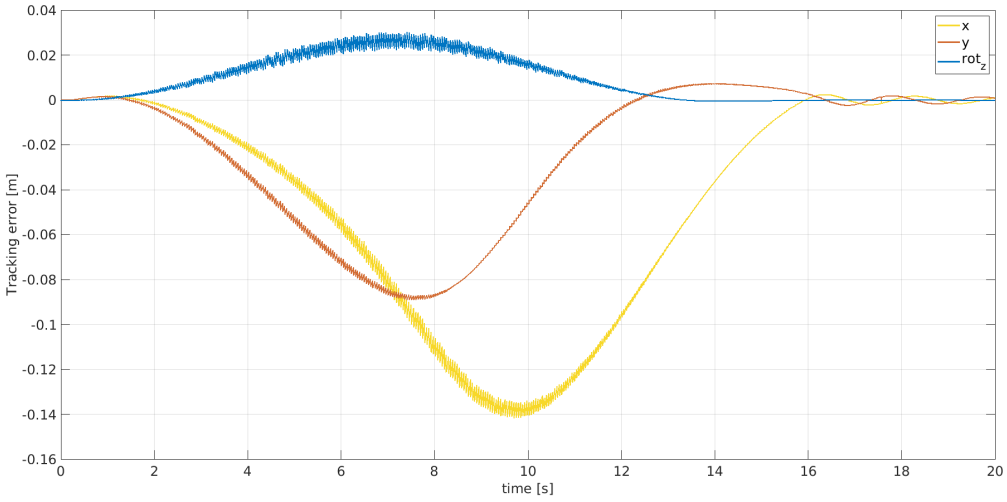
By looking at the velocity tracking it is possible to see that the commands sent from the admittance interface do not match the desired velocities computed by the trajectory generator. The cause, unfortunately, is yet to be discovered, it might be an error in the porting or a discrepancy with the modeled dynamics. Low-level controllers have been excluded from the causes since the commanded velocities are followed well by the robot, also the damping and the stiffness parameters seem to not play a big role in this regard. It has been tried to increase the stiffness or to reduce the damping but both lead to instabilities (the first one because it amplifies the noise, and the second one leads to unstable oscillations at the end of the motion, this is actually unexpected since the system should be theoretically stable for any damping choice).

Amplification of the noise is particularly relevant on rotations, this has already been made clear by fig. 5.15, but increasing $\boldsymbol{K}_d$ to $\boldsymbol{K}_d = diag(10000)$ it becomes even more evident [3]:

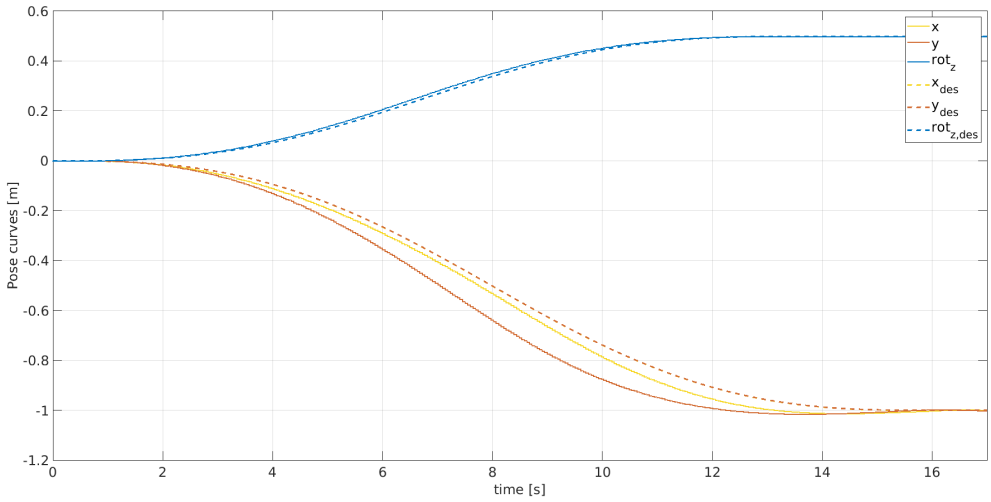
Arm impedance control and redundancy resolution Simulating the arm in Gazebo was more difficult. ROS provides several low-level controllers without thorough documentation. To speed up the developments it has been opted for a position interface, in this way, in fact, there is no need to suddenly activate gravity compensation at the start of the simulation, thus preventing the arm to fall down. Therefore, an admittance interface, similar to the one already adopted for the base, has been implemented inside the Impedance Control class. Other than the poor documentation, the fake sensors that measure the joint states from Gazebo are able to provide the measurements relative to only the type of interface in use (joint angles in this case), thus impeding the measurement of joint velocities. A tentative of differentiating the joint angles to obtain velocities was made but due to the noise in measurements and the lack of samples at certain time instants that led to big jumps to the 0 value, eventually, the command velocity derived from the admittance interface was used as a feedback, which, in general, is not a bad choice if there is good tracking.

