Robotics Force/Torque Control for Manufacturing Operations

Noelie Ramuzat

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :
l'Institut National des Sciences Appliquées de Toulouse (INSA de Toulouse)

Présentée et soutenue le *18/02/2021* **par :**

Noëlie RAMUZAT

Contrôle en force/couple pour des opérations industrielles
Robotics Force/Torque Control for Manufacturing Operations

JURY

Vincent PADOIS	Directeur de Recherche	Président du Jury
Philippe FRAISSE	Professeur des Universités	Rapporteur
Andrea DEL PRETE	Assistant Professeur	Examinateur
Christian OTT	Directeur de Département	Examinateur
Serena IVALDI	Chargée de Recherche	Examinatrice
Olivier STASSE	Directeur de Recherche	Directeur de Thèse
Sébastien BORIA	Ingénieur R&D	Encadrant de Thèse

École doctorale et spécialité :
EDSYS : Robotique 4200046
Unité de Recherche :
LAAS - Laboratoire d'Analyse et d'Architecture des Systèmes (UPR 8001)
Directeur(s) de Thèse :
Olivier STASSE
Rapporteurs :
Philippe FRAISSE et *Vincent PADOIS*

There are no safe paths in this part of the world.
Remember you are over the Edge of the Wild now,
and in for all sorts of fun wherever you go.

J. R. R. Tolkien
(The Hobbit, or There and Back Again)

Remerciements

Je voudrais remercier en premier lieu mon directeur de thèse Olivier Stasse qui m'a guidée et accompagnée durant ma thèse. Je le remercie pour sa disponibilité et son aide qui m'ont permis de réaliser mes travaux, même dans des domaines nouveaux pour l'équipe. Merci de m'avoir soutenue constamment pendant ces trois années assez particulières.
Merci également à Sébastien Boria, mon tuteur entreprise, et Damien Van Damme pour leur soutien côté entreprise. Je vous remercie pour vos échanges d'idées et votre aide afin que ma thèse se passe sans encombre au sein d'Airbus.

Je remercie Philippe Fraisse et Vincent Padois d'avoir accepté d'être mes rapporteurs et de m'avoir apportée beaucoup d'éléments de réflexion et de correction sur mon manuscrit. Merci pour votre lecture approfondie et de votre bienveillance pendant ma soutenance. Merci aussi à Christian Ott, Serena Ivaldi et Andrea Del Prete pour avoir fait partie de mon jury de thèse et pour vos retours constructifs lors de ma soutenance. Merci encore à Andrea pour son soutien sur TSID et merci à Vincent Bonnet pour son aide pour mon premier papier.

Je souhaite remercier tout spécialement l'équipe Gepetto, qui a été un environnement de travail et de vie incroyable. Je vous remercie pour toute la joie, la bonne ambiance et l'entraide dont l'équipe déborde ; pour les temps de pause autour d'un gâteau, d'un café ou d'un jeu de carte. Un grand merci à tout mon bureau et à ses invités, j'ai eu la chance d'être dans le meilleur bureau du monde.
Merci enfin à ma famille et mes ami.e.s, mes soeurs et ma mère en particulier, et à mon partenaire Axel pour m'avoir soutenue et supportée jusqu'à la fin.

Table of Contents

List of Figures

List of Tables

List of Theorems

List of Algorithms

Chapter 1

Context of the thesis

Contents

In this chapter, some of the figures presented were realized for the Work-Package (WP) reports in the scope of the joint-laboratory Robots For the Future of Aircraft Manufacturing (ROB4FAM). They are indicated to be part of these reports by noting WP in their captions. Moreover, Louise Scherrer has kindly authorized us to use some of her figures from her Master report Scherrer [1] to illustrate some principles

1.1 Introduction

Robotics consists in the study of machines able to perform tasks that were previously executed manually. In an industrial context, it is often repetitive and draining tasks, which may lead to safety and health issues (such as musculoskeletal disorders). Thus, machines, called robots, are studied in terms of design and control in order to execute these types of industrial applications. These operations belong to semi-structured environments, often known a-priori, and their realizations by a robot is called automation. Generally, industrial robots can be separated in two groups: one where the robot is used for mass production and needs only to have satisfying execution repeatability to efficiently execute tasks. And another one where the industrial process favors robots with high position accuracy over repeatability capability; and even versatility to be able to realize different tasks. In aircraft industry, a robot which can react and adapt to unknown environments and unplanned events is desirable. Indeed, some parts of the aircraft production process are closed to unstructured environments and adaptation of the solution would allow to maintain the production process even when failure occurs, which is often a point of concern.

However, adaptability is rarely completely achieved in industry, as the robot needs to process the information given by its sensors to react accordingly to a planned or unplanned situation. Moreover, it is not always the most important or prioritized part of the process. For instance, when the industrial process consists in manufacturing a huge amount of identical products, the predictable quality of the repeatability process is more relevant than the ability to adapt. In general, a robot is programmed to execute a task, and various programs can be implemented to allow the robot to switch between them and fit with the manufacturing process. For instance, a robot can first drill a hole, then change its end-effector (versatility of the solution) and switch automatically to another task program such as deburring the hole. However, this type of programming cannot be called "adaptable" as it does not use actively the robot sensors to react to unplanned event. This scenario is currently the most common in industry Siciliano et al. [2]. Online path and solution modifications for the robot when unplanned events occur often require high computation resources and more complex algorithms.

In the industrial context, the robots are commonly rigid manipulators with six Degrees of Freedom (DoFs) but also parallel robots (Stewart Gough platform) or hybrid robots (for instance, a robot with 3 parallel DoFs and 2 serial DoFs). In particular, these robots are fixed to the ground or to a fixed structure integrating auxiliary axis (to add redundancy to the system). However, they can also be assembled on a mobile base to augment their reachable workspace. This design choice comes from the structured working environment where they perform tasks. Considering that the environment is well identified, the level of autonomy needed by the robot is less important compared to the expected quality results (such as the precision or repeatability). However, when the adaptability criterion becomes dominant, advanced robotics solutions are needed. For instance, if a human operator is not available or is not safe in the working environment, how can a robot still perform the task ? Some disaster or emergency-response scenarios have been addressed in the DARPA (Defense Advanced Research Projects Agency) Robotics Challenge (DRC) [3] (June 2015), which presents complex tasks in dangerous, degraded, human-engineered environments. These scenarios included for instance walking on debris or remove them

from a blocked entryway, climbing industrial ladders and walking across industrial walkways or using a tool to break through a concrete panel. Even for non disaster scenarios, adaptability is a desirable ability to safely work in such environments and alongside humans.

In the scientific domain, researches on advanced robotics solutions lead to more diverse type of robots: from rigid to compliant, from manipulators to drones, from quadruped and biped to humanoids. In particular, the literature presents multiple solutions on humanoid robots as they are complex systems designed to operate in the same environment than humans and perform the same tasks. Humanoid robots were the most represented type of robots in the DRC, indeed, one of their interesting purposes is to achieve tasks that can be dangerous for humans. The humanoid robots offer two main challenges to the literature, represented by their two main capabilities: the locomotion and the manipulation. By addressing the potential issues of the robotic domain on these complex systems, the literature proposes exhaustive solutions that cover most of the simpler systems' needs. In particular, a solution developed for a humanoid robot can be transposed to an industrial 6 DoFs robot by simplifying the formulation of the problem. This is why, in this thesis, we focus on designing solutions for a humanoid robot to then transfer them onto an industrial manipulator robot arm.

1.2 ROB4FAM

On the 21st May 2019, a joint lab between Airbus Operations SAS and the Laboratory for Analysis and Architecture of Systems (LAAS) is inaugurated. It is named ROB4FAM and creates a collaboration between the Airbus Commercial Aircraft Mechatronics and Robotics team and the LAAS-CNRS Gepetto team. Its goal is to develop innovative robotics solutions to achieve reactive manufacturing operations in a manufacturing plant environment, while evolving closed to human operators. The idea is to deploy the scientific toolset developed by the Gepetto team on industrial robots, so that these robots can take their environment into account to detect anomalies (uncertainties or environment changes) and adapt to them, continuously (in real time). Moreover, the proposed solutions must ensure that the robots' behaviors are stable, safe and robust in order to interact with the surrounding environment and in particular humans.

The robotics platforms used by the Gepetto team during my thesis are two robots made by the PAL-Robotics company: the humanoid robot TALOS and the wheeled manipulator robot TIAGo (see Fig.1.4). One advantage of these two robots is that they are built aiming modularity and cross platform compatibility. The hardware and software of the robots are close enough such that the developed solutions on TALOS can be easily transferred to TIAGo (see Section 1.4). After the validation on these research platforms, the implemented solutions must be adapted and deployed on their industrial counterpart, such as the Middle Size Drilling Robot (MSDR), see Chapter 6.

My thesis is part of the ROB4FAM joint lab, focusing on the implementation of a torque whole-body control on the humanoid robot TALOS for manufacturing operations. The results of this joint-lab are open-source, under the license BSD-v2.

1.3 Objectives and Contributions of the Thesis

This thesis is a CIFRE, realized within the Airbus Operations SAS company in partnership with the LAAS. It is one of the two first thesis launched in the scope of the ROB4FAM joint-lab. The thesis tries to answer the issue of robotics whole-body control for manufacturing operations. One major concern is to ensure the safety of the operation for the robot and for its environment, for instance humans. Indeed, the current robotic systems implemented in factories are separated from the workers by safety fences. It requires a huge space compared to what a human needs and when a failure occurs it is necessary to stop the production chain to check the problem in the cell. A smaller and safer solution is thus targeted, which must be adaptable to the robotic platform and able to react to unplanned events. Thus, the thesis focuses on the development of innovative solutions on the more adaptable robots available in the laboratory, in particular the humanoid robot TALOS (see Section 1.4). This robot has high capabilities, can be controlled in torque and is commercially available; a part of the realized works contribute toward the robot evaluation on locomotion and multi-contact scenarios.

The aim is for the robot to succeed manufacturing operations such as drilling, tightening or deburring. Each of these operations requires high precision and a force application using a specific tool. Therefore, during this thesis the question of which type of control to use has been addressed. Applying a specific amount of force can be achieved by different methods, the most common being position control using force sensors and direct torque control. The chosen solution has moreover to respect the safety concern. To resume, this thesis address the following problems:

- ▷ How to efficiently control a humanoid robot during manufacturing operations: using torque or position control;
- ▷ How to ensure the safety of the solution, by studying the stability and convergence of the solution toward an equilibrium.

To answer these problems and fulfill the objectives this thesis proposes the following contributions: firstly the design of a controller protecting the robotic system using a Model Predictive Control (MPC), secondly the benchmarking and implementation of three different humanoid robots control architectures, thirdly the application of one of this controller in a context of human-robot collaboration. And finally, the design of a novel passive whole-body controller, validated through a force application simulation, with the proof of its stability to ensure the safety and robustness of the solution.

1.4 The robotics platforms: TALOS, TIAGo and MSDR

1.4.1 Humanoid robot TALOS

The humanoid robot TALOS has been designed and built by the Spanish company PAL-Robotics following the call to tender of the LAAS. Thus, the requirements and specifications of the robot have been made jointly with the laboratory Stasse et al.

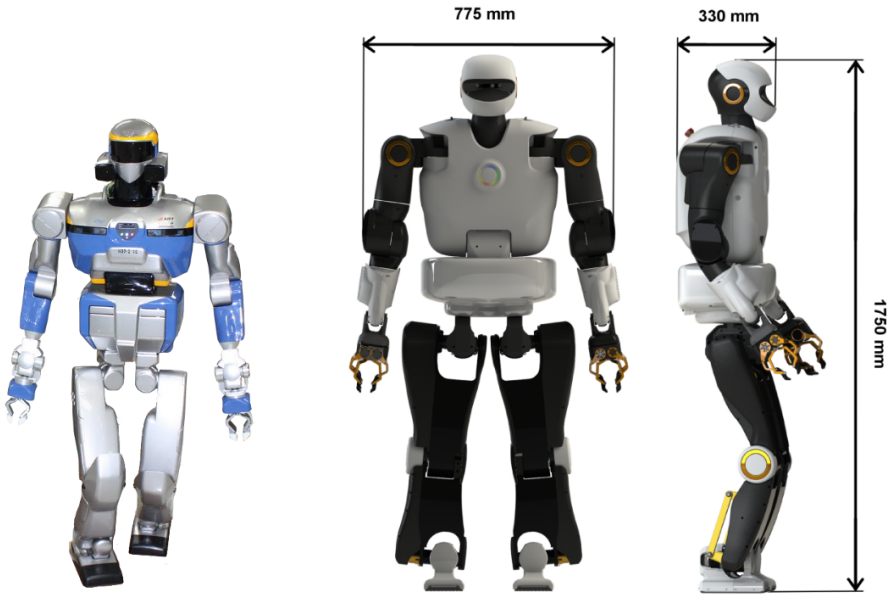

Figure 1.1: From HRP2 to TALOS.

[4], aiming to create a successor to the robot HRP-2 Kaneko et al. [5] developed by the National Institute of Advanced Industrial Science and Technology (AIST). The goal was to create a robot capable of dexterous bi-handed manipulation which can involve high payload, with better locomotion performances and with size and mass better suited to work in human environment. This is why TALOS is taller and heavier than HRP-2 (see Fig.1.1) and has torque and temperature sensors.

In details, the robot is 1.75m tall, weights around 100kg and has electric actuators. It has 32 Degrees of Freedom (DoFs) with the following distributions of its joints: two legs of 6 DoFs, a waist of 2 DoFs, two arms of 7 DoFs with a 1 DoF gripper each and a head of 2 DoFs. The rigid chain actuators of TALOS are composed of a brushless motor, that can be controlled in current by the software, connected to a harmonic drive and a torque sensor attached to its corresponding link (except for the head and the grippers). The motor and joint positions are measured by two high-precision encoders (19 bits or $524,288$ counts per revolution), one encoder just after the motor and one encoder on the link after the harmonic drive. In addition, there are temperature sensors for each joint of the robot. The kinematic structure of TALOS is depicted in Fig.1.2. Each depicted frame is located at the origin of a joint. All joints are revolute. The base link of the robot is located close to its center of mass, in the waist. As every humanoids robots, TALOS is not ground-fixed to the environment, it is an under-actuated system. This under-actuated part, called floating base, represents the position of the robot in the world and relies on external contact forces to control its motion. It is a significant difference compared to fixed industrial manipulator robots (without auxiliary axis, such as the scenario considered for the MSDR).

TALOS has an Inertial Measurement Unit (IMU) in its torso, providing filtered measures of the orientation, the angular speeds and the linear accelerations at 1kHz. Force/Torque sensors are integrated in the ankles and wrists of the robot. TALOS possesses two Ubuntu operating system computers (18.04), one for the control

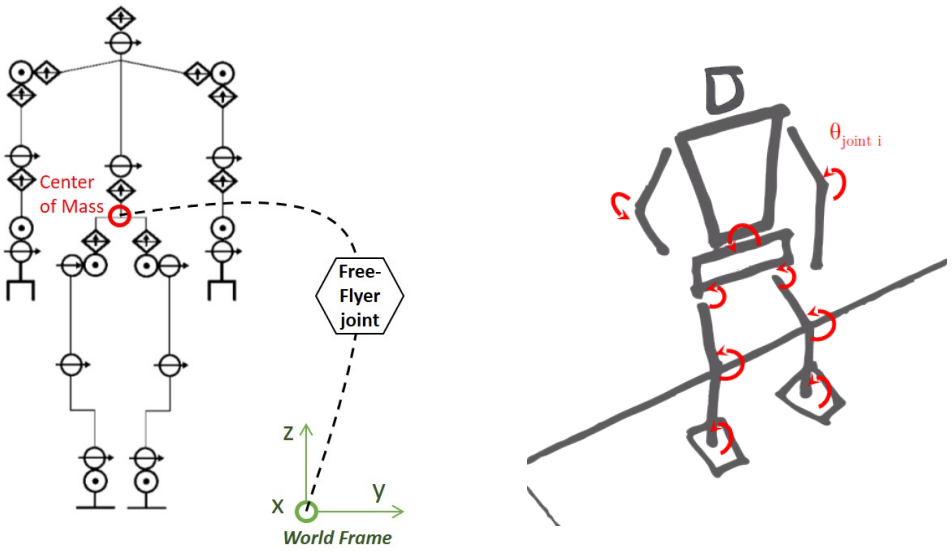

Figure 1.2: Kinematic structure of TALOS Scherrer [1]

(patched with the Linux Real-Time Preempt) and one for the vision processes. The control computer is connected to all the actuators and sensors of the robot by an EtherCAT bus [6]. It uses the middle-ware Robot Operating System (ROS) Stanford Artificial Intelligence Laboratory et al. [7] and reaches a control-loop frequency of 2kHz. This high-frequency allows to obtain a quick reaction time necessary for real time complex behaviors. Because of the ROS architecture it is possible to use MoveIt! Coleman et al. [8], Chitta et al. [9] to plan the robot motions, and the Gazebo simulator Koenig and Howard [10] to simulate dynamically the robot TALOS. PAL-Robotics has additionally implemented a specific simulator for their robot which simulates its actuators dynamics.

The robot TALOS of the Gepetto team is named Pyrène and is the first of the TALOS production series of the PAL Robotics company. Therefore the subsequent iterations have been improved, in particular the torque sensors and the feet have been updated. Pyrène has been eventually revised following these updates but still has some differences with the newest version. Indeed, some changes have been done during the production process of the robot (like the torque sensors hosting and the cabling) and thus cannot be replicated on Pyrène. Moreover, the robot manufacturer parameters calibration was not satisfying, and the actuators parameters are not disclosed (see Chapter 3). Pyrène head has been modified following the request from the Gepetto team to include a LIDAR and a second camera in addition to the color and depth Orbbec Astra camera primarily mounted (see Fig.1.3).

1.4.1.a The flexibility on the hip of TALOS

Humanoid robots often have flexible or compliant components. For instance, the actuators stiffness of the robot WALKMAN Negrello et al. [11] can be directly tuned, creating an intended flexibility. Another example of humanoid robot with compliant material is HRP-2 [5]. It includes a bush rubber in the ankle in order to smooth impacts. In the robot TALOS, a non-intended flexibility on the hip link has been

Figure 1.3: Pyrène head with the LIDAR and the two cameras.

observed and impacts meaningfully the control of its legs and, therefore, its balance and locomotion. Indeed, this flexibility (not modeled in the simulator) leads to errors in the landing positions of the feet on the real robot. However, the deflection is not directly measurable by the encoders and cannot be directly modified. This subject is more detailed in the Appendix 4.

1.4.2 Wheeled Manipulator robot TIAGo

The standard robot TIAGo used by ROB4FAM is constituted of four main components: the head, the torso, the arm and the mobile base (see Fig.1.4). TIAGo's head has 2 DoFs and is equipped with stereo microphones, a speaker and an RGB-D camera. The torso of the robot supports the arm and the head, and is equipped with an internal lifter mechanism to change the height of the robot achieving a minimum and a maximum height of 110cm and 140cm respectively. The arm of TIAGo is composed of 7 DoFs with 3 DoFs for the wrist and its end-effector is a 5 finger under-actuated hand. A Force/Torque sensor is integrated on the end-point of the wrist. Finally, TIAGo mobile base has a differential drive mechanism and contains an on board computer, batteries, power connector, laser-range finder, three rear sonars, a user panel, a service panel and two WiFi networks to ensure wireless connectivity. The IMU is mounted at the center of the mobile base and can be used to monitor inertial forces and attitude.

Because TIAGo has a mobile base, the balancing issues that the humanoid robot encounters are not as critical. Indeed, the wheels allow a stable way of locomotion in opposition to legs. However, they will not allow the robot to reach some areas,

Figure 1.4: Left: Humanoid robot TALOS. Right: Manipulator robot TIAGo

such as the ones requiring stairs climbing, non flat terrain or with an access slope of more than 5 degrees.

The developed solution on the humanoid robot TALOS can be transferred on TIAGo thanks to the similarity of the software and hardware of the arms. Indeed, in both cases it is possible to control the actuators directly using current commands or positions commands. Moreover, using the software stacks of the Gepetto team, the solutions are modular because they take into account the Unified Robot Description Format (URDF) of the robot, thus the kinematics and dynamics of the robot are computed based on it. The relevant part of the solution for the TIAGo is the control of the arm, but using the complete solution validated on TALOS allows to have a safe, stable and robust result (see Chapter 5).

1.4.3 Middle Size Drilling Robot

Airbus manufacturing engineering teams are working on light automation concepts. For this, they are looking for lighter robots than the ones currently used in the factories. Indeed, some of the classical industrial robotics cells design can reach 20 tons in total including the high payload robot, the heavy end-effector and the infrastructure requested to integrate them (see Fig.1.4.3). This type of solution is disproportionate considering that the final goal is to drill holes about $\varnothing 4,8$mm. Moreover, this huge industrial setup is hard to move, thus has a limited travel workspace. It is a fairly high cost investment which is technically complex, not flexible and with a huge blocking point: its dependency to the proprietary implementations. Implementing a more suitable solution is hard, due to the closed source nature of its design. Indeed, there is often no access to the mechanical and inertial parameters of the robot and it is difficult to interface with the proprietary software provided by the robot manufacturer, or even by the robot cell integrator.

The automated fuselage structure assembly line of Hamburg Finkenwerder is presented in Fig.1.4.3, as an example of this type of heavy solution. The manipu-

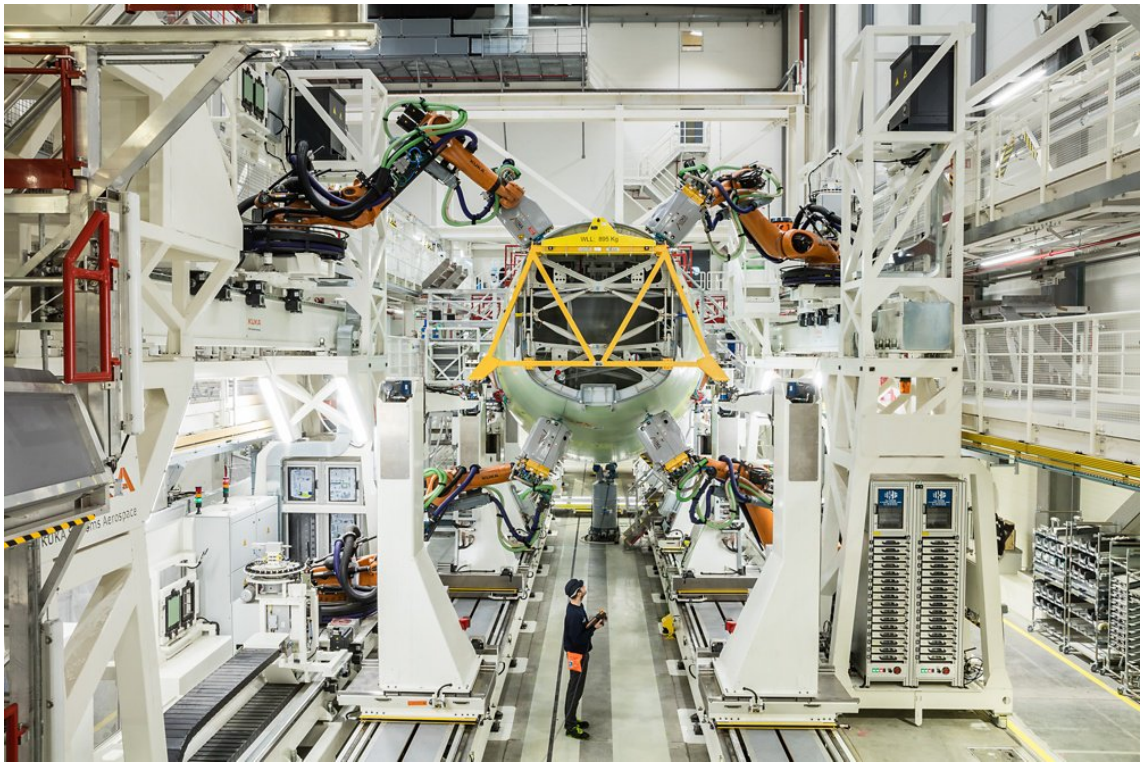

Figure 1.5: Airbus automated fuselage structure assembly line at Hamburg

lators join single fuselage shells into sections and assemble these sections into the final aircraft fuselages.