Regarding the performances, resembling what happened for the base, the arm simulation in Gazebo is not able to match the ones attained in Simulink. Also in this case the velocities do not follow the desired ones, however, the tracking of the position is much better while the

[3] The small spikes in the linear part of the velocity are thought to be caused by the couplings with the rotation part



(a) Tracking error



(b) Desired trajectory vs measured trajectory

Figure 5.13: Gazebo: Mobile base only - $\boldsymbol{K}_d = 5000$, damping factor = 1.
Requested task: motion along -x and -y directions of 1 meter, rotation around z of $\frac{\pi}{3}$

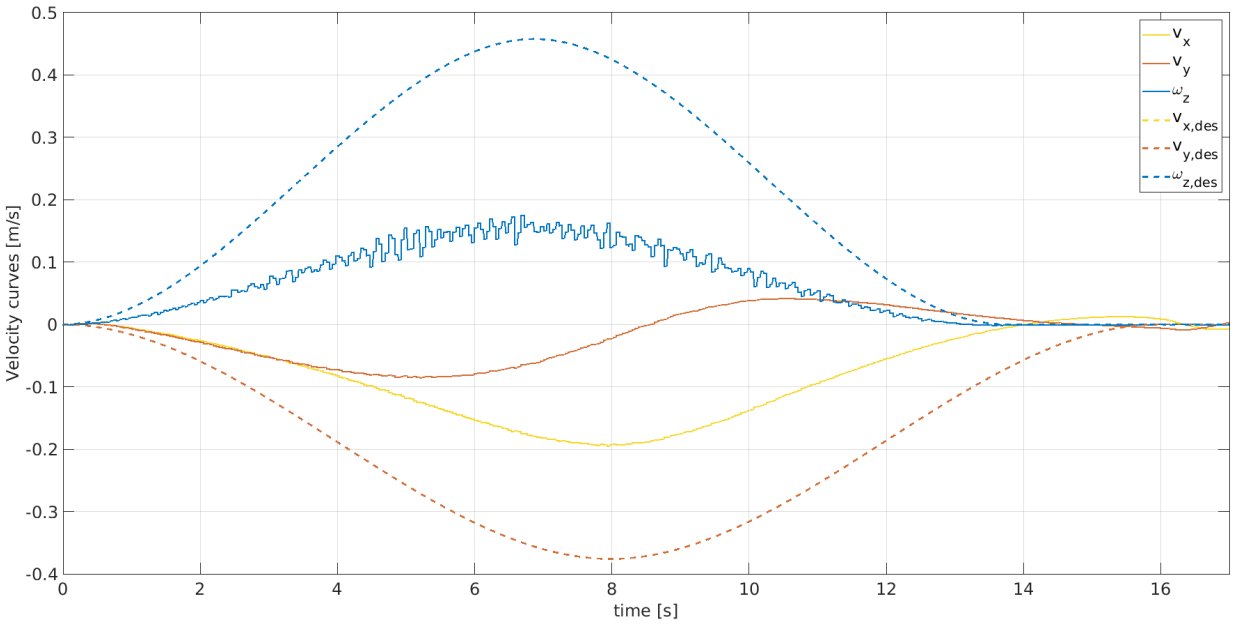


Figure 5.14: Gazebo: Mobile base only - Comparison between desired velocity and velocities computed by the admittance interface. $\boldsymbol{K}_d = 5000$, damping factor = 1.
Requested task: motion along -x and -y directions of 1 meter, rotation around z of $\frac{\pi}{3}$

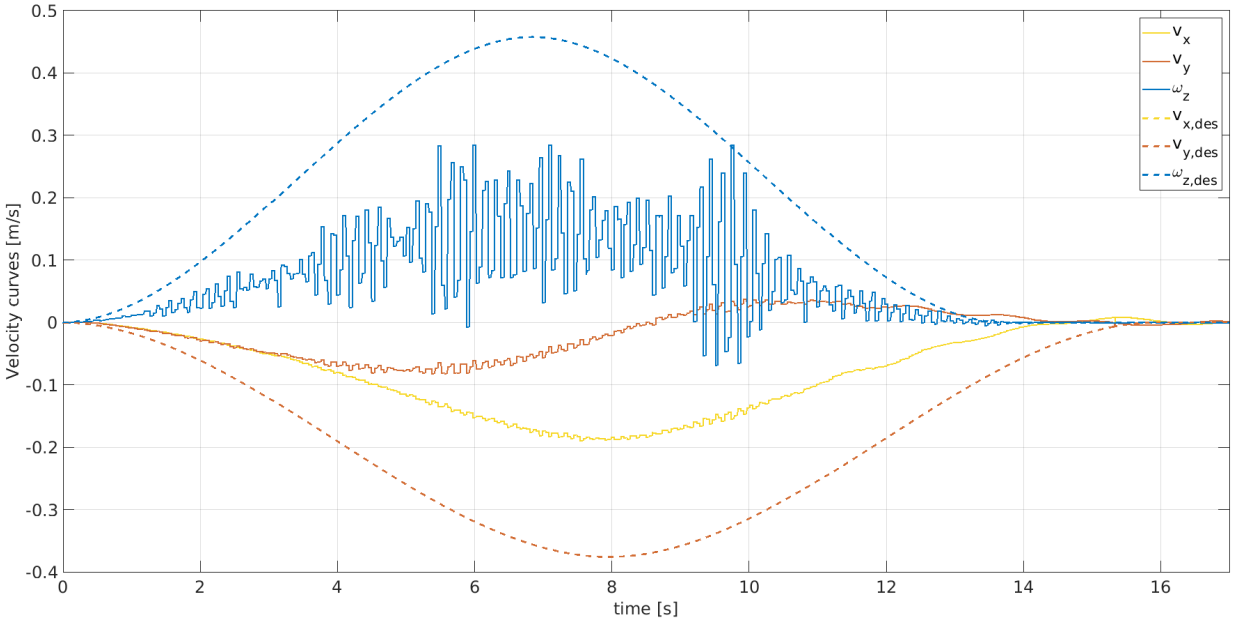


Figure 5.15: Gazebo: Mobile base only - Noise amplification due to high stiffness values.
$\boldsymbol{K}_d = 10000$, damping factor = 1.
Requested task: motion along -x and -y directions of 1 meter, rotation around z of $\frac{\pi}{3}$

orientation has some problems. As a matter of fact, while the position is almost perfectly overlapping the reference, the orientation is subject to oscillations. This is a consequence of a low rotational stiffness, unfortunately, increasing its values leads to instability. The reasons behind this occurrence are not clear, noise shouldn't interfere since the velocity is directly retrieved from the admittance interface.