As an alternative to the typical aeronautic heavy drilling robotic cells, the MSDR has been designed by Airbus to be a medium size solution. The robotic arm of the platform was built by the company Fanuc (the $M-800$ model), see Fig.1.6. It is a 6 DoFs manipulator robot, it weights 820kg, has a payload of 60kg and is fixed on the ground. Similarly to the humanoid robot TALOS, the MSDR has double encoders, one on the motor side and one at the output of the harmonic drive. This secondary encoders is added to compensate eventual deflections or backlash on the axis. This simpler, lighter and cheaper solution has a work envelope of 2.2m and has been thought to realize autonomous manufacturing operations in areas where heavy robotic form factor does not fit or has reachability issues. An example of these areas is closed to the fuselage stiffeners, some holes need to be drilled but the end-effector is too big to correctly fit in the remaining space and accurately drill them.

The MSDR end-effector is equipped with several sensors to improve its capabilities to react to the environment: it has stereo-vision cameras, laser sensors, force sensors and an IMU. The laser sensors are used to measure distances in order to ensure the normality of the Tool Center Point (TCP). Moreover, because the MSDR has been designed with the Airbus specifications, the mechanical aspects and hardware are known. In addition, even if the software and programming language of Fanuc are proprietary, the software openness is high: the software implementations are indeed accessible. Moreover, the end effector behavior with in line path modification is left to the Airbus team, which can thus freely implement the desired behavior (under the constraint of the programming language of the robot).

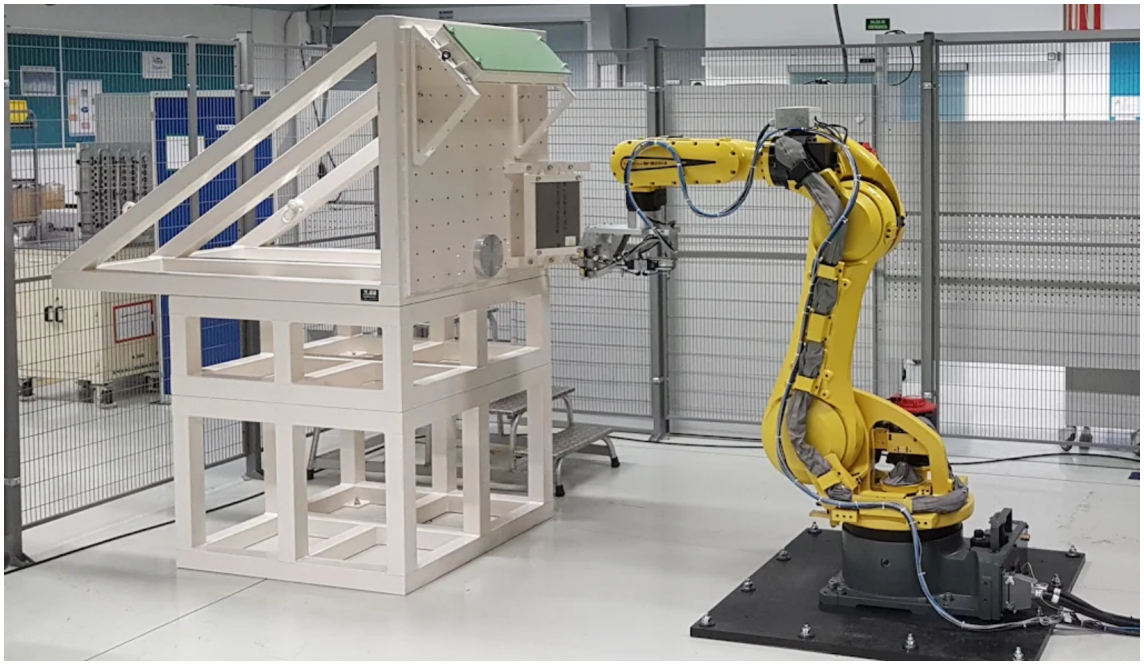

Figure 1.6: Middle Size Drilling Robot.

From TALOS to the MSDR, the gap is more important than from TALOS to TIAGo. In particular, because the MSDR is fixed to the ground, it has no under-actuation part. However, it is possible to use the MSDR with an auxiliary axis mounted under its base (as it is the case with robots in Hamburg) to add redundancy to the robot. This new structure add an under actuated part of one DoF to the robot, but has not been considered during my thesis. Similarly than for TIAGo, the relevant part of the solution is the control of the arm, adding the guarantee to have a safe, stable and robust result (see Chapter 5). In addition, even if the Fanuc implementations are accessible, it will be necessary to create an interface between the proprietary software and our solution. Indeed, our controller needs to read the robot sensors and actuators values in real time (at least 1kHz), and to sends its commands to the actuators. A solution is discussed in the Chapter 6.

1.5 Scientific Context

The Gepetto team of the LAAS aims at analyzing and generating the motion of anthropomorphic systems. The key difficulty of motion generation and control of anthropomorphic systems comes from the redundancy of their tree-like structure with respect to most positioning tasks, the natural instability of the bipedal posture, and the under-actuation of their spatial displacement. The team studies this problem by following an interdisciplinary approach focused on three research objects: the humanoid robot, the numerical mannequin and the human body.

One objective is to provide humanoid robots with motion autonomy by developing methods and software for motion planning and control. As said before, humanoid robotics presents two main challenges: the complexity of the robot mechanical system (more than 30 DoFs) and the physical interaction between the robot and the

real world.

The research activities of the Gepetto team can be described in three levels:

- ▷ The *fundamental research* level concerns theoretical developments related to system modeling and motion generation. The modeling part includes the mechanics of robotics systems, the mathematics of new representations and operators, and the recording and analysis of the human motion. The motion generation part ranges from the global trajectory planning to the local movement control (considered under different kinds of constraints).
- ▷ The *integration* level integrates the theoretical developments in software packages. These ones are maintained and made accessible to the whole robotics community (using standard formats and tools, see Section 1.6).
- ▷ The *application* level concerns the contribution to different domains such as assistive robotics, industrial robotics (specifically the factory of the future), actuator design, human movement imitation and understanding.

My thesis is part of the motion generation research studies of the team and in particular the control stage. For a given mission and a given robot, a control vector has to be computed and sent to the robot motors in a timely manner. This problem is in general complex. More precisely, considering the problem of moving to find an object in an unknown environment is $NP-hard$ Boyd and Vandenberghe [12]. The ROB4FAM joint-lab focuses on the following operational challenges:

- ▷ Climbing stairs with an heavy handled tool (designed for humans)
- ▷ Manufacturing tasks applied on a real structure such as drilling or screwing

Climbing stairs imposes to switch from one contact to another in order to make the humanoid robot TALOS move while handling an heavy object. This implies taking into account balance with multiple contacts, the geometry and the dynamical effect of the tool to handle. The robot has to take into account possible drift due to contact slip, spring back effect from the material part and vibrations when performing the tasks. On the other hand, drilling and screwing mostly involve whole body motion to perform a force control based behavior.

In order to produce a feasible motion, the robot has to decide where to interact with the environment to be able to move and perform its actions. Considering the climbing case, the robot has to locate where to put its feet and possibly its hand. Once this is decided, it needs to find a feasible motion for each of its bodies to generate the overall motion. When a feasible solution is found (i.e. avoiding collision, joint limits, ...) this solution needs to be executed on the robot. This is usually not trivial as discrepancies between the model and reality will affect the result. It is therefore necessary to modify these motions and to create a final control command for the robot in order to make sure that the plan is followed.

These three stages: global trajectory planning, local planning for control, control of the robot (illustrated in Fig. 1.7) are the ones considered by the Gepetto team to realize motion generation.

This approach is generic and allows to formulate solutions for different robots and scenarios. However, the way the problem is solved is dependant on the robot and the scenario, meaning that a solution cannot be reused on a different robot or

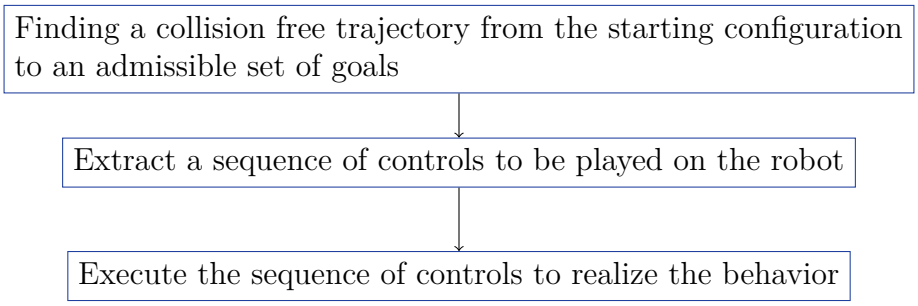

Figure 1.7: Overall approach of the motion generation problem (WP)

scenario, a specific solution will be needed. For instance, using the same problem formulation for stairs climbing and drilling will not work on both scenario; neither will the use of the same problem formulation for drilling using TALOS and the MSDR work. Nevertheless, they will be similar, because the approach uses generic formulation.

Let us focus on the example of the control formulations of an humanoid robot and a manipulator. The formulation used by the team is called the Stack-of-Tasks (SoT) and consists in realizing several **Tasks** at the same time to achieve a goal. These **Tasks** can for example be to move the end-effector at some desired position or to maintain the Center of Mass (CoM) at its position to keep the balance of the robot. Moreover, these **Tasks** can be prioritized to decide which action is more important than another (which can then be not perfectly achieved) or set as constraints, meaning that they must be fulfilled. In Fig. 1.8 is presented an example of the formulation to control a humanoid robot TALOS. Compared to the one for two robot manipulators shown in Fig. 1.9, one can see the similarities of the tasks involved. The h values represent the **Task** functions to be optimized by the control formulation to achieve the B desired values. In both cases the formulations involve **Tasks** on the CoM, on the arms-tips and on the feet-bases. Their prioritization is the same, with the most important **Tasks** at the bottom and the least one at the top. At the lowest is set a constraint as an example, the collision detection constraint to avoid the robot bodies to collide:

$$d(\mathcal{B}_i, \mathcal{B}_j) \geq \epsilon \tag{1.1}$$

for $(\mathcal{B}_i, \mathcal{B}_j) \in Collisions$ where $Collisions$ is the set of body pairs for which collisions should be checked.

Using the Gepetto team approach and tools allows this thesis to propose implementations of controllers for humanoid robots which are versatile and can be used for industrial robots.

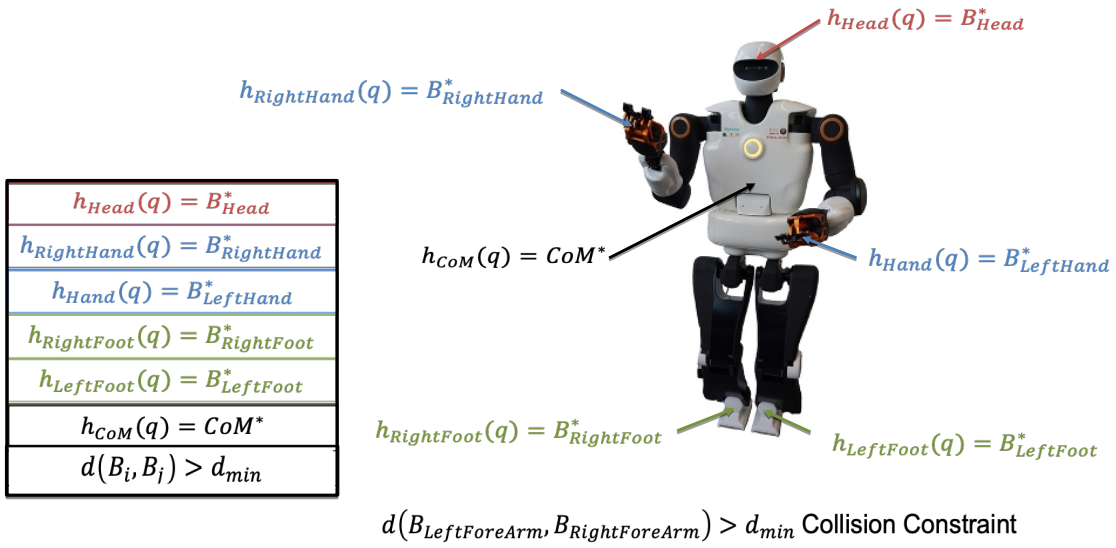

Figure 1.8: Parallel between the control architecture of a humanoid robot and manipulator robots: TALOS stack of tasks (WP).

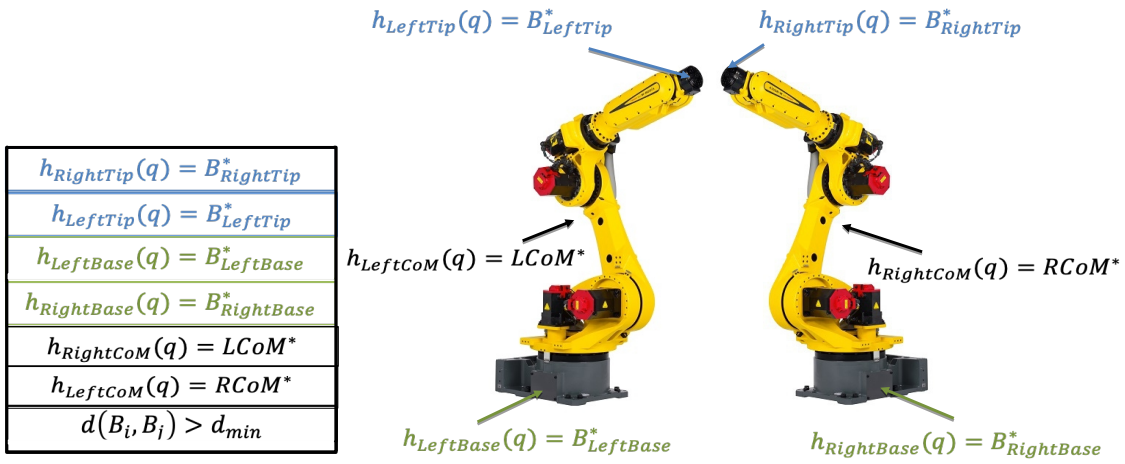

Figure 1.9: Parallel between the control architecture of a humanoid robot and manipulator robots: Manipulators stack of tasks (WP).

1.6 Frameworks and Libraries

In this section are quickly described the frameworks and libraries implemented by the Gepetto team to solve the problem of motion generation which have been used and expanded during my thesis. The advantage of these tools is that they are based on the same mathematical library Pinocchio Carpentier et al. [13], and can interact with each others using the dynamic-graph package [14].

1.6.1 Pinocchio Library

Pinocchio Carpentier et al. [13] is a library which instantiates the state-of-the-art Rigid Body Algorithms for poly-articulated systems based on revisited Roy Featherstone's algorithms Featherstone [15]. Pinocchio efficiently computes the dynamics (and derivatives) of a robot model, or of any articulated rigid-body model (avatars in a simulator, skeletal models for bio-mechanics, etc.). It provides the analytical derivatives of the main Rigid-Body Algorithms like the Recursive Newton-Euler Algorithm or the Articulated-Body Algorithm. The library is open-source, mostly written in C++ with Python bindings, and distributed under the BSD licence. It has been used in this thesis for all the computations of the rigid body kinematics and dynamics.

1.6.2 Stack-of-Tasks (SoT) Framework

The main framework used in this thesis is the SoT Mansard et al. [16], it is a C++ Software Development Kit implementing a control architecture for redundant robots and more specifically for humanoid robots. The SoT framework is a collection of software packages handled by *cmake* and *pkg-config*. The rigid body kinematics and dynamics is computed using the Pinocchio library Carpentier et al. [13].

1.6.2.a Dynamic Graph

The dynamic-graph package is used to connect computation nodes, *entities* together using a Graphical System Design (GSD), akin to what Simulink© does MathWorks [17]. *Entities* are connected through input and output signals. This package implements on-demand signal computation and an efficient caching mechanism to avoid useless data re-computation. This allows fast computation of signal values, which is a critical point for real-time control.

1.6.2.b ROS-Control SoT

This package encapsulates the SoT graph in the *ros-control* framework of the middleware ROS Stanford Artificial Intelligence Laboratory et al. [7]. The intent is to make it generic and adapted to any robot through *rosparam*. Thus, this package allows to have a *ros-control* architecture compatibility. It is particularly useful because the *ros-control* framework provides a generic API for controllers and robot hardware. For instance, one can choose to use the SoT controller or the one provided by the robots manufacturer, PAL-robotics, in the same way. Moreover, with this compatibility,

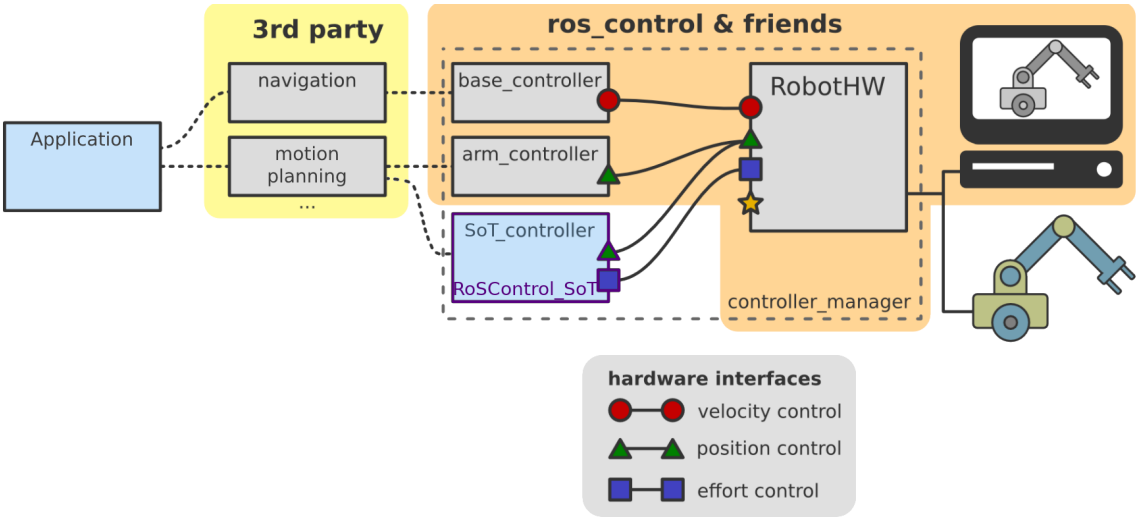

Figure 1.10: ROS-Control framework with the ROS-Control SoT interface

it is possible to use the Gazebo simulator of *ros-control*. The software developed for the real robot can then be tested in simulation with the same architecture. However, one issue of the generic ROS API is highlighted when specific components are used in the robot hardware. Indeed, if the actuators have specific sensors, such as temperature ones as the robot TALOS does, it is not handled by the architecture. Thus, the roscontrol-sot package [18] has been modified to take these sensors into account. This package is represented by the dark blue blocks in the SoT Framework scheme of Figure.1.11. It creates an abstract interface to communicate with the hardware interface of *ros-control*, its integration in the whole *ros-control* framework is displayed in Fig.1.10 in purple, based on [19].

1.6.2.c SoT Core

SoT-core [20] is the package in charge of the control architecture. The package implements hierarchical control using the task-based inverse kinematics described in Mansard et al. [16] (see Section 4.2). The robots can be loaded from their URDF model. The rigid body dynamics is provided through the Pinocchio library Carpentier et al. [13]. It has been used in this thesis to compare hierarchical position control to weighted position and torque control in Section 4.2.

1.6.2.d Overall scheme for real-time control

In the Fig.1.11 is presented the general SoT flow chart, where the **SoT-Core** purple blocks can be replaced by another task-based controller (for instance a controller using TSID) and the **Planner** block can be any planning algorithm giving the appropriate references. Real-time control system are usually driven by a cyclic computational node which needs to send a control reference value to each motors of a robot. To compute this control reference values, sensor values need to be provided. In the SoT, special *entities* called **Device** are used to provide an abstract interface to the hardware. The **Device** has specific inputs which contains the control vector. This control vector is the result of a computation solving a control problem,

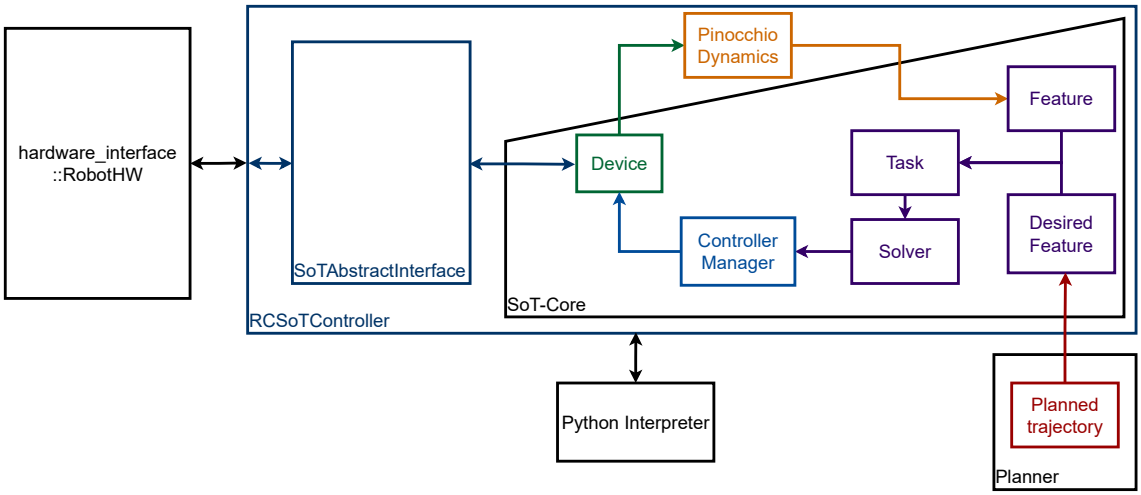

Figure 1.11: SoT framework

in the Fig.1.11, the **Solver** entity of SoT-core. It is an optimization problem built using a control **Task** which regulates the difference with a **Feature** and a **Desired Feature**. For instance the Center-Of-Mass (CoM) position of the robot. The **Feature** is computed using the robot model and the sensor values with Pinocchio. The **Controller Manager** entity checks that the result obtained by the solver does not exceed the limits of the control vector.

1.6.3 Task Space Inverse Dynamics (TSID) Library

TSID is a C++ library for optimization-based inverse-dynamics control based on the rigid multi-body dynamics library Pinocchio Carpentier et al. [13]. It implements a non-strict hierarchical control using tasks functions Prete et al. [21]. It has been used in this thesis to create the weighted position and torque control schemes in Section 4.2 and have been modified to include a passive constraint in Chapter 5.