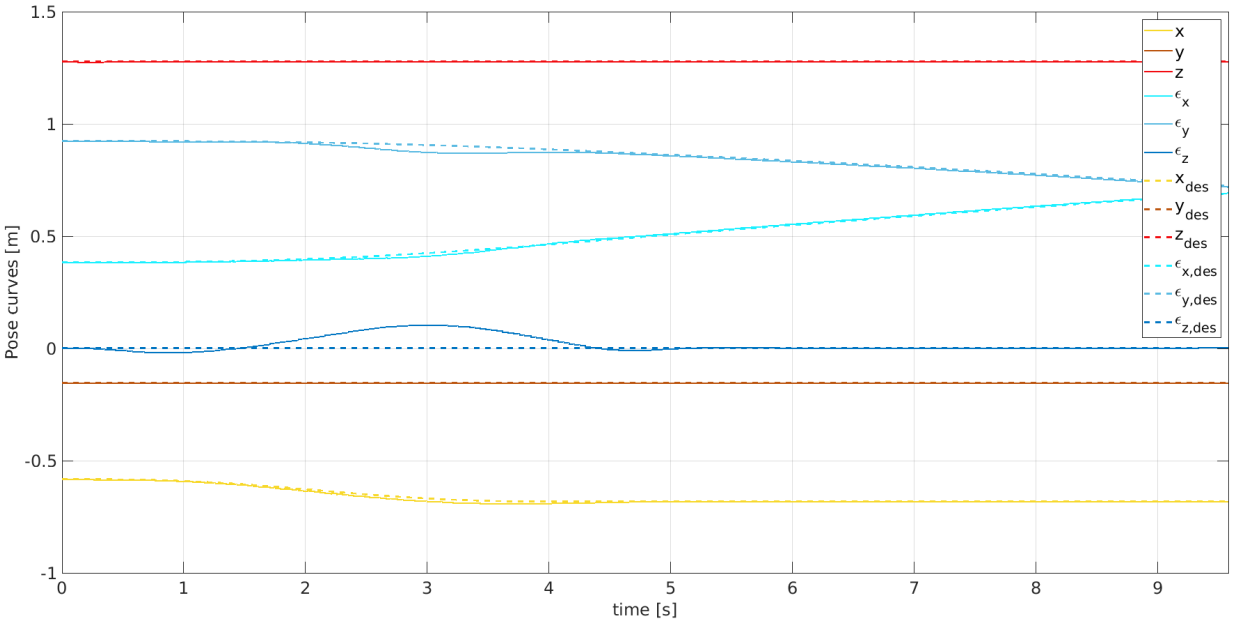


Figure 5.16: Gazebo: Arm only - Measured pose curves vs desired pose curves.
$\boldsymbol{K}_d = (1000, 1000, 1000, 2, 2, 2)$, damping factor = 1.
Requested task: motion along x of 0.1 meter, rotation around z of $\frac{\pi}{3}$

To conclude, since the translational stiffness can be chosen in a very wide range [0,1000], it is probable that there is a non-identified bug in the code/simulation or some non-modelled dynamics that shrinks the range of the rotational stiffnesses to such a low level.

5.2.4 Arm and base coordinated control: results in Gazebo

It is obvious that the whole-body control will suffer from the same problems that its subsystems introduce. Also in this case the rotational elements of the stiffness have to be chosen low. The low accuracy and the resulting oscillations caused by the low stiffnesses make the arm have big and slow oscillations that are made worse by the base movements. It was not possible to find parameters that make the simulation stable. More work has to be done to discover what makes the difference between Simulink and Gazebo.

5.3 Safety implementation

Safety was successfully implemented both in simulation and on the real system (it is currently in use in the DARKO project). Here the results obtained for a POI placed on the arm while the base is in motion are shown.

The gSMU client sends a desired velocity to be attained by the base (in this case an oscillatory motion along x), and the gSMU server receives the request and calculates a safe velocity in the case in which the robot is performing an unsafe motion in the vicinity of a human. The human is considered in proximity when its distance from the robot is below 1 meter. A simple kinematic control was adopted to show the proper functioning of the algorithm.

As it can be seen the base velocity (blue) suddenly drops to a safe one when the human approaches. Certainly, this is not what is directly experienced in a real system because such an abrupt change is not viable, however, there are time relaxation techniques that can hinder a very similar behaviour with a smoother transient.