1.7 Summary of the Chapters

The thesis is organized as follows: Chapter 2 presents a state-of-the-art on the different research domains encountered in this thesis. A particular attention is given to the whole-body control and the force control methods, which are the core of the thesis problem.

Chapter 3 details the first work achieved during this thesis toward the protection of the robotic system. It initially consists in identifying and modeling the robot chain drive and then in using it in a MPC to send safe torque commands. It has led to the publication Ramuzat et al. [22].

Chapter 4 presents the different works realized while investigating the differences and performances of whole-body control architectures for humanoid robots. Firstly, the developed control architectures during this thesis are detailed, a focus on the hip flexibility control of the robot TALOS can be found in Appendix 4. Moreover, the chosen force control to realize the manufacturing operations is explained. The

targeted application is the drilling of a metal plate, requiring high precision and the application of huge forces.

Secondly, three controllers are compared: one solving *torque* commands and two *position* commands. The aim is to choose the most appropriate scheme for the robot TALOS, these controllers are compared in terms of tracking error, energy consumption and computational time by using Gazebo simulations of the robot walking on flat horizontal ground, tilted platforms, and stairs. This work has led to the publication Ramuzat et al. [23].

Thirdly, the efficiency of the previous whole-body torque controller is exposed in a context of human-robot collaboration. The goal is to make the robot proactively walk alongside a human. The results have been obtained during a collaboration with another PhD student of the Gepetto team, Isabelle Maroger and have been published in Maroger et al. [24]. This work combines an optimal control prediction model of the human trajectory, a Walking Pattern Generator (WPG) based on non-linear MPC and the previous real-time whole-body torque controller.

Finally, the last section describes the experiments realized on the real robot, using the presented controllers. This section discusses the different difficulties encountered and the approaches we tried to implement to solve them, leading to the publication Ramuzat et al. [25].

Chapter 5 details the second work realized during this thesis toward the safety of the robotic system. It describes a new passivity-based inverse dynamics (ID) controller, based on the torque controller of Chapter 4 and the passivity theory. The control approach allows to achieve a safe multi-contact scenario on a torque controlled humanoid robot, using a global energy tank. This work has led to the publication Ramuzat et al. [26].

Chapter 6 describes the expected challenges that will be encountered when transposing the obtained results from the TALOS robot to the MSDR or another industrial robot. It highlights the benefits and drawbacks of my complex and state-of-the-art solution compared to the classical proposal for manipulator robots.

Finally the last chapter presents the conclusions of my works and the perspectives and future aims.

1.8 Publications

Journal

Maroger et al. [24] I. Maroger, N. Ramuzat, O. Stasse and B. Watier, "Human Trajectory Prediction Model and Its Coupling With a Walking Pattern Generator of a Humanoid Robot". In *IEEE Robotics and Automation Letters* with *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* option, October 2021, vol. 6, no. 4, pp. 6361-6369.

Ramuzat et al. [25] N. Ramuzat, O. Stasse and S. Boria, "Benchmarking Whole Body controllers on TALOS". In *Frontiers Robotics and AI*, January 2022.

Ramuzat et al. [26] N. Ramuzat, S. Boria and O. Stasse, "Passive Inverse Dynamics Control using a Global Energy Tank for Torque-Controlled Humanoid Robots

in Multi-Contact". In *IEEE Robotics and Automation Letters* with *IEEE/RAS Int. Conf. on Robotics and Automation (ICRA)* option, April 2022, vol. 7, no. 2, pp. 2787-2794.

Conference

Ramuzat et al. [22] N. Ramuzat, F. Forget, V. Bonnet, M. Gautier, S. Boria and O. Stasse, "Actuator Model, Identification and Differential Dynamic Programming for a TALOS Humanoid Robot". In *IFAC - 2020 European Control Conference (ECC)*, May 2020, pp. 724-730.

Ramuzat et al. [23] N. Ramuzat, G. Buondonno, S. Boria and O. Stasse, "Comparison of Position and Torque Whole Body Control Schemes on the Humanoid Robot TALOS". In *20th International Conference on Advanced Robotics (ICAR)*, December 2021, pp. 785-792.

Chapter 2

State of the Art

Contents

In this chapter, some parts of the state-of-the-art presented were elements of the works realized for the Work-Package (WP) reports in the scope of the joint-laboratory ROB4FAM. Some of the figures presented are thus indicated of part of these reports (WP). The presented section on human-robot collaboration is part of the paper [24] realized in collaboration with Isabelle Maroger. Moreover, Louise Scherrer has kindly authorized us to use some of her figures from her master report Scherrer [1] to illustrate some principles and the Walking Pattern Generator of Naveau et al. [27].

2.1 Humanoid Robot Model

2.1.1 Robot Model

Our experimental platform is a robot TALOS from the PAL-Robotics company, it is 1.75m tall weighting 100kg and has electric actuators. It has $n_j = 32$ joints and a floating base of $n_b = 6$ DoFs, we denote $n = n_j + n_b$, and define the configuration space, called **C**-space. This space can be decomposed in three parts, leading to the following representation:

$$\begin{aligned} \mathbf{C} &= \mathbf{C}_{position} \times \mathbf{C}_{orientation} \times \mathbf{C}_{joints} \\ \mathbf{C} &= \mathbb{R}^3 \times SO(3) \times \mathbf{Q} \end{aligned} \tag{2.1}$$

where $\mathbb{R}^3$ represents the coordinates of origin of the floating base frame B and $SO(3)$ is the Lie group representing the rotation matrices that define the orientation of this frame relative to the world frame W. $\mathbf{Q}$ is the subspace of the n_j-joint space, based on the Denavit-Hartenberg Bejczy [28] representation of the joint angles $q_j \in \mathbb{R}^n_j$.

We express the coordinates of the robot $q \in \mathbf{C}$ as:

$$q = \begin{bmatrix} x_b \\ R_b \\ q_j \end{bmatrix} \tag{2.2}$$

with the under-actuated coordinates of the floating base $b = [x_b\, R_b]^T \in \mathbb{R}^3 \times SO(3)$ (x_b the position and R_b the orientation of the base frame B relative to the world frame W) and the Denavit-Hartenberg Bejczy [28] representation of the joint angles $q_j \in \mathbb{R}^n_j$.

Then, we can express the velocity of the robot v, composed of the linear velocity of the floating base (the derivative of the floating base position), the angular velocity of the floating base (which is **not** the derivative of the floating base orientation) and the joint angular velocities of the robot (the derivatives of q_j). Thus, v is not the direct derivative of q, the two vectors does not have the same size, as R_b is a rotation matrix or a quaternion of size 4, whereas the angular velocity of the floating base is a angular velocity vector of size 3. We denote v as:

$$v = \begin{bmatrix} \dot{x}_b \\ \omega_b \\ \dot{q}_j \end{bmatrix} \tag{2.3}$$

with the under-actuated velocities of the floating base $[\dot{x}_b\, \omega_b]^T \in \mathbb{R}^3 \times \mathbb{R}^3$ and the joint angular velocities of the robot $\dot{q}_j \in \mathbb{R}^{n_j}$. There exist a mapping between the two representations $\mathrm{M} : [x_b\, R_b] \mapsto [x_b\, \theta_b]$ with $\theta_b \in \mathbb{R}^3$ ($\dot{\theta}_b = \omega_b$). The acceleration of the robot a is then simply the derivative of v.

The equation of the robot dynamics can be written as:

$$M(q)a + \underbrace{C(q,v)v + g(q)}_{h} = \underbrace{\begin{bmatrix} 0 \\ 0 \\ \tau \end{bmatrix}}_{N^T\tau} + \underbrace{\tau_{ext}}_{J_c^T F} \tag{2.4}$$

$M \in \mathbb{R}^{n\times n}$ the symmetric and positive definite inertia matrix, $C \in \mathbb{R}^{n\times n}$ the Coriolis matrix and $g \in \mathbb{R}^n$ the gravity vector. $q_j \in \mathbb{R}^{n_j}$ is the joint configuration of the robot. $a, v, q \in \mathbb{R}^n$ are the accelerations, velocities and positions of the joint configuration of the robot including the base (free-flyer). The free-flyer information are estimated with a base-estimator from the configuration, IMU and force sensors of the robot (we use in the thesis the one persented in Flayols et al. [29]). $\tau \in \mathbb{R}^{n_j}$ are the joint torques of the actuators and $\tau_{ext} \in \mathbb{R}^n$ are the external torques. N is a selector matrix associated to the actuated joints $N = [0_{n_b}, 1_{n_j}]$, such that $N^T \in \mathbb{R}^{n\times n_j}$.

A notable property of the dynamic model is that the choice for the Coriolis matrix C is not unique. A particular choice is to define this matrix such that: $\frac{1}{2}\dot{M}(q) = \frac{C(q,v) + C(q,v)^T}{2}$ Siciliano et al. [2]. Indeed, with this formulation, the matrix $\dot{M}(q) - 2C(q,v)$ is skew-symmetric, thus their quadratic form is null, i.e. for any vector $x \in \mathbb{R}^n$:

$$x^T[\dot{M}(q) - 2C(q,v)]x = 0 \tag{2.5}$$

It is an important property that is often used for simplifications.

Using the force contacts F, we can write that $\tau_{ext} = J_c^T F = \sum_{i=1}^{n_c} J_i^T F_i$ with n_c the number of contacts and J_i the Jacobian of the contact point x_c^i (with the contact frame V_i) according to the robot coordinate vector q:

$$\begin{aligned} J_i &= [J_u^i \;\; J_a^i] \\ &= \left[\mathcal{T}(V_i, B) \;\; \frac{\partial x_c^i}{\partial q_j}\right] \end{aligned} \tag{2.6}$$

where J_u^i is the "Jacobian" of the under-actuated part: it is the transformation $\mathcal{T}$ between the contact frame and the base frame Siciliano et al. [2]. And J_a^i is the classical Jacobian of the actuated part.

By denoting $\dot{q}_j, \ddot{q}_j$ the velocity and acceleration of the joints and $\dot{x}_b, \omega_b$ the linear and angular velocities of the floating base frame we can decompose Eq.2.4, to have Henze et al. [30]:

$$M(q) \underbrace{\begin{bmatrix} \ddot{x}_b \\ \dot{\omega}_b \\ \ddot{q}_j \end{bmatrix}}_{a} + C(q,v) \underbrace{\begin{bmatrix} \dot{x}_b \\ \omega_b \\ \dot{q}_j \end{bmatrix}}_{v} + g(q) \;=\; N^T\tau \;+\; \tau_{ext} \tag{2.7}$$

If all the external forces act at the end-effector frames V_i with $i = 1...\psi$, with ψ the number of end-effector frames, then we can write:

$$\tau_{ext} = \sum_{i=1}^{\psi} \begin{bmatrix} \begin{bmatrix} I & 0 \\ \hat{x}_{bi} & I \end{bmatrix} \\ J_a^{iT} \end{bmatrix} F_i \tag{2.8}$$

with J_a^i the actuated Jacobian matrix of the end-effector i making the contact point x_i (see Eq.2.6), $F_i = [f_i^T \; \tau_i^T]^T$ the effector wrench. $\hat{x}_{bi}$ is the cross-product matrix of the vector $x_{bi} = x_i - x_b$: the configuration-dependent lever arm between the end-effector frame V_i and the base frame B (defining $J_u = \mathcal{T}(V_i, B)$ the transformation of Eq.2.6).

For balancing it is interesting to replace the floating base frame by the CoM one which has the same orientation as the base link. The equations are similar with $c, w_{com}, x_{com,i}$ instead of x_b, ω_b, x_{bi} and using the transformation matrix T:

$$\begin{bmatrix} \dot{c} \\ \omega_{com} \\ \dot{q}_j \end{bmatrix} = \underbrace{\begin{bmatrix} I & -\hat{x}_{b,com} & J_{b,com} \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix}}_{T} \begin{bmatrix} \dot{x}_b \\ \omega_b \\ \dot{q}_j \end{bmatrix} \tag{2.9}$$

We obtain the following simplified equation Wieber et al. [31], denoting $v_{com} = [\dot{c}^T \ \omega_{com}^T]^T$:

$$M \begin{bmatrix} \dot{v}_{com} \\ \ddot{q}_j \end{bmatrix} + C \begin{bmatrix} v_{com} \\ \dot{q}_j \end{bmatrix} + \begin{bmatrix} m\, g_0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ \tau \end{bmatrix} + \tau_{ext} \tag{2.10}$$

We have $M(q) \in \mathbb{R}^{(n_j+n_b) \times (n_j+n_b)}$, $C(q,v) \in \mathbb{R}^{(n_j+n_b) \times (n_j+n_b)}$, $\tau \in \mathbb{R}^{n_j}$ and $\tau_{ext} \in \mathbb{R}^{(n_j+n_b)}$. Here one can notice that the gravity vector g is projected as the robot weight in its base frame. It is a major difference compared to fixed based robots where this simplification is not possible.

2.1.2 Contacts properties

Using friction model of Coulomb Kajita et al. [32], the slippage is avoided when Henze et al. [30], Ramos et al. [33]:

$$\| f_{\parallel} \| \ \leq \ \mu \, |f_{\perp}| \tag{2.11}$$

with $f_{\parallel}$ and $f_{\perp}$ the tangential and normal components of the contact forces (in our case for each of the i^{th} end-effector). μ is the coefficient of friction, an empirical property of the contacting materials. Similarly a constraint can be added on the torque to limit the friction:

$$\| \tau_{\parallel} \| \ \leq \ \tilde{\mu} \, |\tau_{\perp}| \tag{2.12}$$

To avoid the interpenetration of the end-effector and the surface at the contact points x_c (for instance between the foot and the ground) the Signorini conditions have to be met, called the complementarity constraints:

$$\begin{array}{rcl} f_{\perp} & \geq & 0 \\ \ddot{x}_{c\perp} & \geq & 0 \\ \ddot{x}_{c\perp} \, f_{\perp} & = & 0 \end{array} \tag{2.13}$$

These equations can be simplified by fulfilling the condition of positivity to guarantee the contact:

$$\begin{array}{rcl} f_{\perp} & \geq & f_{\perp}^{min} \\ f_{\perp}^{min} & \geq & 0 \end{array} \iff \begin{array}{rcl} \ddot{x}_{c\perp} & = & 0 \\ J_c \, a + \dot{J}_c \, v & = & 0 \end{array} \tag{2.14}$$

because $\dot{x}_{c\perp} = J_c \, v$, with J_c the Jacobian matrix of the contact point x_c (as defined in Eq.2.6).

Finally to prevent the end-effector from tilting, the Center of Pressure (CoP) $p \in \mathbb{R}^3$, can be restricted as follow to lie in the contact surface:

$$p_{\perp} \ \in \ \left[p_{\perp}^{min} \, , \, p_{\perp}^{max} \right] \tag{2.15}$$

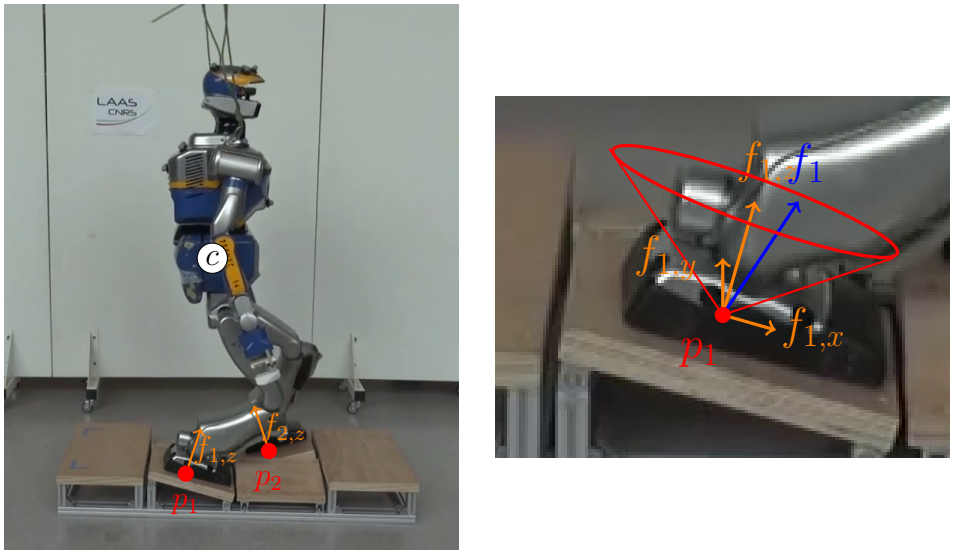

Figure 2.1: (left) The HRP-2 humanoid robot walking on a non-flat terrain. On each contact p_i a wrench (f_i, τ_i) is applied. For sake of clarity all the components are not represented. This figure is part of the WP reports.
(right) A graphical representation of the Coulomb Friction constraint applied to the right foot of the humanoid robot with $f_1 = [f_{1,x}\ f_{1,y}\ f_{1,z}]^T$ and $\mu f_{1,z}^2 > f_{1,x}^2 + f_{1,y}^2$ (Eq.2.11).

These constraints define the allowed force at the contact points, which represents a friction cone defined by an axis along the surface normal and a semi-angle $\theta = \arctan \mu$ (see Fic.2.1 for an illustration of the friction cone).

2.1.3 Task Space and Workspace

It is common to distinguish two spaces from the configuration space of the robot (**C**-space linked to q). These spaces are denoted the **task space** and the **workspace** (usually defined for the end-effectors of the robot).

The task space is a space in which a task performed by the robot can be naturally expressed. For instance, to control the 3D position of the CoM of the robot, the task space would be $\mathbb{R}^3$. However, if one want to control the posture of the robot (the positions of each joint angles), then the task space corresponds to the configuration space. The task space is linked to the configuration space with the Jacobian matrix associated to the task Orin et al. [34]. One can switch between one space to another using the relation: $\dot{x} = J_a \dot{q}_j$ where x is the task point position in the task space, $\dot{q}_j$ the velocities of the joint configuration and J_a the Jacobian of the task point according to the robot state vector $J_a = \frac{\partial x}{\partial q}$. Using the Eq.2.6, we can also define the relation $\dot{x} = Jv$ with v the velocity vector of the robot including its floating base.

The workspace defines the reachability space of the robot, i.e the set of configurations reachable by the robot end-effectors. This definition is independent of any task and is characterized by the robot structure.

A point in the task space or in the workspace may be achievable by more than one

robot configuration, meaning that the point is not a full specification of the robot's configuration. Conversely, some points in the task space may not be reachable at all by the robot. By definition, however, all points in the workspace are reachable by at least one configuration of the robot Lynch and Park [35].

In this thesis, we use and focus on solutions designing the desired motions of the robot in the task spaces rather than in the configuration one. This approach is called the task-function approach Khatib [36], Samson et al. [37], Escande et al. [38]. The objectives to be performed by the robot are expressed in their respective task spaces, using reference trajectories given by the motion planning (for instance the CoM, end-effectors, feet or head desired trajectories). The planning algorithm computes references respecting the reachability space of the robot.

2.2 Centroidal Dynamics

The under-actuated part of the whole-body dynamics of a robot is called the centroidal dynamics, that is the dynamics of the CoM. It is possible to project the entire robot dynamics on it. Considering the robot as a rigid body, one can use the Newton-Euler equations of motion to couple the variations of the centroidal momentum with the contact forces Orin et al. [34] (using Eq.2.10):

$$\begin{cases} m\ddot{c} & = & \sum_i f_i + m\overline{g} & = & \dot{l}_c \\ mc_\times(\ddot{c} - g) + \dot{L} & = & \sum_i (p_i - c_i) \times f_i + \tau_i & = & \dot{k}_c \end{cases} \tag{2.16}$$

with $c, \dot{c}, \ddot{c}$ the CoM position, velocity and acceleration, $\dot{L} = \sum_k [R_k I_k \dot{w}_k - R_k (I_k w_k)_\times w_k]$ and $\overline{g} = [0,\ 0,\ -9.81]^T$, where $R_k \in SO(3)$ is the rotation matrix between the k^{th} body frame and the inertial coordinate frame, I_k its inertial matrix, w_k its angular velocity, m is the mass of the robot, $f_i \in \mathbb{R}^3$ the vector of contact forces at contact point i, $p_i \in \mathbb{R}^3$ their positions and $\tau_i \in \mathbb{R}^3$ their contact torque (represented at the inertial coordinate frame). l_c and $k_c \in \mathbb{R}^3$ are the linear and angular momentum around the CoM. The operator $\times$ denotes the cross product between two terms.

Solving this problem can be complex mostly due to the term $c_\times \ddot{c}$. Because of this term, the dynamics of the system is not linear.Over a receding horizon this term generates polynomials which might be either convex or concave. In general solving such problem is known to be $NP - Hard$ Boyd and Vandenberghe [12].

Several questions need to be answered to make a robot evolve on any terrain based *only* on these equations:

- ▷ How to choose the contact location $p_i \in \mathbb{R}$?
- ▷ What are the CoM trajectory $c \in \mathbb{R}^3$ and the equivalent forces trajectories fulfilling Eq.2.16 ?
- ▷ How to take into account constraints on $\dot{L}, p_i, f_i$?

In order to simplify the problem one might consider the resulting wrench of the external forces f_{ext} acting on the CoM and its application point p, called the CoP.

$$\begin{cases} m\ddot{c} & = & f_{ext} + m\overline{g} \\ mc_\times(\ddot{c} - \overline{g}) + \dot{L} & = & p_\times f_{ext} + \tau_{ext} \end{cases} \tag{2.17}$$

with $c = [c_x\ c_y\ c_z]^T$, $\ddot{c} = [\ddot{c}_x\ \ddot{c}_y\ \ddot{c}_z]^T$, $\dot{L} = [\dot{L}_x\ \dot{L}_y\ \dot{L}_z]^T$.
This is equivalent to pose:

$$\begin{aligned} f_{ext} &= \sum_{i=0}^{n_c} f_i \\ p_{\times} f_{ext} &= \sum_{i=0}^{n_c} p_i \times f_i \end{aligned} \tag{2.18}$$

This model is also called the free wheel model, or the inertia wheel model.

2.2.1 Zero Moment Point

The Zero Moment Point (ZMP) has first been introduced in Surla et al. [39]. In 2D, the ZMP is a point where the moment of the ground reaction force is zero (giving the name of the point), while in 3D it specifies the point where the total of horizontal inertia and gravity forces is null Kajita et al. [40]. The concept assumes planar contact area ($p_z = 0$) with high friction to avoid feet sliding.

The ZMP definition is linked to the support polygon one, which corresponds to a convex hull, the smallest convex set including all the contact points. By definition, the ZMP is the application point p when p is enclosed in the support polygon, otherwise p still exists (it is the CoP) but the system is not guaranteed to be in dynamic equilibrium.

In Fig. 2.2 is presented an illustrative example with the robot standing on flat floor with two contact points: p_{left}, p_{right}. They correspond to the application points of the left and right force resultants (in their friction cones). The resulting wrench of the external forces acting on the CoM is applied on a point inside the support polygon, it is thus the ZMP.

2.2.2 Linear Inverted Pendulum Model

Let us consider the total wrench applied to the robot and the ZMP p where it is applied. When a robot has its contact point on the floor we can set its altitude to zero, i.e. $p_z = 0$. Then, considering the previous model, it is equivalent to a robot with a flying wheel at the top of an inverted pendulum.

Thus Eq. 2.17 gives:

$$\begin{cases} -c_z\ddot{c}_y + c_y(\ddot{c}_z + g) + \dfrac{\dot{L}_x}{m} = p_y(\ddot{c}_z + g) \\ c_z\ddot{c}_x - c_x(\ddot{c}_z + g) + \dfrac{\dot{L}_y}{m} = -p_x(\ddot{c}_z + g) \\ c_x\ddot{c}_y - c_y\ddot{c}_x + \dfrac{\dot{L}_z}{m} = p_x\ddot{c}_y - p_y\ddot{c}_x \end{cases} \tag{2.19}$$

In general, to simplify the problem, $\dot{L}$ is neglected: the moment induced by the limbs of the robot on its CoM is considered much smaller then the one induced by the contact forces Kajita et al. [32].

The Linear Inverted Pendulum Model (LIPM), introduced in Kajita et al. [41], can be obtained by considering an artificial constraint on the motions of the pendulum. That is, the motions of the center of mass of the biped robot is limited to a

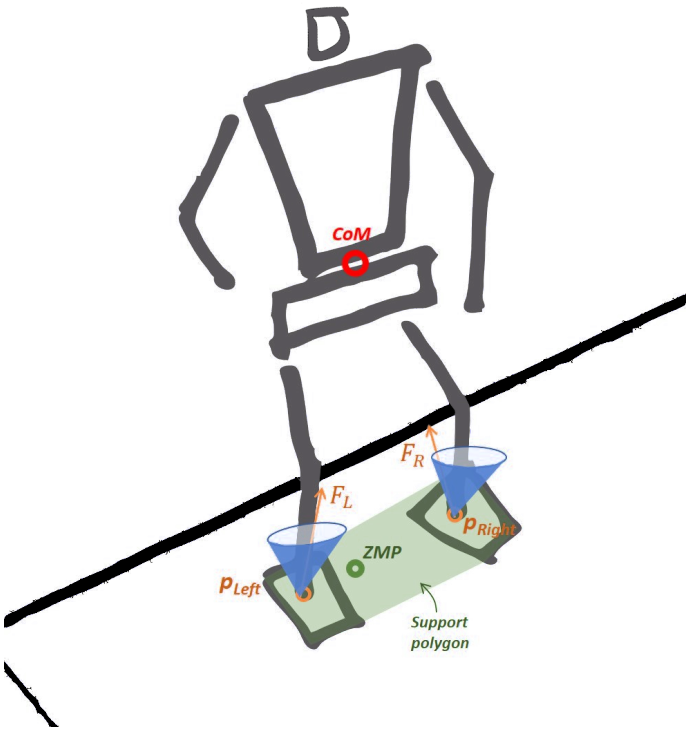

Figure 2.2: Scheme of the contact forces at points p_i for each foot and their contact wrench cones and the ZMP in the support polygon. Scherrer [1]

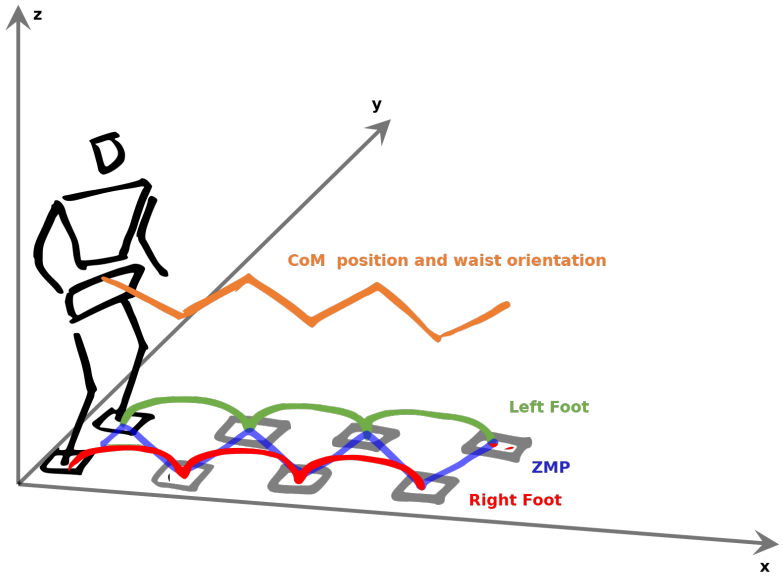

Figure 2.3: Link between the ZMP (in blue), the support contact positions (in green and red) and the CoM (in orange) Scherrer [1]

plan whose normal vector is $\begin{bmatrix} k_x & k_y & -1 \end{bmatrix}^T$ and z intersection is z_c. This constraint can be formulated as :

$$c_z = k_x c_x + k_y c_y + z_c \tag{2.20}$$

and by successive derivation :

$$\ddot{c}_z = k_x \ddot{c}_x + k_y \ddot{c}_y \tag{2.21}$$

To retrieve the equations of the LIPM of Kajita et al. [32] it is common to simplify the problem by considering the planar constraint to be horizontal, i.e. $k_x = k_y = 0$ and then $\ddot{c}_z = 0$. In addition to that hypothesis, the part of Angular Momentum (AM), depicted as $\dot{L}$ previously, is neglected in front of the influence of the center of mass motion (induced by the contact forces).

Thus, from the eq. 2.19, the following linear relationship can be written between the CoM and the ZMP:

$$\begin{cases} p_x = c_x - \dfrac{z_c}{g}\ddot{c}_x = c_x - \dfrac{c_z}{g}\ddot{c}_x \\ p_y = c_y - \dfrac{z_c}{g}\ddot{c}_y = c_y - \dfrac{c_z}{g}\ddot{c}_y \end{cases} \tag{2.22}$$

It is important to notice that the two equations are *decoupled*. Therefore, using the LIPM formulation, one can solve *separately* the equations along the x and the y axis.

Since the ZMP position is imposed by the contact forces and contact positions, one can use the LIPM equations to deduce the CoM trajectory from the ZMP one (see Fig.2.3 for an illustration). This is a common implementation in WPG Naveau et al. [27], Kajita et al. [32], Herdt et al. [42], creating the path of the CoM to respect the constraints on the contact forces, thus maintaining the contacts necessary for a balanced motion. Moreover, keeping the ZMP inside the support polygon with a certain margin is also often implemented in WPG, to obtain a safer balancing of the robot (reduce the risk of fall).

2.2.3 Equilibrium Criteria

2.2.3.a Divergent Component of Motion

We recall that the problem considers the planar constraint to be horizontal, i.e. $\ddot{c}_z = 0$ and thus $c_z = constant$. One can rewrite the eq.2.22 by noting $\omega = \sqrt{g/c_z}$ as follows:

$$\ddot{c}_x = \omega^2 (c_x - p_x) \tag{2.23}$$

And, setting $X = [c_x \ \dot{c}_x]^T$ in eq.2.23, one can obtain the following system of equations:

$$\dot{X} = \begin{pmatrix} 0 & 1 \\ \omega^2 & 0 \end{pmatrix} X + \begin{pmatrix} 0 \\ -\omega^2 p_x \end{pmatrix} = AX + b \tag{2.24}$$

This matrix A can be decomposed as :

$$A = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ -\omega & \omega \end{pmatrix}\begin{pmatrix} -\omega & 0 \\ 0 & \omega \end{pmatrix}\begin{pmatrix} 1 & -\frac{1}{\omega} \\ 1 & \frac{1}{\omega} \end{pmatrix} = \frac{1}{2} P \begin{pmatrix} -\omega & 0 \\ 0 & \omega \end{pmatrix} P^{-1} \tag{2.25}$$

It is then possible to make a change of variables:

$$\tilde{X} = \begin{bmatrix} \zeta_x \\ \\ \xi_x \end{bmatrix} = P^{-1}X = \begin{pmatrix} 1 & -\frac{1}{\omega} \\ \\ 1 & \frac{1}{\omega} \end{pmatrix} \begin{bmatrix} c_x \\ \\ \dot{c}_x \end{bmatrix} = \begin{pmatrix} c_x - \frac{\dot{c}_x}{\omega} \\ \\ c_x + \frac{\dot{c}_x}{\omega} \end{pmatrix} \tag{2.26}$$

The dynamics of the LIPM can therefore be divided in two: the upper part which is naturally converging, and the lower part which is diverging. The diverging part is called the Divergent Component of Motion (DCM), and its definition is given by:

$$\xi_x = c_x + \frac{\dot{c}_x}{\omega} \tag{2.27}$$

The DCM has been introduced first in Pratt et al. [43] under the name "Capture Point", while the presentation given here is the one provided by Takenaka et al. [44] from Honda, and popularized by the work of Englsberger et al. [45] on the humanoid robot TORO.

2.2.3.b CoM Admittance Control

The net contact wrench (that is the contact wrench expressed at the level of the CoM) can be computed indirectly by defining the ZMP position that will produce the said contact wrench. Under the assumptions of the LIPM, one can retrieve the following two coupled first-order equations Englsberger et al. [45], Takenaka et al. [46]:

$$\begin{array}{rcl} \dot{c} & = & \omega(\xi - c) \\ \dot{\xi} & = & \omega(\xi - p) \\ \xi & = & c + \frac{\dot{c}}{\omega} \end{array} \tag{2.28}$$

with p, ξ respectively the ZMP and DCM and $\omega = \sqrt{g/c_z}$.

These equations show that the DCM diverges from the ZMP, while the CoM converges to the DCM. Thus, the DCM can be controlled to control the CoM deviations and stabilize the system Englsberger et al. [45], Kajita et al. [47], Sugihara [48], Mesesan et al. [49]. This strategy is called the admittance control of the CoM.

From a CoM reference trajectory provided by a planner, a reference DCM can be defined as well. This reference, along with a measure of the DCM (based on the estimation of the base position of the robot) can be used in a Proportional–Integral (PI) controller. This controller has been presented in Caron et al. [50] (the integral term is used to eliminate the steady-state error):

$$\dot{\xi} = \dot{\xi}^* + K_{Pdcm}(\xi^* - \xi) + K_{Idcm} \int (\xi^* - \xi) \tag{2.29}$$

with ξ^* the desired DCM given by the planning, ξ the estimated DCM and K_{Pdcm}, K_{Idcm} the proportional and integral gains. In terms of ZMP (see Eq.2.28), the obtained control law is Caron et al. [50]:

$$p^* = p^{ref} - \left[1 + \frac{K_{Pdcm}}{\omega}\right](\xi^{ref} - \xi) + \frac{K_{Zdcm}}{\omega}(p^{ref} - p) - \frac{K_{Idcm}}{\omega} \int (\xi^{ref} - \xi) \tag{2.30}$$

with p^{ref}, ξ^{ref} the respective ZMP and DCM reference values, given by the planning. And K_{Zdcm} the proportional gain on the ZMP.

Finally, this desired ZMP is used into a CoM admittance control as Caron et al. [50]:

$$\ddot{c}^* = \ddot{c}^{ref} + K_{Padm}(p - p^*) \tag{2.31}$$

with K_{Padm} the proportional gain on the ZMP for the admittance control. The two position control schemes presented in the Section 4.4 use this stabilization formulation.

2.2.4 Angular Momentum

As said before, in the LIPM formulation, the Angular Momentum (AM) $\dot{L}$ is neglected. However, when a human dynamically walks, its AM is non-null and keeping it to 0 can lead to instability. In Kajita et al. [51], one of the objective is to consider the AM part generated by the contact transition. A controller may thus aim to have a task regulating the AM of the robot. Indeed, commonly a posture task is defined to keep the uncontrolled joints of the robot at fixed positions, in particular on the upper-body. However, this does not allow motions of the arms to help the walk. Using an AM task fixes this issue, as the controller solution to generate this momentum is in general making the robot arms move Lee and Goswami [52].

Using the equation Eq.2.16, the centroidal dynamics is therefore defined by $h_c = [l_c \ k_c]^T \in \mathbb{R}^6$. In Wensing and Orin [53], the task formulation of the centroidal dynamics control is given by $h_c = A_G(q_j)\dot{q}_j$ where $q_j, \dot{q}_j$ are the joint position and velocity vectors of the robot and A_G is the Centroidal Momentum Matrix Orin et al. [34].

The linear and AM tasks dynamics are given by the following equations:

$$\begin{cases} \dot{l}_c &= m\left[\ddot{c}^* + K_{Dcom}(\dot{c}^* - \dot{c}) + K_{Pcom}(c^* - c)\right] \\ \dot{k}_c &= \dot{k}_c^* + K_{Pam}(k_c^* - k_c) \end{cases} \tag{2.32}$$

with $\dot{l}_c$ and $\dot{k}_c$ the commanded rates of change of the linear momentum and angular momentum. K_{Pcom}, K_{Dcom} the proportional and derivative gains of the linear momentum task (CoM position) and K_{Pam} the proportional gain for the AM task. The AM task has been successfully implemented in Lee and Goswami [52] and used in the Section 4.4 in the controllers based on TSID. Another formulation of this task can be obtained by expressing the robot dynamics equation at the centroidal momentum level in the Quadratic Programming (QP) Koolen et al. [54].

2.3 Actuator Model

In this part is described briefly how the dynamic model of electric actuators is formulated in the literature. Part of this description will be used in the Chapter 3 to model the elbow chain actuation of TALOS. An actuator controls a motor which is followed by a gear reduction system (often a wave generator), itself fixed to a link, see Fig.2.3 for the TALOS actuator. The motor delivers forces or torques that cause the robot's links motion.

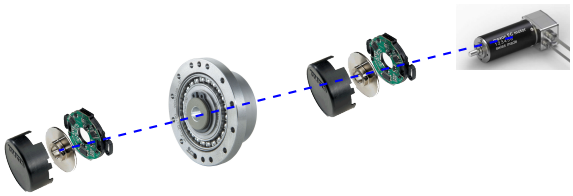

Figure 2.4: TALOS Actuator illustration: two modular-magnetic-encoders surround the AC-motor connected to the Harmonic Drive. This figure is part of the WP reports.

2.3.1 General formulation

The overall robot dynamic and its actuators are governed by the following equations Albu-Schäffer et al. [55]:

$$\begin{array}{lcl} Ma + Cv + g & = & \tau + DK^{-1}\dot{\tau} + \tau_{ext} \\ I_m\ddot{\theta}_j + \tau + DK^{-1}\dot{\tau} & = & \tau_m - \tau_f \\ \tau & = & K(\theta_j - q_j) \end{array} \tag{2.33}$$

with $q_j \in \mathbb{R}^{n_j}$, $\theta_j \in \mathbb{R}^{n_j}$ the link and motor side positions respectively. The gear reduction ratio is defined as $r_j = \frac{\theta_j}{q_j}$ for actuator j. High ration reduces the coupling effect between the joints W. Khalil [56]. τ_m is the motor torque. We recall that $v = [\dot{x}_b, \omega_b, \dot{q}_j]^T$ with $(\dot{x}_b, \omega_b)$ the linear and angular velocity of the robot base, $\dot{q}_j$ the velocity of the actuated joint. τ_f represents the friction (see the following section). $K = diag(k_i) \in \mathbb{R}^{n_j \times n_j}$ the diagonal, positive definite joint stiffness matrix and $D = diag(d_i) \in \mathbb{R}^{n_j \times n_j}$ the diagonal, positive semi-definite joint damping matrix. $I_m = diag(i_i) \in \mathbb{R}^{n_j \times n_j}$ is the matrix of rotor inertia.

In general, for rigid robot, $K \to \infty$, thus the last equation of Eq.2.33 is not taken into account and the positions of the joint and the motor are considered equal $(\theta = q_j)$. And then $K^{-1} \to 0$, modifying the first equation of Eq.2.33.

2.3.2 Friction

A simple approximation of the frictions τ_f model can be expressed the following way:

$$\tau_f = F_v\dot{q}_j + F_s\ sign(\dot{q}_j) \tag{2.34}$$

where $sign(.)$ is the function returning the sign of a variable.

Two kinds of frictions can take place all over the mechanisms: viscous frictions F_v and dry frictions F_s. This model has some short comings in the sense that it creates oscillations when the speed is close to zero. In this case, Coulomb frictions might appear and when they are not symmetric, leads to hysteresis.

2.3.3 Parameters Identification

Parameters identification of the actuator model consist in identifying all the parameters previously defined as I_m, D, K, F_v, F_s but also:

- ▷ The geometrical parameters of the robot,
- ▷ the gear reduction ratio r_j for actuator j, defined by $r_j = \frac{\theta_j}{q_j}$,
- ▷ the motor constant relating current i_j to motor torque $\tau_{m,j}$ for each actuator j such that $\tau_{m,j} = K_{m,j} i_j$.
- ▷ the Center-Of-Mass position for each link c_j,

This is realized by generating trajectories which are maximizing the observability of the parameters W. Khalil [56], called optimal exciting motions (OEM) Bonnet et al. [57]. On TALOS it is possible to measure the joint positions, the motor positions, the current and the external torque $(q_j, \theta, i, \tau_{ext})$. From this, the gear ratio r_j and the motor constant $K_{m,j}$ can be identified.

For I_m, D, K, F_v, F_s a common approach is to solve a Least Square Problem (LSP), based on Eq.2.33. Note that most of the non-linear parts of the equations are related to quantities which can be measured. Therefore while the equations are non-linear, the Least Square Problem is not. More precisely all the parameters n_{par} (for each actuator) to be identified are put together in a vector of free variables called $X \in \mathbb{R}^{n \times n_{par}}$. All the measurements n_{meas} are stack together to obtain a vector Y. The theoretical relationship is then $Y = AX$ with $A \in \mathbb{R}^{n_{meas} \times n_{par}}$. A large number of measurements will make this equality over-constrained, thus to identify the parameters, the following LSP is solved Gautier and Jubien [58]:

$$min_X ||Y - AX|| \tag{2.35}$$

When the measured variables are not precise or compromised, it is possible to use a Ransac (RANdom SAmple Consensus) algorithm or another iterative method to still estimate the parameters of the model.

2.3.4 Actuator Control

In general, the whole body controller provides a control vector u to the robot in two possible forms (see Section 2.5):

- ▷ q_j^*, $\dot{q}_j^*$, or $\ddot{q}_j^*$ a desired position or velocity or acceleration for each actuator,
- ▷ τ^* a desired torque for each joint.

However, the real control variable that is provided by the power electronics of the robot is the motor current. Thus, one have to add a layer of control creating a relationship between the output of the controller and the motor current Lynch and Park [35]. For the first form of u one can use a position control, while for the second form one will need to implement a torque control. In the following sections, for each motor j the current is noted i_j.

2.3.4.a Position control

The position control is usually done by a simple Proportional Integral Derivative (PID) embedded in the electronic board of the robot:

$$i_j = P_j(q_j^* - q_j) + I_j \int_0^t (q_j^*(h) - q_j(h))dh + D_j(\dot{q}_j^* - \dot{q}_j) \tag{2.36}$$

with q_j^*, q_j the desired and measured joint j positions, P_j the proportional gain, D_j the derivative gain and I_j the integral one. Proportional gains for this kind of control are typically very high (10000 for P_j), with I_j often very small (0.1) or equal to zero. This is making the robot extremely rigid, but it is quite simple to deploy.

On TALOS and TIAGO this low level control is realized at 10 kHz by the Elmo board sending the current to the motors.

2.3.4.b Torque control

On the other hand to have a robot which is compliant, one would like to control the torque applied τ_j and not the position.

There is a non linear relationship between the joint torque and the motor current which depends on the actuator model (see Section 2.3):

$$f : \mathbb{R} \mapsto \mathbb{R} \tag{2.37}$$

$$i_j \rightarrow \tau_j \tag{2.38}$$

Once the chain actuation is modeled and its parameters identified, one can express this relationship in a low-level controller to directly send current commands from desired torques to the robot. Chapter 3 presents one implementation of such a controller for the elbow of the TALOS robot.

2.4 Motion and Locomotion Planning

During this thesis, two distinct planning approaches have been used to compute the reference trajectories used by our controllers: a Walking Pattern Generator WPG Stasse et al. [59], [60] and a multi-contact locomotion framework Carpentier et al. [61], [62]. Because motion planning is not the core of our research subject, only this two approaches are described in this section. They have been chosen because they are part of the algorithms developed by the Gepetto team and can thus easily be connected to our architectures. But, we could have used any other methods providing reference trajectories for the CoM and the feet. Moreover, the two used approaches meet the requirements and needs of the thesis context: allowing to plan real-time locomotion solution for the first approach and multi-contact scenarios for the second one. Finally they are state-of-the-art solutions. We quickly present their position in the literature in this section.

2.4.1 Walking Pattern Generator

The generation of a stable humanoid robot gait, on flat floor without obstacles, is a commonly tackled issue in robotics. Introduced by Kajita et al. [41], the LIPM is widely used to approximate the non linear dynamics of a humanoid robot CoM during gait as a function of the CoP. Indeed, the CoP is a relevant variable to control as the CoP has to stay within the support polygon of the robot at all time to ensure the humanoid robot balance Wieber [63].

In Kajita et al. [64], the authors present a CoP preview control which computes the CoM of the humanoid robot over a prediction horizon from imposed footsteps

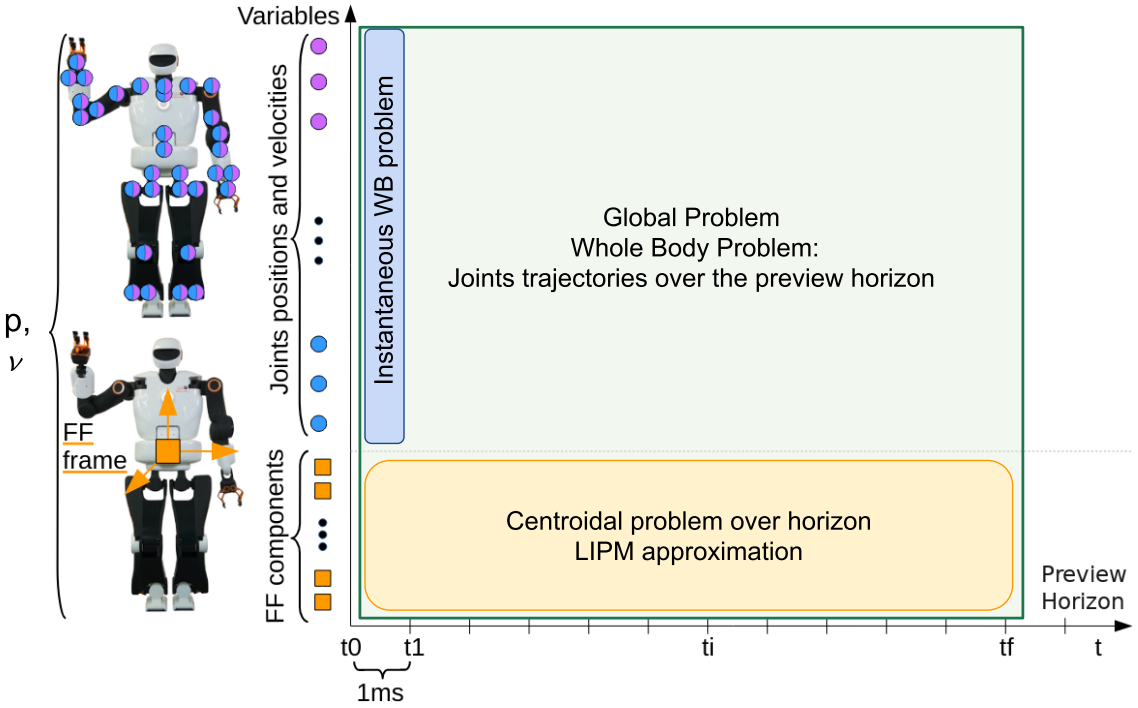

Figure 2.5: Scheme of the locomotion problem for humanoid robots, based on Naveau et al. [27], Giraud-Esclasse [71]. The global problem in the green rectangle (whole-body trajectory for the full preview future) is currently impossible to solve in milliseconds. However, it is possible to solve the blue box: the whole-body problem for the nearest future, as well as the linearized problem at the CoM level (orange box), over the full preview.

given to the controller. In Wieber [65], a new formulation of this CoP preview control using a Model Predictive Control (MPC) scheme is proposed. Nowadays, this MPC approach is the most commonly adopted method to design a WPG for humanoid robots Faraji et al. [66], Griffin and Leonessa [67], Scianca et al. [68]. Moreover, the original MPC has been improved to achieve automatic footstep placements by Herdt et al. [42]. Even non-linear reformulations of this MPC (thus a NMPC) were proposed in Naveau et al. [69], Caron and Kheddar [70].