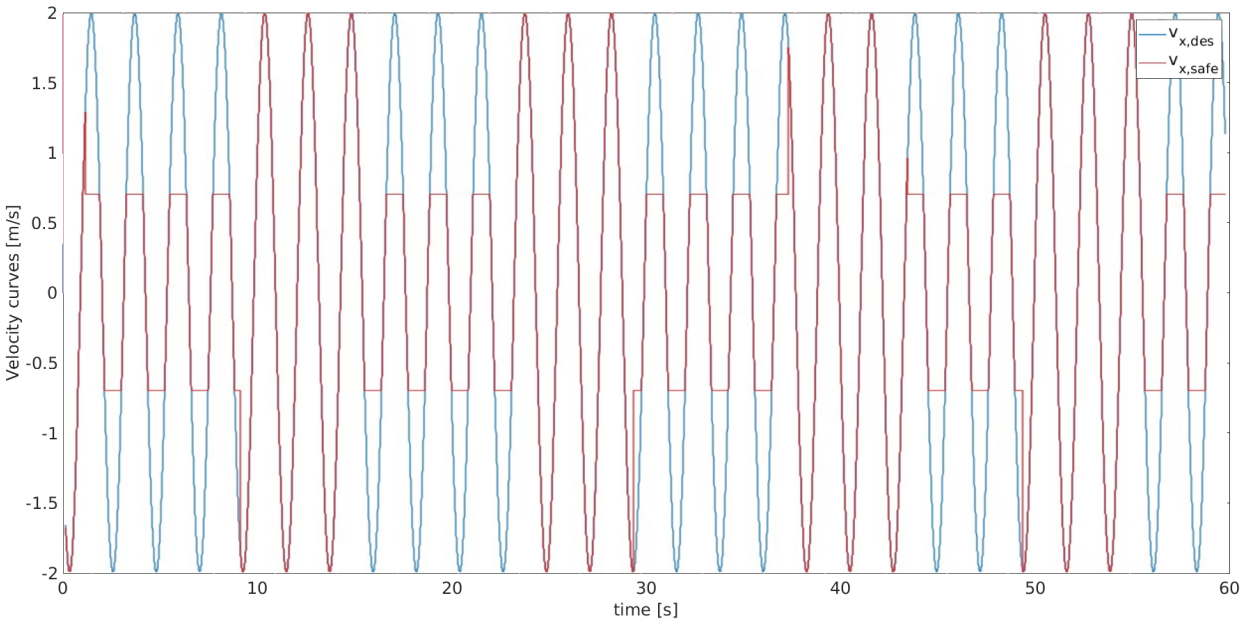


Figure 5.17: gSMU simulation in Gazebo: safe velocity versus desired velocity

6 Conclusions

In this work, a model of the dynamics and kinematics of an omnidirectional-wheeled mobile manipulator was introduced. Starting from this, a model-based control framework was developed with the intention of attaining a safe and compliant behaviour. Safety is addressed through an application of the generalized Safe Motion Unit [28] while compliance is accomplished by means of impedance control which is applied to both the mobile base and the robotic arm. In addition to this, a way of tackling base-arm coordinated motions in a non-programmatic way was presented exploiting the theory of artificial potential fields.

The theoretical formulations have then been tested in Simulink, showing promising results and confirming that the chosen path is viable. To approach the implementation of the framework on a real system, the platform *RB-KAIROS+*, currently in use in the DARKO project, was taken as an example. An identification of its parameters was performed and a ROS package was created to match the software architecture available. The package has been tested on a ROS/Gazebo simulation that was adapted from the one provided by the DARKO project. Unfortunately, the complexity of the simulation led to many difficulties that couldn't be all addressed in a so short amount of time. Currently, there are some problems that didn't allow to prove the framework on the real system but that seem to be mostly related to undiscovered bugs in the software or in the simulation environment. Nonetheless, the ROS/Gazebo simulation was enough to prove the functioning of the implementation of the gSMU and the developed package was even deployed on the real system by the colleagues of the DARKO project.

Surely, this framework is far from being a thorough one and doesn't solve many of the problems that mobile manipulation brings in. Therefore, this work has to be seen as a starting point that with some refining and the addition of new features could really bring lead to interesting applications.

6.1 Future improvements

The final stages of this thesis have opened some questions without an answer. Though really promising, the control framework should be tested on a real system to get the final confirmation that the theory matches the practice. Similarly, the identification procedure adopted for the center of mass has to undergo a validation experiment and has to be followed by an inertia identification to get an exhaustive model of the system. Regarding, safety, instead, the gSMU has to be assessed for a list of simultaneous POIs.

Subsequently, it is possible to extend the framework by adding new features. Luckily its inherent structure and the high number of DOF facilitate this process. Hereby some ideas

are disclosed:

- **Simultaneous separated tasks through hierarchical control:** The artificial potential field adopted allows to use the DOF of the base to automatically extend the workspace of the manipulator. Though this is certainly useful, it might be interesting to perform disjoint motions, like reaching a certain destination while grasping an object or performing manipulation. A direct implementation of this could be done through hierarchical control. Essentially, this is equivalent to defining multiple tasks, one task can be performed using a certain amount of DOF, therefore the others can be exploited for tackling some other action. In order to distinguish between tasks the concept of *prioritized Jacobians* is introduced [31], each of these Jacobians will lead to a torque that can be added up to execute all the tasks. A similar approach has been followed also in [32]
- **Artificial potential field enhancements:** The artificial potential field, at the moment, is provided with a fixed arm configuration to be attained. However, this configuration should be planned accordingly to the task to be accomplished. A possible criterion for his choice could be the maximization of manipulability or to avoid joint limits.
- **Choosing impedance parameters on the line:** The impedance controller introduced in this thesis implements a fixed stiffness. This is certainly undesirable. A possible solution would be to experimentally determine primitives for associating a correct stiffness to each application, otherwise, an optimization approach could be used (e.g. [20])[1]
- **Active vibration suppression:** Mecanum wheels are well known for being a cause of big vibrations. This side effect surely impacts on the accuracy of the manipulation, this is why probably a vibration control should be implemented at the manipulator level or through an active suspension placed between the arm and the mobile base.
- **Solution to the subsystems' different dynamics:** Wheels' actuators are usually provided with low-level controllers that run at a very lower frequency with respect to the actuators available in cobots. As a consequence, this discrepancy should be taken into account in order to avoid undesired closed-loop behaviours.