Using these formulations, it is possible to solve the centroidal problem over a preview horizon. Then, these given references are used in a whole-body controller to compute position, velocity or torque commands for the complete joint configuration for the instantaneous future. This approach is depicted in Fig.2.5, the FF frame corresponds to the free-flyer frame, or the base frame, thus the under-actuated part of the robot where the centroidal dynamics acts.

The planning method used in this thesis to compute the reference trajectories can be detailed as follows and is presented in Fig.2.6:

First, the WPG uses a Non-linear Model Predictive Control (NMPC) Stasse et al. [59], [60] to compute a reference trajectory for the CoM, the ZMP, the waist orientation and the feet of the robot. The implementation uses the centroidal dynamics proposed in Kajita et al. [32]. The NMPC is sub-sampled internally at 200Hz, which represents a new trajectory planned over the entire horizon every 5ms.

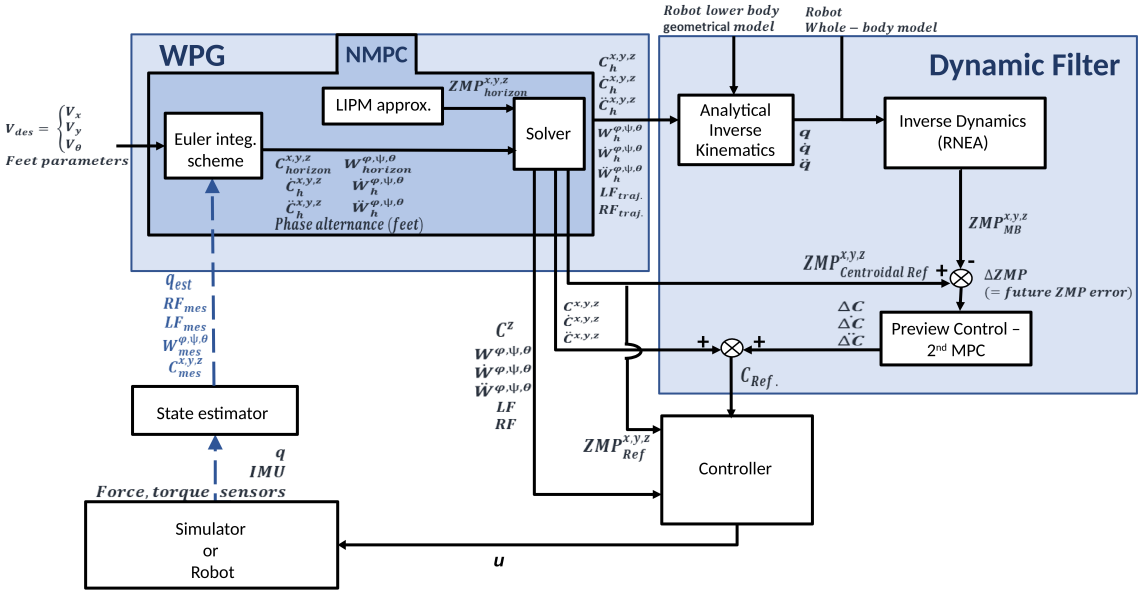

Figure 2.6: Scheme illustrating the Walking Pattern Generator and the Dynamic Filter Scherrer [1].

Then, a dynamic filter Kajita et al. [32] is used in order to take into account the influence of the whole body on the CoM trajectory and to correct it. Indeed, the LIPM neglects the torque induced by the limbs ($\dot{L}$) over the movement of the CoM and in the case of TALOS (which weighs around 100 kg) it leads to too important errors to maintain a balanced walking. The filter is computed with the Recursive Newton-Euler Algorithm Featherstone [15] implemented in the Pinocchio library Carpentier et al. [13]. It modifies the CoM trajectory to take into account the momentum generated by the limbs motion.

Finally, this algorithm provides desired trajectories for the ZMP (z^* or ZMP_{ref} in the scheme), the CoM (c^* or C_{ref} in the scheme), the waist orientation (W in the scheme) and the feet (p_i^* or LF, RF in the scheme), for a given set of foot steps or a desired velocity. The desired DCM ξ^* is deduced from the desired CoM and desired ZMP trajectories. A complete description of the WPG algorithm can be found in Scherrer [1], Naveau et al. [27].

The trajectories used in this thesis for straight walk simulations (Sections 4.4 and 5.4) have been computed using this method, with pre-defined set of foot steps. The complete framework of the WPG associated with control schemes is presented in Appendix 1, in the Fig.6.4. A modified version of the WPG is used for the planning of human like trajectories to achieve human-robot collaboration in Section 4.5. The new WPG is based on the same method presented here and in Naveau et al. [27], the main differences are detailed in 4.5.3.b.

2.4.2 Multicontact-locomotion-planning

The multicontact-locomotion-planning framework decomposes the global locomotion problem in several sub-problems solved sequentially, as shown in Fig.2.7. This approach aims at driving the motion planning problem by injecting knowledge on

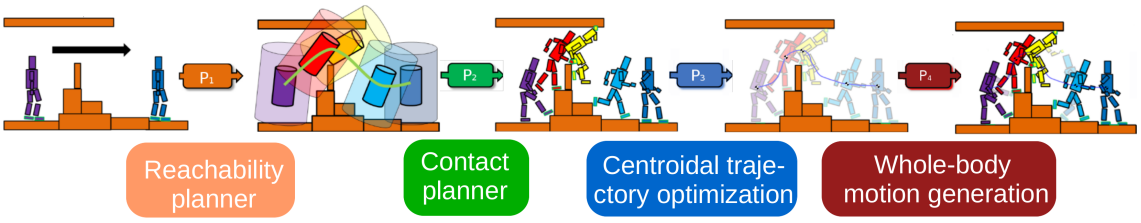

Figure 2.7: Planning a guide (a) Finding contacts poses (b) MPC on Centroidal dynamics (c) Whole-body control (d) [62]

the system.

- ▷ The first stage aims at finding a guide trajectory from the starting position to the goal one; making sure that the robot is in contact with the environment and that the trunk is not in collision with the environment.

- ▷ The second stage consists in finding contact stances for which there are quasi-static stable poses.

- ▷ The third stage consists in finding a centroidal trajectory which is dynamically balanced for the chosen set of contacts.

- ▷ The fourth stage consists in generating a whole body consistent trajectory.

Each transition from one stage to the next one assumes that the latter is able to realize the input. This decomposition breaks down the problem into lower dimensional problems. Each of them is solved in an efficient manner with real-time time answer. The framework gives the possibility to chose which algorithms is used to solve each sub-problem. A more precise description of this approach is provided in Carpentier et al. [61], [62].

In our case, we use the following solution: Given the initial and final poses of the robot, the reachability plan and a contacts sequence (stage one and two) are computed using the method SL1M (Sparse L1-norm Minimization) presented in Tonneau et al. [72]. It is a convex relaxation of the mixed integer (MI) programming approach for planning contact sequences for legged robots. Then, the centroidal dynamics (see Section 2.2) is optimized using two convex relaxations based on trust regions Ponton et al. [73]. Similarly to the WPG dynamic filter method, it takes into account the momentum generated by the swing leg thanks to iterations between a kinematic whole-body formulation and the centroidal dynamic optimization. In contrast, when solving Eq. 2.16, it does not assume that $\dot{L} = 0$. Finally, as in the WPG approach, the steps placements and CoM trajectory are given to a whole body controller in charge of generating the joints trajectories (or torques depending on the robot low-level control).

The trajectories used in the platforms and stairs simulations have been computed using this open-source framework *multicontact-locomotion-planning* [62].

2.5 Real-time Whole Body Control

2.5.1 Introduction

The goal of this section is to report the existing solutions on real-time whole body controller. From a mathematical viewpoint, a real-time whole body controller aims to find a control vector $u \in \mathbb{R}^{n_j}$ for each motor of a robot ($n_j = 32$ for TALOS, and $n_j = 10$ for TIAGo), for a given set of tasks, at high frequency (for instance 1kHz). The controller is said to be instantaneous and corresponds to the blue block of Fig.2.5.

In all the thesis, we assume that u can take only three values:

- ▷ $u = v$, i.e. the control problem is solved considering the free variables as being velocities.
- ▷ $u = a$, i.e. the control problem is solved considering the free variables as being accelerations.
- ▷ $u = \tau$, i.e. the control problem is solved considering the free variables as being joint torques.

For the first and second case, the robot is position controlled and the low-level servo control transform the commanded joint positions into motor currents with a PID at high frequency (typically 2 kHz for TALOS, see Section 2.3.4). This means that the output of the optimization is integrated one or two times before being sent to the low level system of the robot. In general, for locomotion scenarios, those two controls give the most efficient walk, achieving the fastest results with less tracking errors along the trajectories Caron et al. [50].

Torque control, in another hand, requires a low level controller able to transform the desired joint torques into motor currents (see Sections 2.3.4,2.3). To perform this transformation, the low level controller needs the model of the robot's chain actuation. This model can be given by the robot's manufacturer, or the manufacturer can directly provide the low-level controller. However, this model often presents inaccuracies and the identification of the robot inertial parameters and the motor's chain drive is required to correct it. This issue is partially why torque control was less popular until recently, the transition from the simulations (which often do not model the actuators dynamics) to the real robot proving to be much harder than position control Mesesan et al. [49], Englsberger et al. [74].

2.5.2 Whole-body controller

2.5.2.a Introduction: Task Function

The most widely-used technique to generate whole-body motions is to design the motion in a space dedicated to the task, rather than directly at the whole-body level (i.e. in configuration space, see Section 2.1.3). Indeed, it is easier to design the reference motion in the task space, and then transcript this reference to the configuration space.

Once the planning has computed reference trajectories for the CoM and feet

positions, they are given to the controller as desired *features* noted s^*. The task functions Khatib [36], Samson et al. [37], Escande et al. [38] are defined as errors e between the current features s and the desired ones and are often non-linear. For instance, the geometric relationship between the configuration space and the Euclidean space of the end effector (hands or feet) is non-linear. In particular, there can be multiple configurations possible to reach a desired end effector position.

Therefore, the task functions are usually computed as an optimization problem:

$$\min_q ||s - s^*|| \tag{2.39}$$

with the relation between s and q defined by the differentiable function f such that:

$$\begin{array}{rccc} f: & \mathbf{C} & \rightarrow & \mathbf{S} \\ & q & \mapsto & s \end{array} \tag{2.40}$$

with $\mathbf{C}$ the configuration space and $\mathbf{S}$ the task space (for example the Special Euclidian group $SE(3)$).

This problem is solved iteratively using a gradient descent approach Chevallereau and Khalil. [75]. Once expressed as an optimization problem, it is interesting to see that this opens up an incremental approach where constraints can be added, and various formulations of equivalent problems can be considered.

To generate a complex motion on a humanoid robot, several tasks have to be combined, sequentially Mansard et al. [76] or simultaneously. The simultaneous execution of two tasks on a robot can be achieved in two ways: by setting respective weights between the tasks, or by imposing a strict hierarchy. Recently, a solution has been proposed to handle both strict and non-strict priorities of an arbitrary number of tasks Liu et al. [77]. These first two solutions have been implemented in the Gepetto Stack-of-Task (SoT) software (see Section 1.6):

- ▷ The library implementing the task formulation in velocity domain using a strict hierarchy as in Mansard et al. [16], Mansard and Chaumette [78] is **SoT-Core** [20],
- ▷ The library implementing the task formulation in torque using weights and inequalities is **SoT-Torque-Control** [79] (C++ wrapper of TSID [80] using the dynamic-graph [14]).

2.5.2.b QP Formulation: Task space velocity or acceleration control

Introduction

Generally these optimization problems involving multiple tasks and constraints are solved using Quadratic Programming. A QP is a convex optimization problem with a convex objective function (cost function) which is quadratic and constraint functions which are affine Boyd and Vandenberghe [12]. A famous unconstrained QP is the Least Square Problem (LSP), for instance the Eq.2.39 is a LSP, which has an analytical solution (given by using pseud-inverse).

However, a QP with equality and inequality constraints is not easily solved. One method to solve these optimization problem is to use active set solvers Nocedal and Wright [81]. These solvers use the Lagrangian multipliers formulation and the

Karush-Kuhn-Tucker (KKT) conditions Boyd and Vandenberghe [12], Nocedal and Wright [81]. They project the free variables of the problem in the null space of the equality and inequality constraints. Then, the objective function is minimized in this subspace. Such solvers guarantee that the constraints are properly fulfilled, but might be computationally expensive, in particular when the matrix of the linear constraints is ill-conditioned.

Another way to cope with the constraints is to reformulate a new optimization problem with all the constraints as tasks inside the objective function. Therefore, the constraints might not be fulfilled. This new problem is mathematically equivalent to the one with constraints but it means that the related tasks might not be completely performed Hofmann et al. [82]. This can be helpful for tasks that are subject to the environment disturbances.

Therefore, an important question is to decide which constraints need to strictly be respected (equality constraints), and which constraints can be relaxed (cost function). In general the constraints related to the physical laws are strictly enforced because any violation would induce damages to the robot. Constraints related to tasks such as the position of the right hand might be performed at best. It is important to notice that some tasks, such as foot contact with the ground, while being subject to modifications from the environment, may have a strong impact on the robot if they are not fulfilled correctly.

Mathematical formulation for velocity control

Let us simplify the motion equation based on the rigid body dynamics (see Eq.2.4) when there is no contacts:

$$Ma + h = N^T \tau \tag{2.41}$$

with $h = C(q, v)v + g(q)$.

If we have a task e regulating a feature s, the dynamics of the task can be imposed, for instance as an exponential decay (last equation):

$$\begin{aligned} e &= s \ominus s^* \\ \dot{e} &= \dot{s} - \dot{s}^* \\ \dot{e} &= -\lambda e = -\lambda(s - s^*) \end{aligned} \tag{2.42}$$

with $\ominus$ the difference operator in the task space, for instance the operator of the Lie group. λ the control gain of the task, which can be constant, or adaptive in order to control the speed of convergence to the desired feature. The task is realized when $\dot{e}$ reaches 0 or is close to 0.

Using the previous function f such that $s = f(q)$ one can define the relationship between $\dot{s}$ and v using the Jacobian of s Lynch and Park [35] and the formulation given in Eq.2.6 for the under-actuated part:

$$\dot{s} = [J_u \; J_a]v = Jv \tag{2.43}$$

One can notice that the Jacobian for the task error e is the same one as its feature s, because s^* depends only on the time and not on the configuration. It is thus possible to compute v with Eq.2.43 as:

$$v = J^\dagger \dot{s} \tag{2.44}$$

with $J^\dagger$ the Moore-Penrose pseudo-inverse of J, since J can be non square and thus not invertible.

It is possible to reformulate Eq.2.42 to obtain the dynamics of the feature and then the desired joint velocities:

$$\begin{array}{rrcl} & \dot{s} & = & \dot{s}^* + \dot{e} \\ \Leftrightarrow & Jv & = & \dot{s}^* - \lambda(s \ominus s^*) \\ \Leftrightarrow & v & = & J^\dagger(\dot{s}^* - \lambda(s \ominus s^*)) \end{array} \tag{2.45}$$

One can introduce a slack variable w (an implicit optimization variable) to express the realization of the task, then the solver will optimize this variable instead of the error directly, allowing a relaxation of the problem if no solution is found (from Eq.2.42):

$$\begin{array}{rcl} w & = & \dot{e} + \lambda e \\ w & = & Jv - (\dot{s}^* - \lambda(s - s^*)) \end{array} \tag{2.46}$$

A simple formulation of the QP problem can then be expressed as:

$$\begin{array}{ll} \min\limits_{v,\tau} & \|w\|^2 \\ \text{s.t.} & Ma + h = N^T\tau \end{array} \tag{2.47}$$

Where the free variables are the robot velocity v and the torque τ.

Mathematical formulation for acceleration/torque control

In this thesis, we also used the formulation of the task errors at the second order to achieve acceleration control. In Eq.2.42 one can define the dynamics of the second derivative of the error $\ddot{e}$ (third equation) Lynch and Park [35]:

$$\begin{array}{rcl} \ddot{e} & = & \ddot{s} - \ddot{s}^* \\ \ddot{s} & = & Ja + \dot{J}v \\ \ddot{e} & = & -2\lambda\,\dot{e} - \lambda^2\,e \end{array} \tag{2.48}$$

And thus, similarly as before one can obtain the desired joint acceleration and the slack variable w as:

$$\begin{array}{rcl} a & = & J^\dagger(\ddot{s}^* - \dot{J}v - 2\lambda\,\dot{e} - \lambda^2\,e) \\ w & = & \ddot{e} + 2\lambda\,\dot{e} + \lambda^2\,e \\ w & = & Ja + \dot{J}v - (\ddot{s}^* - 2\lambda\,\dot{e} + \lambda^2\,e) \end{array} \tag{2.49}$$

Leading to the following QP:

$$\begin{array}{ll} \min\limits_{a,\tau} & \|w\|^2 \\ \text{s.t.} & Ma + h = N^T\tau \end{array} \tag{2.50}$$

Where the free variables are the accelerations a and the torque τ. It is possible to work only on one type of variables (to achieve acceleration or torque control on the robot) by substitution.

Formulation with contacts

If the system is in contact with the environment, its dynamics must account for contact forces f_{ext}. If contacts are soft, measured/estimated contact forces $\hat{f}_{ext}$ can be easily included:

$$\begin{aligned} \min_{a,\tau} \quad & \|w\|^2 \\ \text{s.t.} \quad & Ma + h = N^T\tau + J_c^T \hat{f}_{ext} \end{aligned} \tag{2.51}$$

with J_c the Jacobian of the contact points x_c according to the robot state vector, as defined in Eq.2.6.

But if contacts are rigid, they constrain the motion. They can be implemented as nonlinear functions, which are differentiated twice:

$$\begin{aligned} x_c &= 0 \quad \Leftrightarrow \quad \text{Contact point does not move} \\ J_c^T v &= 0 \quad \Leftrightarrow \quad \text{Contact point velocity is zero} \\ J_c^T a + \dot{J}_c v &= 0 \quad \Leftrightarrow \quad \text{Contact point acceleration is zero} \end{aligned} \tag{2.52}$$

This representation leads to the following optimization problem:

$$\begin{aligned} \min_{a,f,\tau} \quad & \|w\|^2 \\ \text{s.t.} \quad & \begin{bmatrix} J_c & 0 & 0 \\ M & -J_c^T & -S^T \end{bmatrix} \begin{bmatrix} a \\ f \\ \tau \end{bmatrix} = \begin{bmatrix} -\dot{J}_c v \\ -h \end{bmatrix} \end{aligned} \tag{2.53}$$

These optimization problems represent Equality-Constrained Least-Square Problem (ECLSP) because they only have equality constraints. Unconstrained LSP can be directly solved using the Moore-Penrose pseudo-inverse Boyd and Vandenberghe [12] (if J is surjective). For instance, for a fully actuated robot ($q = q_j, v = \dot{q}_j, a = \ddot{q}_j$), if the desired feature does not move ($\ddot{s}^* = 0$) and using Eq.2.49 one can find the solution:

$$\ddot{q}_j \quad = \quad -J^\dagger \left(2\lambda \dot{e} + \lambda^2 e + \dot{J}\dot{q}_j\right) \tag{2.54}$$

The ECLSP can also be resolved with pseudo-inverse by using null-space projector Boyd and Vandenberghe [12]. But the main benefit of QP solver is that they can handle inequality-constraints. It is mainly used to define boundaries of the system such as torque, velocity or joint limits; and also friction cones for the contacts:

$$\begin{aligned} \tau_{min} &\leq \tau \leq \tau_{max} \\ q_j^{min} &\leq q_j \leq q_j^{max} \\ \dot{q}_j^{min} &\leq \dot{q}_j \leq \dot{q}_j^{max} \\ f_{min} &\leq f \leq f_{max} \end{aligned} \tag{2.55}$$

2.5.2.c HQP and Weighted Sum

Complex robots are typically redundant with respect to the main task they must perform, this redundancy can be used to execute secondary tasks. This multi-objective optimization can be achieved by setting respective weights between the tasks, or by imposing a strict hierarchy between them (cascade of QP or HQP) Kanoun et al. [83].

We assume that the robot must perform N tasks, each defined by a task function and its optimization variable w_i:

$$g_i = \|w_i\|^2 \tag{2.56}$$

In the following we chose to use the formulation of Eq.2.49 for w_i.

Weighted Sum

The easiest strategy is to sum all functions using user-defined weights μ_i:

$$\begin{aligned}
\min_{a,f,\tau} \quad & \sum_{i=0}^{N} \mu_i \, g_i \\
\text{s.t.} \quad & \begin{bmatrix} J_c & 0 & 0 \\ M & -J_c^T & -S^T \end{bmatrix} \begin{bmatrix} a \\ f \\ \tau \end{bmatrix} = \begin{bmatrix} -\dot{J}_c \, v \\ -h \end{bmatrix} \\
& \tau_{min} \leq \tau \leq \tau_{max} \\
& q_j^{min} \leq q_j \leq q_j^{max} \\
& \dot{q}_j^{min} \leq \dot{q}_j \leq \dot{q}_j^{max} \\
& f_{min} \leq f \leq f_{max}
\end{aligned} \tag{2.57}$$

This problem remains standard computationally-efficient (compared to LSP). But, finding proper weights is hard and too extreme weights can lead to numerical issues. It is easier to find the priority of a task than its weight, but a cascade of QP can quickly become too computationally expensive if computed in a naive way. For this reason several teams working on humanoid robots are using this formulation for their controllers:

- ▷ In Koolen et al. [54], the IHMC whole body controller is described. The tracking of the feet position and the foot contact wrenches tasks are considered as strict equalities, whereas the momentum rate is tracked at best inside the cost function.
- ▷ In Bouyarmane et al. [84], the JRL/IDH team is using strict inequalities to maintain the physical constraints: joint limits, velocity limit, acceleration limit, and strict equality to maintain the dynamical consistency. All the other tasks are tracked at best in the cost function.

The weighted approach tends to imply new gains when considering new applications. Finding these new gains is not straightforward and often require time-consuming tuning and expertise. However some procedures exist to automatically tuned these gains, such as in Pucci et al. [85] for a momentum based controller or in Teshnehlab and Watanabe [86] by using learning approach. Penco et al. [87]

proposes a multi-objective optimization approach to learn the gains and also the priorities (strict and soft) of a controller.

Lexicographical Hierarchy

The alternative strategy is to use strict priorities to order the task functions, which means to solve a cascade of QP problems:

$$
\begin{aligned}
& g_i^* = \min_{a,f,\tau} \quad g_i \\
\text{s.t.} \quad & \begin{bmatrix} J_c & 0 & 0 \\ M & -J_c^T & -S^T \end{bmatrix} \begin{bmatrix} a \\ f \\ \tau \end{bmatrix} = \begin{bmatrix} -\dot{J}_c\, v \\ -h \end{bmatrix} \\
& g_j = g_j^* \quad \forall j < i \\
& \tau_{min} \leq \tau \leq \tau_{max} \\
& q_j^{min} \leq q_j \leq q_j^{max} \\
& \dot{q}_j^{min} \leq \dot{q}_j \leq \dot{q}_j^{max} \\
& f_{min} \leq f \leq f_{max}
\end{aligned}
\tag{2.58}
$$

The following works have investigated the use of strict inequalities on humanoid robots:

- ▷ In Hoffman et al. [88], the IIT team of WALKMAN presents a scheme to perform Cartesian Impedance Control in a hierarchical manner. The new task at level i is performed at best, while preserving the integrity of previous tasks. The dynamics is preserved through its formulation as a task equality.
- ▷ In Henze et al. [89], the DLR team describes a whole body controller to perform contact control. In this work, the authors use 4 levels of hierarchy. The first level is used to maintain contact with the environment and to make sure that the planned external wrench is applied to the robot. The second level contains the tracking tasks of the end-effectors not used for the balancing. The third level is used to control the CoM and the fourth level is used to control the posture. This controller is moreover passive (see Section 2.6.3)

To illustrate this last solution with inequalities, let us take two tasks with priority order $e_1 \prec e_2$ and bounds $\overline{e_1}, \underline{e_1}$ and $\overline{e_2}, \underline{e_2}$ respectively. When there is no contacts, the expression of the Lexicographical QP applied on inverse kinematics (see the previous velocity formulation of Section 2.5.2.b: $\dot{e} - Jv = w$) can be written as Saab et al. [90]:

$$
\begin{aligned}
H_1 = \min_{v,w_1} \quad & \|w_1\|^2 \\
\text{s.t.} \quad & \underline{\dot{e_1}}^* \leq J_1\, v + w_1 \leq \overline{\dot{e_1}}^* \\
H_2 = \min_{v,w_2} \quad & \|w_2\|^2 \\
\text{s.t.} \quad & \underline{\dot{e_1}}^* \leq J_1\, v + w_1^* \leq \overline{\dot{e_1}}^* \\
& \underline{\dot{e_2}}^* \leq J_2\, v + w_2 \leq \overline{\dot{e_2}}^* \\
\text{with} \quad & w_1^* = \arg\min_{w_1} H_1
\end{aligned}
\tag{2.59}
$$

A naive implementation would be to solve the second problem H_2 in the same space than H_1, i.e. $v \in \mathbb{R}^n$. However by working in the base of the null space

of H_1, the computation becomes extremely efficient Nocedal and Wright [81]. In Section 4.4 is presented a hierarchical controller which exploit this specific structure to keep its control frequency higher than 1kHz in average with 4 hierarchy levels (see Section 4.4.4). Note however that when the tasks are compatible the projection in null space is useless. In addition, when a constraint is activated it means that the robot is working at one of its limit which is rarely desirable. The main interest of strict inequalities is the guarantee that a constraint is never violated and that a task of lower priority will not interfere.

Once the QP solver find a solution (using for example the active sets method Nocedal and Wright [81]), the desired joint torques/accelerations or velocities are sent to the robot. However, most robots such as TALOS or HRP-2 ultimately controls their motor in current (electric motors). Thus the low-level controller must transform these commands into motor in current, this can be done by two general approaches: position control or torque control (see Section 2.3.4).

	TALOS	WALK-MAN	TORO	HRP-4	JAXON	Atlas	Gazelle
Date	2018	2018	2014	2010	2015	2016	2020
Actuation	Electric	Electric/ Compliant	Electric/ Compliant	Electric	Electric	Hydraulic	Electric
DoFs	32	32	39	34	32	28	13
Size	1.75m	1.85m	1.74m	1.51m	1.88m	1.50m	1.30m
Weight	100kg	102kg	76.4kg	39kg	127kg	75kg	60kg
Torque sensors	22	22	25	0	32	28	13
F/T sensors	Ankle/ Wrist	Ankle/ Wrist	Ankle	Ankle/ Wrist	Ankle/ Wrist	Ankle/ Wrist	Ankle
IMU/ Accelero.	1	2	2	1	1	1	1
Cameras	2	1	3	4	1	2	0
LIDAR /Laser	1	1	0	0	1	1	0

Table 2.1: Comparison of Humanoid robots specifications.

2.5.3 Recent Humanoid Robots Whole Body Controllers

In this section, a short presentation of recent humanoid robots and their control schemes is given. A review on the hardware design of humanoid robots can be found in Ficht and Behnke [91]. The aim is to highlight the particularities of the robot TALOS and investigate the promising control schemes compatible for our robot. PAL Robotics demonstrated, during IROS 2018, an accurate whole-body balancing control on TALOS PAL-Robotics [92], however it is only position controlled.

In the Table.2.1 are presented some specifications of 6 humanoid robots and a biped. The robot Gazelle is presented in this section because it can be entirely torque controlled and achieves great results while walking on uneven terrain because of its hardware composition and control Jeong et al. [93]. One has to notice that the TORO robot has more DoFs than the other robots because its 39 DoFs include its hands (6 DoFs each). The only robot presented here which does not have electric actuation is the Atlas robot, with hydraulic actuation using valve current (however other hydraulic robots exist such as HYDROïD from LISV and Hydra from the AIST). The HRP-4 robot has no torque sensors to achieve torque control but may be controlled in current.

The robot WALK-MAN (see Fig.2.8) was first presented by the IIT in 2015 and has then been updated to reach the presented final version. The humanoid robot has been design to operate in damaged buildings and hostile environment Tsagarakis et al. [94]. It can be torque controlled. Recently a two strict hierarchy whole-body controller has been implemented, using energy shaping Subburaman et al. [95] to control a falling-over scenario. It is an interesting strategy because it allows to control the energy of the system, as energy shaping is an approach close to the one of the passivity Folkertsma and Stramigioli [96].

TORO (see Fig.2.8) was made specifically for torque control Englsberger et al. [97] by the DLR, it was first presented in 2013 but the project begun in 2010.

Figure 2.8: Left: WALK-MAN. Center: TORO. Right: HRP4.

Figure 2.9: Left: JAXON. Center: Atlas. Right: Gazelle.

This humanoid robot is robust to walking, balancing and multi-contact scenarios, Henze et al. [89] presents a passivity-based whole body controller to perform multi-contact balancing. This work has led this thesis to investigate the passivity theory to ensure the stability of the whole-body controllers. Englsberger et al. [74] presents a weighted whole-body controllers with DCM control on TORO, and Englsberger et al. [98] describes an implementation proving the Lyapunov and passivity stability even for conflicting sub-tasks.

The HRP-4 robot (see Fig.2.8) is the follow up of the HRP-2 robot designed by the Kawada Industries and AIST. It is a light-weighted robot which is designed to collaborate with humans, and able to react to external forces. The recent papers are from the common laboratory of the CNRS-AIST JRL realized at the LIRMM. In Caron et al. [50] the HRP-4 robot succeed stairs climbing in Airbus Plant, using a whole-body admittance controller with strict hierarchy. The CoM admittance control used in this paper is similar to what is described in Section 2.2.3.b. The stabilization of the control is based on the LIPM and feet wrench distributions. The robot is controlled in position, while the QP is solved for accelerations free variables, thus the command is integrated two times. In Bouyarmane et al. [84] is presented a weighted whole body controller achieving task force control with multi-robot. It also uses admittance control and the robot is controlled in position.

TORO and HRP-4 have been used in Kheddar et al. [99] to achieve some manufacturing tasks in an assembly line. The results of a complete solution embedding visual tracking, localization, planning and control is impressive. One can notice that HRP-4 is used for tasks which do not require force application and have limited contact. When multi-contact is needed or to realize the manufacturing applications, the torque controlled robots TORO and HRP-2Kai are used. Compared to what is achieved, we want to answer the first limiting factor encounter in this study, which is the security of the robotic platform. The repeated falls and weakness of the humanoid robots lead to damages during the experiments. Thus, we want to ensure the stability of our solution while conserving the robot capabilities to achieve demanding tasks (such as applying huge torque for drilling).

The robot JAXON (see Fig.2.9) from the JSK laboratory is also a humanoid robot designed for torque control, to perform disaster relief assistance. In Shirai et al. [100] is presented a whole body torque controller which can absorb impulsive disturbances. Associated with an online walking controller, the framework demonstrates high robustness to strong collision with the robot's legs. This robot is however the heaviest.

The robot Atlas (see Fig.2.9) from Boston Dynamics and the MIT is a humanoid robot using state-of-the-art hardware which demonstrates human-level agility. However, the hardware specifications of this robot are not disclosed. In Koolen et al. [54] is detailed a momentum based whole body torque control, using a soft hierarchy. This formulation controls the AM of the robot and achieve impressive results with the robot while walking on different terrains (tilted cylinder blocks, rubble) and fast on flat one (0.43m/s). These results confirmed our interest to add a control on the AM of the robot for our control schemes.

Finally, the robot GAZELLE (see Fig.2.9) was recently designed by the KAIST, aiming to build a fast and reliable biped platform for walking experiments. As said before, the paper Jeong et al. [93] presents the results obtained with this new

platform. The controller split the control between a CoM task and ankles force tasks. It controls the damping of the CoM and then solves the Inverse Kinematics to obtain the desired joint configuration; while using the F/T sensors to regulate the torques on the ankles and send directly a torque command for them. This architecture allows the biped robot to walk fast on uneven terrain and to be robust to external pushes.

Compared to the robots in the literature, TALOS has average capabilities, however, it is the only robot commercially available (and not too expensive). It is possible to implement torque or position control, and it has been built to be able to sustain heavy load for a long duration and to apply high torques/forces to complete manufacturing operations.

2.6 Stability Analysis: Convergence toward an equilibrium

2.6.1 Introduction

Humanoid robots are meant to act in the same environment than humans, interacting with it, but this environment can be unknown and subject to disturbances. Even performing simple tasks in fixed environment can lead to instabilities, the authors encountered this problem while testing a classical torque control scheme using inverse dynamics on the robot TALOS on a simple postural task. After some repetitions of a sinusoidal motion on the robot arm, the system diverged brutally and blocked some of its harmonic drives. The gains tuned in simulation (simulating the rigid chain actuators) were too high for the real robot. Thus, even with gains tuned on a proper simulator modeling the actuators, we cannot expect the solution to remain stable (i.e. remains in a bounded distance from an equilibrium). To provide a safe and reliable interaction with the environment and possibly humans, it is necessary to ensure the stability of the robotic system. This problem is commonly tackled in the community using two strategies: the Lyapunov analysis or the passivity-based analysis. In this section, we first present briefly the Lyapunov theory and recent works proving the Lyapunov stability of humanoid robots whole-body controllers. Then, the passivity theory is described with more details as this analysis is used in the Chapter 5 to prove the stability of our new controller.

2.6.2 Lyapunov Theory

The Lyapunov theory consists in analysing the stability of an equilibrium, looking if the solution remains in a bounded distance from the equilibrium or converges asymptotically (Asymptotic stability) or exponentially (Exponential stability) Bacciotti and Rosier [101]. This analysis ensures that the controller will not find a diverging solution. We have the following definitions for local and global strict asymptotic stability Bacciotti and Rosier [101]:

Definition 1 (Lyapunov local asymptotic stability)**.** *For a system with the state space model of* $\dot{x} = f(x)$ *(*$x \in \mathbb{R}^n$*) and an equilibrium state* $x^* = 0$*. If there exists*

a Lyapunov function $V : \mathbb{R}^n \to \mathbb{R}$ *such that:* V *is continuous, has continuous first derivatives, is locally positive definite and its time derivative is locally negative definite, i.e:*

$$\begin{cases} V(0) = 0 \\ V(x) \in \mathcal{C}^1 \, on \, \mathbf{B} \\ V(x) > 0 \, \forall x \in \mathbf{B} \backslash \{0\} \\ \dot{V}(x) = \nabla V(x).f(x) \\ \dot{V}(x) < 0 \, \forall x \in \mathbf{B} \backslash \{0\} \end{cases} \tag{2.60}$$

where $\mathbf{B}$ *is the local neighborhood of the origin. Then the equilibrium is proven to be locally asymptotically stable (local Lyapunov stability).*

Definition 2 (Lyapunov global asymptotic stability)**.** *For a system with the state space model of* $\dot{x} = f(x)$ $(x \in \mathbb{R}^n)$ *and an equilibrium state* $x^* = 0$*. If there exists a Lyapunov function* V *respecting the local Lyapunov stability (Definition 1), with* $\mathbf{B} = \mathbb{R}^n$ *and radially unbounded then the equilibrium is proven to be globally asymptotically stable (global Lyapunov stability).*

We recall, V *radially unbounded if:* $\|x\| \to \infty \implies V(x) \to \infty$

From these definitions, one can obtain the exponential stability by majorating $\dot{V}$ by an exponentially converging function.

In the control literature, Lyapunov functions are often built based on the relative kinetic and potential energy of the system to prove its stability. Recently, Nava et al. [102] proposed a momentum-based QP with two strict hierarchies and proved its Lyapunov stability by using integral gains in the task functions. In a similar way, Cisneros et al. [103] adds an integral term to the computed torque while using it in the constraints of their QP to prove the Lyapunov stability. Finally, Englsberger et al. [98] has designed a new controller creating separate Lyapunov functions for each task and proving the overall stability and passivity of the system.

2.6.3 Passivity Theory

The passivity-based analysis consists in investigating the energy flows within a system Schaft [104]. It is also possible to directly control the system energy without proving the passivity, which requires an analysis of the energy derivative of the system. For instance, Joseph et al. [105] proposes a formulation of a kinetic energy constraint in a QP to prevent a manipulator robot from transferring dangerous amounts of this energy during an impact. Yet, it has been proven in Camlibel et al. [106] that when a robot interacts with its environment, a passively-controlled robot is a necessary condition for stability Folkertsma and Stramigioli [96]. Using the derivative of the energy in the system, the passivity can be analyzed to ensure that the system converges to the desired equilibrium or to detect which components have an unbounded energy consumption.

2.6.3.a Definition

The principle of passivity comes from the concept of dissipative dynamical system Willems [107], in particular proving that the internal power of the system is less

than or equal to the power transferred to the system through its port. This concept is associated with the Port-Hamiltonian System theory; which is an energy-based formulation of the physical systems Schaft [104].

The Hamiltonian $\mathcal{H}$ is the sum of the potential and kinetic energy of the system. Using the Poisson framework representation (input-state-output representation), one can obtain the following formulation:

For a state vector $x \in \mathbb{R}^n$, $u, y \in \mathbb{R}^m$:

$$\begin{aligned} \dot{x} &= \Big[J(x) - R(x)\Big]\frac{\partial \mathcal{H}}{\partial x}(x) + g(x)u \\ y &= g^T(x)\frac{\partial \mathcal{H}}{\partial x}(x) \end{aligned} \tag{2.61}$$

with $J(x) = -J^T(x)$, $R(x) = R^T(x) \geq 0$. g represents the interconnection, and therefore effect, of the port variables on the state variables and vice versa. The matrix J is a power-continuous interconnection and is skew-symmetric, whereas R models pure resistive losses of the system. The latter property is obtain by taking the time derivative of the Hamiltonian:

$$\begin{aligned} \dot{\mathcal{H}} &= \frac{\partial \mathcal{H}^T}{\partial x}(x)\dot{x} \\ &= \frac{\partial \mathcal{H}^T}{\partial x}(x)\Big[J(x) - R(x)\Big]\frac{\partial \mathcal{H}}{\partial x}(x) + \frac{\partial \mathcal{H}^T}{\partial x}(x)g(x)u \\ &= -\frac{\partial \mathcal{H}^T}{\partial x}(x)R(x)\frac{\partial \mathcal{H}}{\partial x}(x) + y^T u \end{aligned} \tag{2.62}$$

using the fact that J is skew-symmetric ($\frac{\partial \mathcal{H}^T}{\partial x}(x)J(x)\frac{\partial \mathcal{H}}{\partial x}(x) = 0$). Thus, $\dot{\mathcal{H}}$ is the power supplied through the port $y^T u$, minus the power lost to friction, quadratic on $R(x)$ Schaft [104].

Using the dissipative definition, a system is defined as a passive system Schaft [104] if it respects the following definition:

Definition 3 (Passive System)**.** *A system with the state space model of $\dot{x} = f(x, y)$ with initial state $x(0) = x_0 \in \mathbb{R}^n$, input vector $u \in \mathbb{R}^m$ and output $y = h(x, u)$ is said to be passive, if there exists a positive semi-definite function $H : \mathbb{R}^n \to \mathbb{R}^+$, called storage function, such that:*

$$H(x(T)) - H(x(0)) \leq \int_0^T y^T(t)u(t)dt \tag{2.63}$$

$\forall y : [0, T] \to \mathbb{R}^m$, $x_0 \in \mathbb{R}^n$ and $T > 0$.

One can see that by choosing $H = \mathcal{H}$, the storage function is the energy in the system and the supply rate $y^T u$ is the power transferred to the system through its port. If $R = 0$, there is no dissipation and the system is conservative.

From the Definition. 3, the passivity of the system can equivalently be proven by finding an appropriate storage function $H(x)$ respecting the property 1:

Property 1 (Passivity Storage Function)**.** *A system with the state space model of $\dot{x} = f(x, y)$ with initial state $x(0) = x_0 \in \mathbb{R}^n$, input vector $u \in \mathbb{R}^m$ and output*

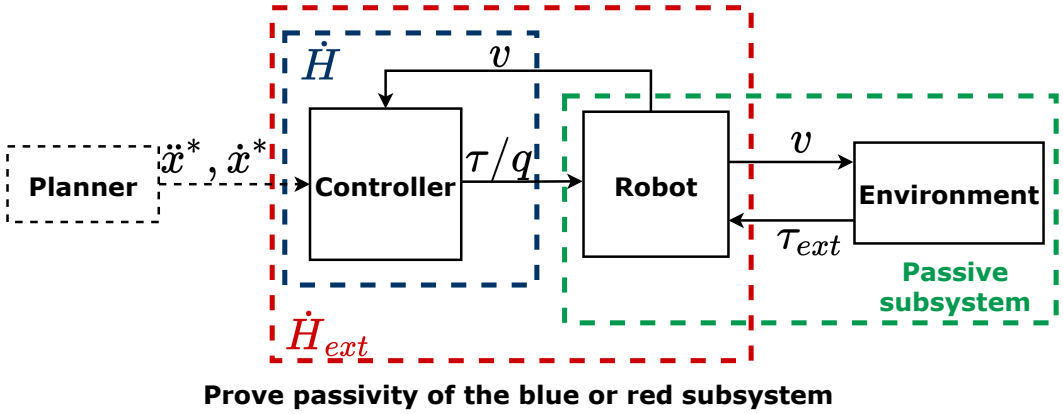

Figure 2.10: Simple Analysis of the passivity of a robotic system.

$y = h(x, u)$ is said to be passive, if there exists a positive semi-definite function $H : \mathbb{R}^n \to \mathbb{R}^+$, called storage function, such that:

$$\dot{H}(x) \leq y^T(t)u(t) \quad \forall (x, y) \in \mathbb{R}^n \tag{2.64}$$

This is why the storage functions are defined as potential and kinetic energy in the literature (for Lyapunov and Passivity analysis). Using the equation of the robot dynamics Eq.2.4, on can define a controller sending torque commands τ to the robot. From a Port-Hamiltonian System point of view, the controller is thus interconnected to the power port of the robot $\{v, \tau\}$: the robot receives an input τ and outputs a velocity v onto the environment (see the Fig.2.10 for a simple scheme example). The following section describes a simple analysis of the interconnected systems of a robotic system.

2.6.3.b Analysis of a robotic system

A robotic system can be decomposed in three components: the robot, the controller and the environment. It is often assumed that the environment of the robot can be described by a passive mapping ($v \to -\tau_{ext}$), in connection with impedance control Garofalo and Ott [108], Ott et al. [109] (green block in the block in the Fig.2.10, see Section 2.7.3). Then, because the group robot-environment is passive, to prove the overall passivity of the system, one has to prove that the group controller-robot is passive (blue block in the Fig.2.10, with τ or q as control variable). Indeed, the interconnection of passive systems leads to a general passive system Folkertsma and Stramigioli [96]. A second solution is to directly prove that the system {robot+controller} is passive with respect to the environment (red block in the Fig.2.10).

2.6.3.c Energy Tank

The energy tanks are artificial energy storing elements that keep track of the energy naturally dissipated by the system agents Giordano et al. [110]. The concept of

virtual energy tank is to give an energy budget to the controller and to change the behavior of the controller in function of it. The tank monitors the energy flow of the system to guarantee the passivity. The energy stored in this tank can be re-used to accomplish different goals without violating the passivity of the system. By augmenting one control architecture with a virtual energy tank Folkertsma and Stramigioli [96], it is possible to limit the energy generated by the system and then make it passive. For instance, in the case of a controller with multiple tasks, one can introduce an energy tank in the framework to store the energy dissipated by these tasks. The tank will then be used to transfer the stored energy when a task needs more energy to be completed. On the contrary, if the tank is empty, it will be used to deteriorate the tracking of this task to ensure the passivity of the system Folkertsma and Stramigioli [96]. For instance, if a robot lifts a mass its actuators consume potential energy (the energy flows through them) thus this energy is removed from the tank. In contrary, if someone pulls the mass down while the robot lifts it, the actuators extract energy from the system and the energy flows back to the tank. When the tank is empty, the controller is forbidden to inject energy into the system: the robot stops lifting the mass and the system is guaranteed to be passive Folkertsma and Stramigioli [96]. The energy tank is commonly defined as a virtual storage element with flow variable $\dot{s}$ and effort variable s such that:

$$E_{tank} = \frac{1}{2}s^2 \tag{2.65}$$

The flow variable $\dot{s}$ is often defined by the damping energy of the tasks Garofalo and Ott [108], Dietrich et al. [111, 112] and then integrated to find the energy value in the tank. It is possible to create a global tank monitoring the energy of all the tasks, or one tank for each task Dietrich et al. [112]. According to the value of the tank, the tracking of a task (or all tasks in the case of a global tank) is deteriorated to ensure the passivity of the system. This can be achieved by different ways:

- ▷ By using a Dirac structure to link the flow, effort and control variables Dietrich et al. [111, 112] (this method is not used in our case because of the complexity of the structure that will be involved)
- ▷ By modifying the dynamics of the controller states according to the energy in the tank Garofalo and Ott [108]
- ▷ By computing coefficients based on the tank value to directly regulate the tasks Dietrich et al. [112], Shahriari et al. [113].

An elegant way of using the first method on manipulators has been presented for multi-tasks in a strict-hierarchy using null-space projection in Dietrich et al. [111]. Moreover, regulating the size and flow of the energy tank are challenges that have been expressed in the domain Dietrich et al. [112], Tadele et al. [114]. Providing too much energy to the system may lead the controller to diverge from the equilibrium or to over-accumulate this energy. Likewise, an uncontrolled sharp release of the tank energy might lead to dangerous motions Shahriari et al. [113].