[1] Notice that in this case optimization doesn't threaten safety since the control low is stable for any positive definite stiffness matrix

Contents

Bibliography

[1] Bruno Siciliano et al. *Robotics: Modelling, Planning and Control*. 1st. Springer Publishing Company, Incorporated, 2008. ISBN: 1846286417.

[2] Oussama Khatib. *Advanced Robotic Manipulation*. Stanford University, 2005.

[3] Ciro Natale. *Interaction Control of Robot Manipulators*. Springer Tracts in Advanced Robotics. Springer Berlin, Heidelberg, 2003. ISBN: 978-3-540-00159-1. DOI: https://doi.org/10.1007/3-540-36155-3.

[4] Christian Ott. *Cartesian Impedance Control of Redundant and Flexible-Joint Robots*. Springer Berlin, Heidelberg, 2008. DOI: https://doi.org/10.1007/978-3-540-69255-3. URL: https://doi.org/10.1007/978-3-540-69255-3.

[5] J. Luh, M. Walker, and R. Paul. "Resolved-acceleration control of mechanical manipulators". In: *IEEE Transactions on Automatic Control* 25.3 (1980), pp. 468–474. DOI: 10.1109/TAC.1980.1102367.

[6] John J. Craig. *Introduction to Robotics : Mechanics and Control. 3rd ed.* Upper Saddle River: Pearson Prentice Hall, 2005.

[7] Kevin M Lynch and Frank C. Park. *Modern Robotics: Mechanics, Planning, and Control*. English (US). Cambridge Univeristy Press, 2017. ISBN: 978-1107156302.

[8] Wansoo Kim et al. "MOCA-MAN: A MObile and reconfigurable Collaborative Robot Assistant for conjoined huMAN-robot actions". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)* (2020), pp. 10191–10197.

[9] Alexander Dietrich et al. "Whole-body impedance control of wheeled mobile manipulators". In: *Autonomous Robots* 40.3 (2016), pp. 505–517. ISSN: 1573-7527. DOI: 10.1007/s10514-015-9438-z. URL: https://doi.org/10.1007/s10514-015-9438-z.

[10] Yuqiang Wu et al. "A Teleoperation Interface for Loco-Manipulation Control of Mobile Collaborative Robotic Assistant". In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 3593–3600. DOI: 10.1109/LRA.2019.2928757.

[11] X. Yun and Y. Yamamoto. "Internal dynamics of a wheeled mobile robot". In: 2 (1993), 1288–1294 vol.2. DOI: 10.1109/IROS.1993.583753.

[12] V. Padois, J.-Y. Fourquet, and P. Chiron. "Kinematic and dynamic model-based control of wheeled mobile manipulators: a unified framework for reactive approaches". In: *Robotica* 25.2 (2007), 157–173. DOI: 10.1017/S0263574707003360.

[13] O. Khatib. "A unified approach for motion and force control of robot manipulators: The operational space formulation". In: *IEEE Journal on Robotics and Automation* 3.1 (1987), pp. 43–53. DOI: 10.1109/JRA.1987.1087068.

[14] Thomas N. E. Greville Adi Ben-Israel. *Generalized Inverses*. Springer-Verlag, 2003. DOI: 10.1007/b97366.

[15] Jonghoon Park et al. "Multiple tasks kinematics using weighted pseudo-inverse for kinematically redundant manipulators". In: *Proceedings - IEEE International Conference on Robotics and Automation* 4 (Feb. 2001), 4041 –4047 vol.4. DOI: 10.1109/ROBOT.2001.933249.

[16] CARSTEN SCHEDLINSKI and MICHAEL LINK. "A SURVEY OF CURRENT INERTIA PARAMETER IDENTIFICATION METHODS". In: *Mechanical Systems and Signal Processing* 15.1

(2001), pp. 189–211. ISSN: 0888-3270. DOI: https://doi.org/10.1006/mssp.2000.1345. URL: https://www.sciencedirect.com/science/article/pii/S0888327000913451.