2.6.3.d State-of-the-art

For manipulator robots with flexible joints, Ott et al. [109], Albu-Schäffer et al. [115] present a passive framework using impedance control. Another method, successfully

implemented in Giusti et al. [116], couples Inverse Dynamics (ID) and passivity-based control by modifying the ID formulation to exploit the passivity-properties of the robot model.

However, using these approaches on humanoid robots is not straightforward. Indeed, humanoids robots are not ground-fixed to the environment, they are under-actuated systems. This under-actuated part, called floating base, prevents from using the full feedback linearization of the underlying system Acosta and Lopez-Martinez [117]. Moreover, it relies on external contact forces to control its motion. Besides, the penalization of the tracking tasks used in Dietrich et al. [112], Shahriari et al. [113] can compromise the balance of the robot. Thus, passivity-based control on humanoid robots is still an open-problem. In Garofalo and Ott [108], the method of Dietrich et al. [112] has been extended for a *fully* actuated humanoid robot for limit cycle control. Thus, this solution does not guarantee the passivity when constraints are activated in the QP due to the under-actuation of the humanoid robot.

The passivity method using impedance control has been first implemented on humanoid robots by Cheng et al. [118] and has lately been extended on the DLR robot TORO, leading to impressive results in multi-contact scenarios in Henze et al. [119]. Finally, Englsberger et al. [98] has designed a new controller respecting the Lyapunov stability and proving the passivity of the system even for conflicting sub-tasks, for *fully* actuated robots. This controller covers the complete range between Inverse Dynamics (with non-strict hierarchy) and Proportional Derivative with feed-forward (PD+) Control. However, as in Garofalo and Ott [108], for under-actuated robot, if a constraint becomes active, the passivity is no longer guaranteed. Nonetheless, it achieves great results with the humanoid robot TORO.

2.7 Force Control for Manipulation

2.7.1 Introduction

In order to perform an operation involving interactions of the robot arm with the environment it is needed to implement a specific control. Force control consists in using the force information given by the robot sensors in the controller. It allows to increase the robot adaptability to uncertain environments and to monitor the contact forces. Thus, force feedback can be used to enforce the safety of the control, avoiding huge contact forces which may lead to damages. This is particularly important for manufacturing and assembly tasks such as drilling or deburring where the amount of force required to perform the operation is important. Moreover, the controller used for these operations must also control the position of the arm or mechanical tool used by the robot. Indeed, the manipulation requires contact with the environment at precise position. Thus, the motion of the robot arm is constrained by this contact, constraints represented by the interaction forces.

In this section are presented the most common force control schemes present in the literature. These schemes have been mostly developed for manipulator robots and then adapted to robots with more DoFs. In particular, for humanoid robots, the solutions presented can be used on the end-effectors of the robot, for instance as specific tasks in the whole-body controller.

In the case of the robot TALOS, the forces are measured by the force sensors connected to the wrists and ankles of the robot. These force sensors can accurately provide forces along six DoFs. This vector F^m consists of the three orthogonal force components and the three orthogonal torque components operating at the end-effector with respect to the fixed Cartesian coordinate system (named the world W). Similarly, the position of the end-effector x in the constraint or task space is a six DoFs vector with three orthogonal displacements (distance between the end-effector and the world) and three angles (orientation of the end-effector with respect to the world). The Denavit-Hartenberg Bejczy [28] representation of the joint angles are set in the variable q_j and the generalized joint torques are given by τ.

To achieve manipulation operations with both position and force requirements the researchers have implemented solutions which create a dynamic relationship between position and force variables in addition to the control of one variable. Indeed, it is not possible to simultaneously impose force and position values on the environment. The two most common solutions are dual, one performing a direct closed-loop force control, the other performing an open-loop force control, and are the following Siciliano et al. [2]:

- ▷ The admittance control: Commands a force reference and builds a relationship between the end-effector motion and the error from the reference force Albu-Schäffer et al. [115], Keemink et al. [120], Yu and Perrusquía [121]. This solution allows to control the contact force to a desired value, with an open-loop position control (see Fig.2.11).
- ▷ The impedance control: Commands a position reference and builds a relationship between the external forces and the error from the reference position Lu and Meng [122], Anderson and Spong [123], Albu-Schaffer and Hirzinger [124], Schindlbeck and Haddadin [125]. This solution allows to control the position of the end-effector to a desired value, with an open-loop force control (see Fig.2.12). The resulting impedance can be linear or nonlinear (if there is a force feedback).

These two control strategies are complementary, they implement opposed solutions to achieve the same goal and thus behave well in different situations: the two controllers exhibit opposite stability characteristics depending on the stiffness of the environment. The admittance control is better suited for interactions with soft environments or operations in free space (see Section 2.7.2) and the impedance control is better suited for dynamic interactions with stiff environments (see Section 2.7.3).

Both methods aim to implement a task-space relationship following a mechanical impedance Siciliano et al. [2]:

$$\begin{aligned} \Delta x &= x^m - x^* \\ F_{ext} &= M_x \Delta\ddot{x} + D_x \Delta\dot{x} + K_x \Delta x \end{aligned} \tag{2.66}$$

where x^*, x^m are the desired and measured positions of the end-effector, F_{ext} the aimed external force measured at the end-effector, M_x is a mass matrix, D_x a damping matrix and K_x a stiffness matrix. Because the measurement of the acceleration $\ddot{x}^m$ can be noisy, the mass term is often neglected and set to 0.

Others strategies have been implemented to combine the advantages of the admittance and impedance control or of the position and force control. The first one is called the hybrid position/force control Anderson and Spong [123], Raibert and Craig [126], and more recently the hybrid admittance-impedance control Ott et al. [127], Kim et al. [128] (see Section 2.7.5). The second one is the parallel force/ position control Chiaverini and Sciavicco [129], Siciliano [130] described in Section 2.7.6.

2.7.2 Admittance Control

In the admittance control approach the robot is position controlled. The environment is assumed to behave as an impedance system, with elastic and dissipative components while the robot behaves as an admittance system. This method uses closed-loop force control and open-loop position control. It presents interesting properties when creating contact with compliant surfaces (direct control on the force and little use of position control) or when the environment is used to guide the end-effector. A simple control scheme is presented in Fig.2.11. In this scheme the mass term of Eq.2.66 is neglected and the desired force F^* to apply is related to a reference position $x^{ref} = K_c(F^* - F^m)$ where K_c is the compliance (or stiffness) matrix Salisbury [131]. Thus, Eq.2.66 is simplified in Siciliano et al. [2]:

$$K_x K_c(F^* - F^m) = D_x \Delta\dot{x} + K_x \Delta x \tag{2.67}$$

And the close-loop can be done by a PD+ on the end-effector velocity (see Eq. 2.68). The PD+ control can be of different forms, it can take an inner close-loop in position control with the term proportional on the position error $\Delta x = x^* - x^m$ (modified admittance control) or not.

In the example given in the Fig.2.11, we set $D_x = I$ and choose the gains of the PD+ to be $K_x K_c = D_{adm}$, $K_x = P_{adm}$. Thus we obtain the following equations and notations:

$$\begin{aligned} e_f &= F^* - F^m \\ \dot{x}^* &= \dot{x}^m + D_{adm} e_f - P_{adm} \Delta x \end{aligned} \tag{2.68}$$

with F^*, F^m respectively the desired and measured forces at the end-effector, $\dot{x}^*, \dot{x}^m$ respectively the desired and measured velocities of the end-effector, P_{adm}, D_{adm} the proportional and derivative gains for the admittance control, q^*, q^m respectively the desired and measured joint positions, $\dot{q}^*, \dot{q}^m$ respectively the desired and measured joint velocities and i the motor currents.

Then in the example, the desired end-effector velocity is saturated in the example and then v^*, q^* are computed using the Inverse Kinematics. The position controller to obtain the motor current is usually done by a simple PID:

$$i = P(q^* - q^m) + I \int_0^t (q^*(h) - q^m(h))dh + D(\dot{q}^* - \dot{q}^m) \tag{2.69}$$

with P, I, D the respective proportional, integral and derivative gains. Proportional gains for this kind of control are often very high, and integral ones small or equal to zero. This is making the robot extremely rigid, but the solution is simple to deploy.

An overview of admittance control for physical interactions with human can be found in Keemink et al. [120], which proposes guidelines to achieve high-performance

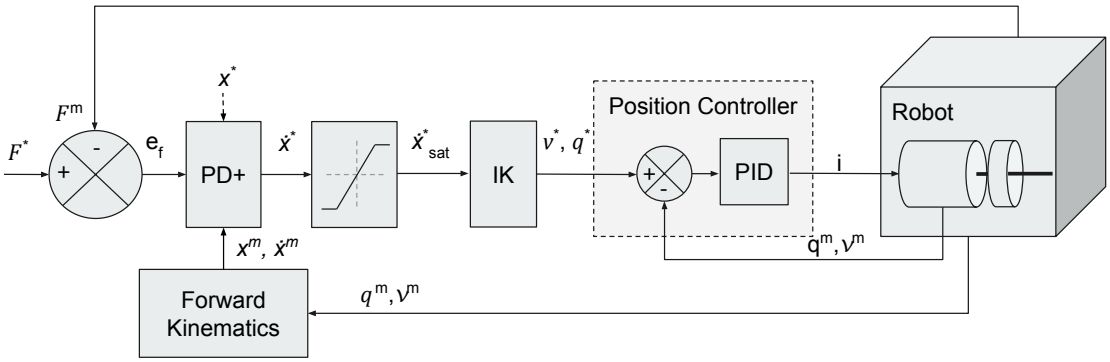

Figure 2.11: Simple Admittance Control Scheme.

control. A passive based admittance controller is proposed in Albu-Schäffer et al. [115] for flexible joint robots, by choosing appropriate gain values. In Fonseca et al. [132], the authors propose a task-space admittance controller adapting the inertia matrix online to avoid its ill-conditioning.

2.7.3 Impedance Control

As opposed to admittance control, the impedance approach assumes that the environment has inertia or kinematic constraints, which given force inputs produce motion as outputs. This means that the environment acts as an admittance and thus the robot controller is defined as an impedance for consistency. When in rigid contact with the environment, the impedance approach allows to control the position of the contact, ensuring steady contact forces (depending on the rigidity of the surface) which can be regulated through the open-loop, or directly if a feedback loop is added.

Using impedance control, the robot can be directly torque controlled because the outputs of the impedance scheme are forces. Using the Jacobian matrix of the contacts J one can obtain the desired torques τ^* from the desired forces Albu-Schaffer and Hirzinger [124]. A simple impedance scheme is presented in Fig.2.12 where a mass-spring-damper model is used to achieve the mechanical impedance described in Eq.2.66. Using the equation of the dynamics (see Eq.2.4) adapted in the Cartesian space, one can link F^* and F_{ext} Siciliano et al. [2]. Indeed, using the system dynamics, a solution which respects Eq.2.66 is simply given by $\tau^* = J^T F^*$ Lu and Meng [122] with the following equations:

$$\begin{aligned} \Delta x &= x^* - x^m \\ F^* + F_{ext} &= M_x \ddot{x}^m + C_x \dot{x}^m + g_x \\ F_{ext} &= M_x \Delta\ddot{x} + D_x \Delta\dot{x} + K_x \Delta x \\ \tau^* &= J^T (M_x \ddot{x}^m + C_x \dot{x}^m + g_x - F_{ext}) \end{aligned} \tag{2.70}$$

with $g_x = J^{-T} g$ and for simplification we choose the mass matrix M_x to be equal to the natural Cartesian inertia $\Lambda = J^{-T} M J^{-1}$.

Then, the desired torque is given to a torque controller which computes the motor currents using the chain actuator models (see Chapter 3 for an example on

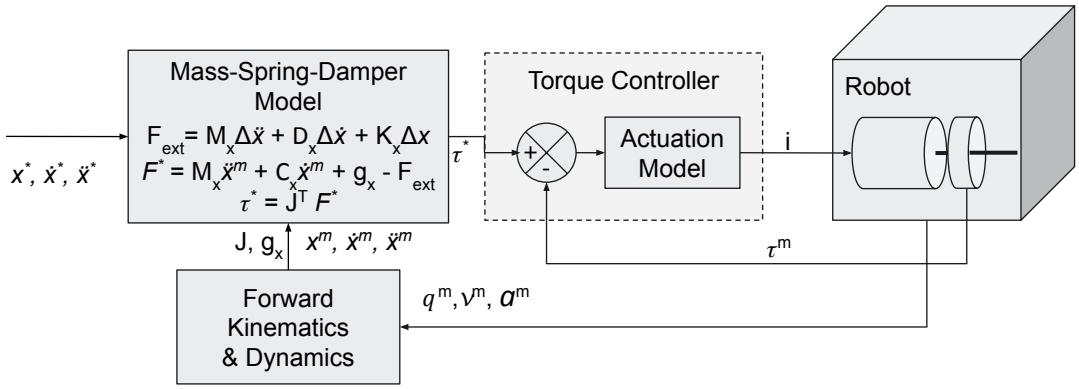

Figure 2.12: Simple Cartesian Impedance Control Scheme.

the elbow joint of TALOS). As opposed to the previous position control, torque control leads to compliant behaviors but is more complex to deploy on the robot.

Most of the impedance controllers neglect the mass term and only use a feed-forward in velocity and do not regulate the error Albu-Schäffer et al. [115], Schindlbeck and Haddadin [125], Ott et al. [127]. One reason is that, with this simpler formulation, it is possible to prove the passivity of the controller Albu-Schäffer et al. [115], Schindlbeck and Haddadin [125]. Moreover it is possible to add an outer force close loop to regulate the forces Schindlbeck and Haddadin [125], Ott et al. [127].

2.7.4 Stiffness/Compliance Control

Another solution to perform indirect force control using a closed-loop position control is the compliance (or stiffness) control Albu-Schäffer et al. [115], Albu-Schaffer and Hirzinger [124], Salisbury [131], Ajoudani et al. [133]. In this control scheme the position error is related to the contact force through a mechanical stiffness set in a specific matrix. If the contact forces exerted by the end-effector on the environment are not null while regulating its position, at the equilibrium the end-effector behaves like a generalized linear spring with compliance K_c with respect to the equivalent force F:

$$F = K_c \Delta x \tag{2.71}$$

K_c is the Cartesian stiffness matrix associated to the forces F. The stiffness control consists in designing this matrix to achieve the desired compliant control. A simple Cartesian stiffness control scheme is detailed in Fig.2.13. As opposed to the impedance control, the external forces behavior is chosen to respect the dynamics fixed by the stiffness matrix of Eq.2.71. Then the desired forces F^* are computed using a PD control with a gravity compensation Siciliano et al. [2], Albu-Schäffer et al. [55]:

$$F^* = K_c \Delta x + D_x \dot{x}^m + g_x \tag{2.72}$$

As in the impedance control, the output control can be torques which are transformed into motor currents by a torque controller.

To use the Cartesian stiffness matrix in the joint space, it needs to be transformed Salisbury [131], Albu-Schaffer et al. [134]. The stiffness matrix in the joint space

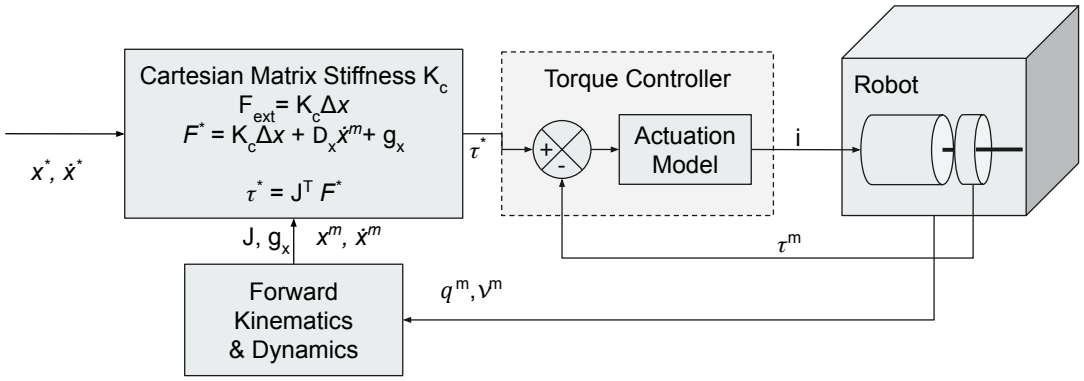

Figure 2.13: Simple Stiffness Control Scheme.

depends on the Jacobian J and thus on the robot configuration. And consequently the transform operator also depends on it. This mapping was first expressed in a simple form, by computing the stiffness matrix around the equilibrium position. However, this simplification removed the dependency of the stiffness matrix to J. Specifically, the terms representing the product of the changes in geometry and external forces of the stiffness matrix were removed Albu-Schaffer et al. [134].

Recent works have been improving the stiffness control by regulating the task space stiffness in function of the robot configurations Ajoudani et al. [133, 135], while solving the redundancy of the manipulator. In Albu-Schaffer and Hirzinger [124] is presented an impedance control enhanced by the addition of a local stiffness control. And a passive based control has been implemented for flexible joint robots in Albu-Schäffer et al. [115].

2.7.5 Hybrid Control

During interaction tasks, the environment imposes constraints on the end-effector, defining a constraint frame orthogonal to the task Siciliano et al. [2]. If the environment is rigid and there is no friction, the DoFs of the end-effector can be constrained in velocity (no translation or rotation along an axis) or in force (no force application along a direction or torque around an axis); these are called *natural constraints* because they are determined by the task geometry. Thus, the task space can be divided in two sub-spaces: velocity and force control, since both velocity and force cannot be controlled along any given direction. Moreover, the end-effector can only be controlled using the variables that are not already used by the natural constraints. The reference values of these variables define the *artificial constraints* which depend on the type of control chosen to realize the task Raibert and Craig [126]. The hybrid force/position or force/motion control consists in not controlling the variables imposed by the natural constraints to avoid conflicts and over-specification of the problem.

The structure of the hybrid control can be described as the following: two controllers are implemented, one acting on the force variables and another on the position/velocity variables. These two controllers can act on the whole system (on each DoF and tasks) and the final control is composed of a mix of the two control

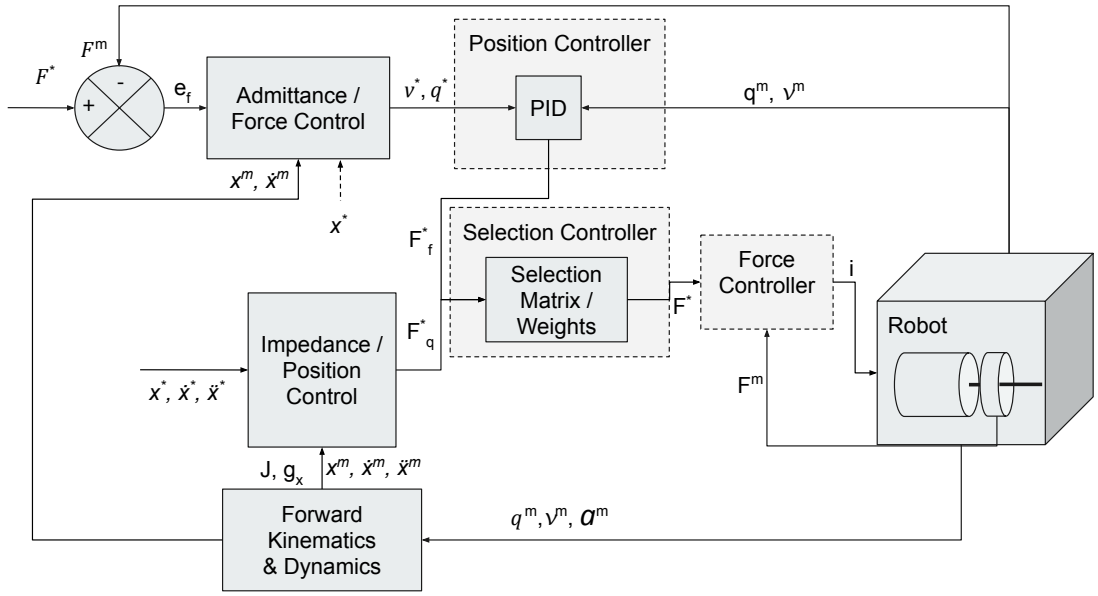

Figure 2.14: Example of Hybrid Control Scheme.

laws depending on the selected behavior for the tasks. This composition is realized using complementary selection matrices Siciliano et al. [2], Anderson and Spong [123], Raibert and Craig [126]. A simple example of such a scheme is presented in Fig.2.14.

In this example, the chosen control input is the forces F^*:

$$F^* = S_q F_q^* + S_f F_f^* \tag{2.73}$$

with S_q, S_f the respective selection matrix for the position and force control. Therefore, a task can be assigned specifying a desired force F^* and a desired velocity $\dot{x}^*$. This method can be used for velocity input in the same fashion, using a force controller to transform the desired forces from the position/velocity or impedance control in desired velocity. A more elegant way of handling this composition is to rewrite the equations 2.68 and 2.70 in terms of acceleration. Using the inverse dynamics control law, one can decouple the force and velocity variables to use the acceleration as control input for both the force and position/velocity controllers, simplifying the control scheme Siciliano et al. [2].

The first implementations of hybrid position/force control did not account for the dynamic coupling between the robot and the environment Raibert and Craig [126]. In our scheme (Fig.2.14) it corresponds to the case where the force control and position control blocks are not admittance and impedance ones. The drawback of this early method was that the structure of the problem needed to be changed accordingly to the task phases. Indeed, it can be necessary to switch from force to velocity/position control quickly, but ensuring a stable transition is hard. It mostly depends on the precision of the geometric environment model and planning. Contact loss and unexpected impacts were typically difficult to handle by these controllers because the selection matrices did not take into account these disturbances.

Then, hybrid methods have been developed to take into account the dynamics

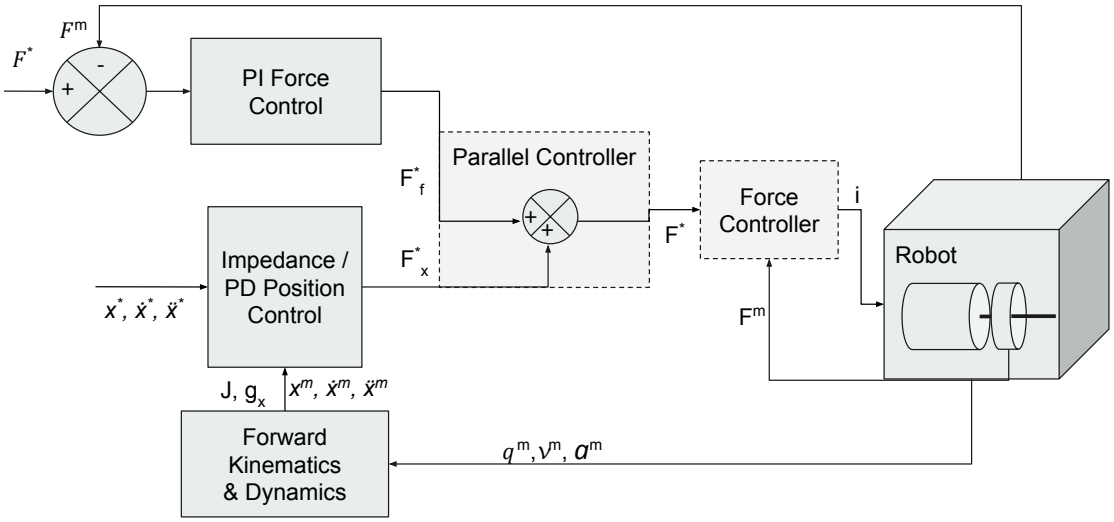

Figure 2.15: Example of Parallel Control Scheme.

of the interactions, such as using impedance control in Anderson and Spong [123]. More recent researches on hybrid control tries to couple admittance and impedance control using switching or weighting variables instead of selection matrices Ott et al. [127], Kim et al. [128]. The former paper presents an adjusting switching ratio depending on the duty cycle for fixed environment stiffness whereas the second one adjusts its control distribution weights according to the stiffness changes. In our illustrative scheme (Fig.2.14), it can correspond to the case where the force control and position control blocks are admittance and impedance ones and the selection controller uses ratio or weights.