[17] Homayoun Seraji. “A Unified Approach to Motion Control of Mobile Manipulators”. In: *The International Journal of Robotics Research* 17.2 (1998), pp. 107–118. DOI: 10.1177/027836499801700201. eprint: https://doi.org/10.1177/027836499801700201. URL: https://doi.org/10.1177/027836499801700201.

[18] Jesse Haviland, Niko Sünderhauf, and Peter Corke. “A Holistic Approach to Reactive Mobile Manipulation”. In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 3122–3129. DOI: 10.1109/LRA.2022.3146554.

[19] Neville Hogan. “Impedance Control: An Approach to Manipulation: Part I II and III”. In: *Journal of Dynamic Systems, Measurement, and Control* 107.1 (1985), pp. 1–24.

[20] Mathew Pollayil et al. “Choosing Stiffness and Damping for Optimal Impedance Planning”. In: *IEEE Transactions on Robotics* PP (Jan. 2022), pp. 1–20. DOI: 10.1109/TRO.2022.3216078.

[21] A. Albu-Schaffer et al. “Cartesian impedance control of redundant robots: recent results with the DLR-light-weight-arms”. In: 3 (2003), 3704–3709 vol.3. DOI: 10.1109/ROBOT.2003.1242165.

[22] Christian Ott, Ranjan Mukherjee, and Yoshihiko Nakamura. “A Hybrid System Framework for Unified Impedance and Admittance Control”. In: *Journal of Intelligent & Robotic Systems* 78.3-4 (2014), pp. 359–375. DOI: 10.1007/s10846-014-0082-1.

[23] F. Caccavale et al. “Six-DOF impedance control based on angle/axis representations”. In: *IEEE Transactions on Robotics and Automation* 15.2 (1999), pp. 289–300. DOI: 10.1109/70.760350.

[24] Oussama Khatib. “The Potential Field Approach And Operational Space Formulation In Robot Control”. In: *Adaptive and Learning Systems: Theory and Applications*. Ed. by Kumpati S. Narendra. Boston, MA: Springer US, 1986, pp. 367–377. ISBN: 978-1-4757-1895-9. DOI: 10.1007/978-1-4757-1895-9_26. URL: https://doi.org/10.1007/978-1-4757-1895-9_26.

[25] Sheldon Jay Axler. *Linear algebra done right*. Springer, 1997, p. 251. ISBN: 0387982590.

[26] Sami Haddadin et al. “On making robots understand safety: Embedding injury knowledge into control”. In: *The International Journal of Robotics Research* 31.13 (2012), pp. 1578–1602. DOI: 10.1177/0278364912462256. eprint: https://doi.org/10.1177/0278364912462256. URL: https://doi.org/10.1177/0278364912462256.

[27] Buckley RE Rüedi TP and Morgan CG. “AO Principles ofFracture Management. vol 1, 2nd edition.” In: *Stuttgart: ThiemeVerlag* (2007).

[28] Mazin Hamad et al. “Modularize-and-Conquer: A Generalized Impact Dynamics and Safe Pre-collision Control Framework for Floating-Base Tree-Like Robots”. In: *IEEE Transactions on Robotics* (2023), pp. 1–22. DOI: 10.1109/TRO.2023.3257515.

[29] J. Russakow and O. Khatib. “A New Control Structure for Free-Flying Space Robots”. In: *i-SAIRIS: International Symposium on Artificial Intelligence, Robotics, and Automation in Space*. Vol. 2. Toulouse, France, 1992.

[30] J.R. Taylor. *An Introduction to Error Analysis: The Study of Uncertainties in Physical Measurements*. ASMSU/Spartans.4.Spartans Textbook. University Science Books, 1997. ISBN: 9780935702422. URL: https://books.google.de/books?id=ypNnQgAACAAJ.

[31] Federico Moro and Luis Sentis. “Whole-Body Control of Humanoid Robots”. In: Jan. 2019, pp. 1161–1183. ISBN: 978-94-007-6045-5. DOI: 10.1007/978-94-007-6046-2_51.

[32] Alexander Dietrich, Christian Ott, and Jaeheung Park. “The Hierarchical Operational Space Formulation: Stability Analysis for the Regulation Case”. In: *IEEE Robotics and Automation Letters* 3.2 (2018), pp. 1120–1127. DOI: 10.1109/LRA.2018.2792154.