2.7.6 Parallel Control

The parallel force/position scheme aims to combine the characteristics of the impedance and admittance schemes; controlling both position and force variables as the hybrid control schemes. It has been first presented to answer the issue of robustness with respect to the environment uncertainties of hybrid control Chiaverini and Sciavicco [129]. This solution is achieved by implementing two controllers acting in parallel, avoiding conflicts by prioritizing the force control over the position one Chiaverini and Sciavicco [129], Siciliano [130]. The dominance is achieved by implementing a PI force control loop in parallel to a PD position control loop. Thus, in a steady state situation it is possible to have a constant position error while the force error is equal to zero. This approach is close to the hybrid impedance control of Anderson and Spong [123] but does not rely on selection strategy. In both approaches the task planning provides force or position references along suitable task space directions. However, the parallel control is robust in case of planning errors in particular during contacts thanks to the force dominance rule Siciliano [130].

The Fig.2.15 illustrates the concept of parallel control, under the following set

of equations Chiaverini and Sciavicco [129], Siciliano [130] with force control input (acceleration input can be defined similarly):

$$\begin{aligned} F^* &= F_f^* + F_x^* \\ F_f^* &= K_f e_f + K_i \int_0^t e_f dt \\ F_x^* &= M_x \ddot{x}^* + D_x \Delta\dot{x} + K_x \Delta x \end{aligned} \tag{2.74}$$

One can choose a simpler impedance control as explained before by simply setting $F_x^* = K_x \Delta x - D_x \dot{x}^m$ Siciliano et al. [2].

In the Chapter 4 is presented the solution implemented for our whole-body controller to achieve a force operation task. It is close to a parallel force/position scheme with the addition of Coulomb contact constraints (see Section 4.3.4).

2.8 Model Predictive Control

It is possible to create controllers which consider a time horizon on a short time window instead of an instantaneous control. This type of control is called Model predictive control (MPC) Rawlings et al. [136] and is becoming an important field of research in robotics because it can handle non-linearity, constraints and system dynamics Huba [137]. The MPC is part of the branch of applied mathematics called Optimal Control. This method is able to predict the future behavior of the robot over a receding finite-time horizon, using the mathematical model of the system dynamics Rawlings et al. [138], Findeisen and Allgöwer [139]. Because of its prediction capability, MPC can for instance explicitly take into account stability. Passivity can be implemented by using the value function as a storage function and the optimal feedback as an output of the system, or by adding a sate constraint over the trajectory to respect the passivity inequality Raff et al. [140]. It is also possible to explicitly change the tasks which are activated inside the time horizon and to ensure state smoothness. Thus, using MPC, one can formulate and solve a sequence of controllers Todorov et al. [141] to handle complex problems. Recent MPC formulations are based on explicit discrete variables of the problem, in order to be able to choose contact phases Dai et al. [142].

The two main problems of the MPC approach are its computational cost and its limited time preview window (which is short, typically $2-3$ s). Indeed, as the Optimal Control Problem (OCPs) are often non-convex due to the robot model non-linearity, they lead to high computational load and to local minimum Boyd and Vandenberghe [12]. This is why the MPC has first been applied on simplified model Kajita et al. [32], Kim et al. [143] or systems with few DoFs Neunert et al. [144], Grandia et al. [145]. A first solution was to use the OCP to re-plan trajectories at low frequency while the robot was controlled by another controllerNeunert et al. [144].

For a system state $x \in \mathbb{R}^n$ and a control $u \in \mathbb{R}^m$, such that $\dot{x} = f(x,u)$, one can formulate the OCP as follows Mayne [146], Dantec et al. [147]:

$$\begin{aligned} \min_{x,u} \quad & \sum_{t=0}^{T-1} l^t(x(t),u(t))dt + l^T(x(T)) \\ \text{s.t.} \quad & x(0) = x_0 \\ \forall t \in [0,T], \quad & \dot{x}(t) = f^t(x(t),u(t)) \end{aligned} \tag{2.75}$$

with T the horizon time where is made the prediction, l^t, l^T are the running and terminal cost functions and x_0 is the initial state.

The cost functions are used to define the robot tasks, to regularize the state and control trajectories. The system constraints (such as the joint, velocity or torque limits) can be added in the cost as well or be put as constraints of the problem. Thus, the cost functions are often non-linear. During an iteration of control, the robot state is estimated and a new OCP is solved over the finite horizon between the estimated state and the future shifted from one sampling period.

In order to efficiently use a whole-body MPC directly as the main robot controller one will want to tackle the following problems:

- ▷ The computational time: To solve this problem one will need to use an OCP algorithm working at high frequency Carpentier et al. [13], Featherstone [15], or to rewrite the OCP to obtain a more efficient formulation in terms of computation.
- ▷ The number of iterations before convergence: To reduce this number, one can use a convex formulation Majumdar et al. [148], or a warm start.

To answer the first issue, shooting methods like Differential Dynamic Programming (DDP) Tassa et al. [149] have been introduced. In Bertsekas [150], Tassa et al. [151], Xie et al. [152] are presented efficient solvers able to deal with non linear costs, constraints and dynamics (see the following Section 2.8.1). Because of this problem of computational time, it was not possible to connect the MPC directly to the low-level servo controller Englsberger et al. [74], Herzog et al. [153] until Dantec et al. [147]. Indeed, to connect the two controllers it is necessary to evaluate the MPC at a compatible high frequency (at least 100Hz).

To find a good warm start for the second issue, Diehl et al. [154] proposes to use the solution from the previous iteration, but this heuristic is not robust to disturbances. Mansard et al. [76] uses the learning approach and specifically offline reinforcement learning (see Section 2.9) to find a warm start. In Dantec et al. [147] is presented the first successful experiment implementing whole-body model predictive control with state feedback on a torque-controlled humanoid robot, running at 100Hz. The control scheme is able to do whole-body target tracking, control the balance in front of strong external perturbations and avoid collision with an external object.

2.8.1 Differential Dynamics Programming Formulation

The DDP algorithm considers the OCP of Eq.2.75 in discrete form and denotes the control and state variables along the time horizon T as $U = \{u_0, u_1, ..., u_{T-1}\}$ and $X = \{x_0, x_1, ..., x_T, U\}$. To minimize the cost function of Eq.2.75, the DDP uses the "Bellman's principle of optimality" to transform the general problem into a sequence of simpler sub-problems. Indeed, this principle states that the remaining decisions of an optimal policy constitute an optimal policy with respect to the first decisions resulting state (in the absence of disturbances) Bellman [155]. For the DDP algorithm, it is used to proceed backward in time from $T-1$ to 0 while evaluating the optimal cost function at every node in the discretized state-time space. The

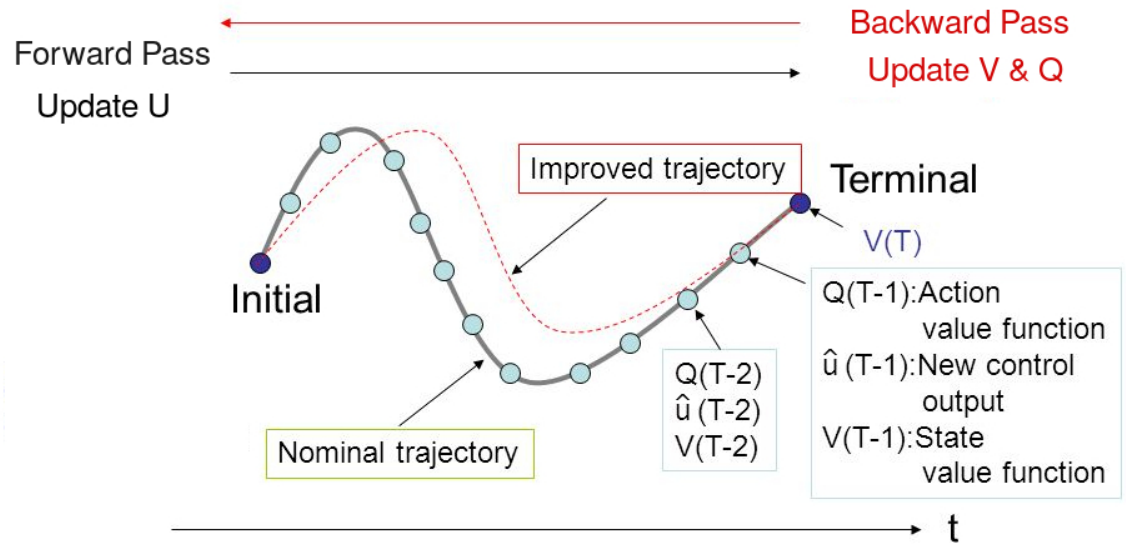

Figure 2.16: Scheme illustrating the DDP algorithm: first the backward pass is solved to update the value function and then the forward pass is executed to obtain the new state and control values.

evaluation is performed using a value function V based on the Bellman equation:

$$V^i(x_i) = \min_u \left[l^i(x_i, u_i) + V^{i+1}(f(x_i, u_i)) \right] \tag{2.76}$$

One can note Q, the function of the small variations of the value function around the state and command of iteration i. Q is locally approximated by a quadratic function (Gauss-Newton approximation) as follows Tassa et al. [151]:

$$Q(\delta x, \delta u) \approx \frac{1}{2} \begin{bmatrix} 1 \\ \delta x \\ \delta u \end{bmatrix}^T \begin{bmatrix} 0 & Q_x^T & Q_u^T \\ Q_x & Q_{xx} & Q_{xu} \\ Q_u & Q_{ux} & Q_{uu} \end{bmatrix} \begin{bmatrix} 1 \\ \delta x \\ \delta u \end{bmatrix} \tag{2.77}$$

Where the subscripts denote the partial derivatives of the terms for clarity, for instance $f_x = \frac{\partial f(x,u)}{\partial x}$. Using this approximation, one can obtain the algorithm of the DDP, detailed in the Appendix 2, see Algorithm 1. The optimal control local policy updating u (l.25 – 26) is defined as in Forget et al. [156], by:

$$\delta u^* = \arg\min_{\delta u} Q(\delta x, \delta u) = -Q_{uu}^{-1}(Q_u + Q_{ux}\delta x) \tag{2.78}$$

α (l.25) is a backtracking search parameter, set to 1, which can then be iteratively reduced (not presented in the algorithm for clarity) Tassa et al. [157]. An illustrative scheme of the DDP algorithm can be found in Fig.2.16.

It is possible in some cases to linearize this formulation and approximate the cost function to its quadratic form along the state trajectory Li and Todorov [158] to obtain an iterative linear quadratic regulator (iLQR). In the Chapter 3, we present a MPC using this DDP-iLQR formulation to control the elbow of TALOS in torque using the identified model of the actuator.

2.9 Learning approach

This thesis uses a model base approach to solve the different algorithms, relying on the kinematics and dynamics models of the robot. Another method is to use machine learning techniques, which propose new solutions in the robotics field, such as control, modeling or planning. We briefly expose their principles in this section. Two major approaches are Reinforcement learning Sutton and Barto [159] and Deep Learning LeCun et al. [160]. The former is a methodological approach which aims at finding a policy π to maximize a reward function assuming a value function. The value function $V(x, u)$ based on state and control provides a way to evaluate the cost to achieve a behavior. It can use a parametric functional space to find an appropriate control law and/or the model to be controlled Sutton and Barto [159], Jemin Hwangbo and Hutter [161]. Deep learning is a supervised approach which has recently gained a large popularity in computer vision LeCun et al. [160]. A current method is to use Convolutional Neural Network (used in Deep Learning) in the context of Reinforcement Learning Esteban et al. [162]. Interestingly, in order to discover the policy a large part of these works are using MPC Todorov et al. [141]. More generally, the assumptions related to optimal control are used in the different learning approaches Bottou and Bousquet [163].

An important part of the learning method is to generate an action and to evaluate its corresponding reward. The exploration of the controlled system and its specific environment is in general time consuming and very costly on a real robot. For this reason simulations are often used in a pre-exploration phase, which can require high computing performance. For instance, Reinforcement Learning can be used on the physical parameters of the robot to learn a model in simulation (see Section 2.3). The training of the control policy has to satisfy the physical constraints such as the joint limits. The main problem is then to deploy the trained policy on the physical system. Indeed, the methodology implies to perform some run on the real robot to take into account the differences between the simulation model and the real one. Recent works (Hwangbo et al. [164], Tan et al. [165]) demonstrate great results on quadruped robots, however, it is unclear how this can be realized on humanoid robot which are much more unstable than quadruped robots. This impossibility to realize numerous tests on the real robotic platforms has lead to develop specific learning algorithms which require few training data, called one-shot and few-shot learning Vinyals et al. [166], Snell et al. [167], James et al. [168].

2.10 Human-Robot Collaboration

Interactions between humanoid robots and humans raise great challenges as humanoid robots are complex systems due to their numerous DoFs and natural instability. During collaborative interactions, the tasks are generally performed by the human while the robot is passively following or assisting K. et al. [169]. However, by anticipating and predicting the human behavior and motions the robot would be able to perform its task more efficiently. To achieve this aim, cobotic applications require a good knowledge of the human behaviour. For example, to make a human and a humanoid robot perform a co-navigation or a co-manipulation task, a model of human walking trajectories is essential to make the robot follow or even anticipate

the human movements Maroger et al. [170].

In this section is presented a short state-of-the-art on the prediction processes and human-likeness trajectory planning methods used to proactively interact with humans. This section focuses on the literature used in the paper [24] realized in collaboration with Isabelle Maroger, presented in Section 4.5.

2.10.1 Proactive human-robot interactions

To proactively collaborate with a human, a humanoid robot needs to predict, or at least guess, its partner's future actions Jarrassé et al. [171]. The first experiment where a humanoid robot proactively interacts with a human was performed in Bussy et al. [172]. In this work, motion primitives like Stop, Side, Turn and Walk/Turn were used to generate the robot locomotion according to the human velocity. Since then, collaborative tasks are often aimed to be as proactive as possible in order to smooth the human-robot interactions.

For example, in Otani et al. [173], a co-manipulation task is aimed. To achieve this goal, the authors design a robot controller which can generate optimal motions in real time equivalent to those generated by a human. As this controller takes into account a whole-body dynamics of the human, it allows a more proactive interaction between the humanoid-robot and the human.

Furthermore, proactive co-navigation tasks have already been well studied. Indeed, in Teja S. and Alami [174], the authors propose a reactive trajectory planner for robots which takes into account the human predicted motions and goals to handle human-robot co-navigation. The estimation of the human movement is based on its current velocity and three different navigation modes are presented to make the robot move forward without colliding with the human. In this work, the goal is to smoothly avoid the human partner.

In Lanini et al. [175] and Lanini et al. [176], the authors target a collaborative handling task while walking, where the humanoid robot identifies its human partner's intentions. To guess future human motions, a training of a multiclass classifier of human intentions was performed using measurements collected from a human-human handling collaboration. The results of this training, namely the optimal features to discriminate a set of human motions, was then used to predict human intentions. Machine learning Aarno and Kragic [177] Kelley et al. [178] Li et al. [179] and probabilistic state machines Awais and Henrich [180] have already been studied to guess human intentions. In Maroger et al. [24], an OCP model is proposed to predict the human partner's future trajectory (briefly described in Section 4.5). A real-time approach is more complex but gives more versatility for future works. Indeed, optimisation problems allow to incrementally add obstacles or new constraints by adding new terms to the cost function.

2.10.2 Human trajectories during gait

Healthy human gait is a well studied field in bio-mechanics Winter [181]. However, the focus of those works is rarely on the modeling and the simulation of human walking trajectories. Nevertheless, in robotics, some authors studied this problem

to generate human-like paths for a robot to follow. Thus, models of human walking trajectories using parametric curves called clothoids Raković et al. [182] or OCP models Papadopoulos et al. [183], Arechavaleta et al. [184], Mombaur and Laumond [185] have been designed.

In Maroger et al. [186], the authors assessed those models according to a metrics evaluating the linear and angular distances between human trajectories and generated trajectories. The most human-like model was the one introduced in Mombaur and Laumond [185] which approximates the human by a holonomic system instead of a non-holonomic one. Indeed, a holonomic model allows lateral and oblique motions which better fits human locomotion. This is why a new OCP model, inspired from Mombaur and Laumond [185], is defined in Maroger et al. [187]. This model gives an accurate approximation of human behaviour. This OCP model is detailed and used in the Section 4.5.

Chapter 3

Identification, Modeling and Protection of the Robotic Chains

Contents

In this chapter are detailed the results obtained in the publication Ramuzat et al. [22].

3.1 Introduction

Having torque control on a humanoid robot is interesting for safe interaction with the environment and for its potential impact on cobotic applications. Historically, most of the humanoid robots are controlled in position and few robots, such as the TORO robot Englsberger et al. [97] or the DYROS-JET robot Park et al. [188], are torque controlled. Recently, a new generation of humanoid robots, including the TALOS robot, have shown the possibility to use their actuators either in position or in torque control modes Stasse et al. [189]. Several whole-body control architectures have been proposed (Cisneros et al. [103], Khatib et al. [190]) to compute and follow desired joint torques. However regulating joint torques is still an opened problem, which requires a low level controller able to follow a desired joint torque (see Section 2.3.4). In particular, some humanoid robots do not provide access to their low level controller (HRP-2 for instance), as well as a majority of industrial robots. In this situation, Khatib et al. [191] proposed a controller which takes in input a desired torque and provides a desired position to reflect the desired torque at the actuator. In a similar spirit, Prete et al. [192] implemented a torque control and inverse-dynamics control on the HRP-2 robot, originally designed for position control. They estimate the joint torques from the force sensors and the IMU by using the method proposed for the iCub robot by Nori et al. [193].

In this chapter, we present a solution which takes advantage of the capabilities of the commercially available TALOS robot. Its industrial standard EtherCat bus allows to gather numerous information about the actuators at a very high frequency ($\sim$ 2kHz). Thus, information related to the chain drive, the motor and joint positions or the joint torques can be read simultaneously. We would like to be able to generate extreme but safe motions, such as carrying heavy object or applying huge forces ($\sim$ 600N) for manufacturing operations (like drilling), without breaking or causing damages to the motor. This event may occur if a large torque is sustained for an extended period of time or when the robot is carrying a large additional payload. To avoid it, the current drawn in an actuator can be limited by adapting its motion. Moreover, the motor current can be saturated by the joint controller if the desired joint torque is above its limit.

Ideally, this could be done using a MPC (see Section 2.8). The use of a DDP as MPC allows to cope with the actuator flexibility while meeting the control loop timing constraints. However, such a controller requires the knowledge of the inertial parameters of the robot, of the motor chain drive and to have a well calibrated joint torque sensor. As said before, these parameters may not be provided by the manufacturer of the robot or may be inaccurate. Indeed, for TALOS, the chain drive parameters are not disclosed, to protect industrial conception, and they do not take into account the element modifying the inertia (such as cabling, glued elements or protection covers). Fortunately, the literature in system identification of serial manipulators proposes methods that allow to identify the joint drive gains and inertial parameters at once Bonnet et al. [57], Gautier and Jubien [58], Gautier and Briot [194] (see Section 2.3).

Alternatively, Reinforcement Learning algorithms can be used on the physical parameters of the robot to learn a model in simulation. The training of the control policy beware to satisfy the physical constraints such as the joint limits. The main

problem is then to deploy the trained policy on the physical system. Indeed, the methodology implies to perform some run on the real robot to take into account the differences between the simulation model and the real one. Recent works (Hwangbo et al. [164], Tan et al. [165]) demonstrate great results on quadruped robots, however, it is unclear how this can be realized on humanoid robot which are much more unstable than quadruped robots.

In this context, the objective of this chapter is to push the TALOS robot to its limits during joint torque control. For that purpose, we identify the joint chain drive of the elbow joint and the corresponding inertial parameters and use a DDP approach to protect the system (see Section 2.8.1). We organize the chapter as follows: Section 3.2 presents the methods used to create the mechanical and thermal model of the chain actuation. In Section 3.3 are described the mathematical process to identify the parameters of these two models. Section 3.4 details the optimal control scheme which takes into account the identified models to protect the robotic system. Finally, Section 3.5 presents the simulations and experiments used to identify the parameters and validate the controller on the robot TALOS.

3.2 Modeling of the Rigid Chain Actuators

3.2.1 Mechanical Model

The inverse dynamics model of the investigated system is used to calculate the motor torque τ_m and joint torques τ_j as a function of the joint and/or motor positions, velocities and accelerations. It is usually calculated using Newton-Euler equations W. Khalil [56]. The rigid chain actuators of TALOS are composed of a brushless motor, that can be controlled in current, connected to a harmonic drive and a torque sensor attached to its corresponding link (see Fig. 2.3). In this study the investigated sub-system is composed of the chain actuation of a single joint as represented in Fig. 3.1. The motor and joint positions are measured by two high-precision encoders (19 bits or 524,288 counts per revolution) and are considered equal thanks to the harmonic drive high stiffness. Consequently, in this chapter, the variables q, $\dot{q}$ and $\ddot{q}$ will be used to refer to the joint position, velocity and acceleration, respectively (we do not consider the under-actuation of the robot). The total inverse dynamics

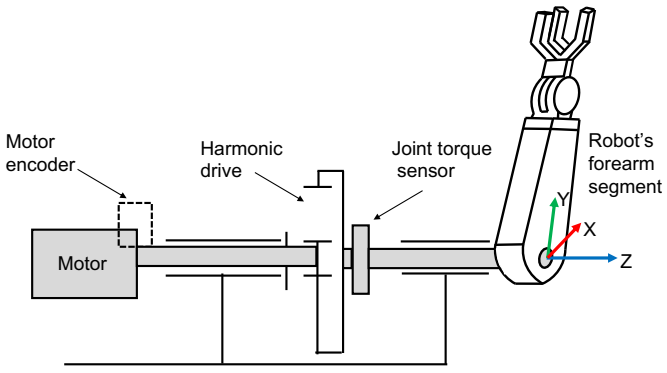

Figure 3.1: Scheme of the retained mechanical model.

equations of the retained model are:

$$\begin{aligned} \tau_m &= (I_m + I_j)\ddot{q} + g\Big(MY\sin q + MX\cos q\Big) + \tau_{fm} + \tau_{fj} \\ \tau_j &= I_j\ddot{q} + g\Big(MY\sin q + MX\cos q\Big) + \tau_{fj} \end{aligned} \tag{3.1}$$

and

$$\begin{aligned} \tau_{fm} &= F_{vm}\dot{q} + F_{sm}sign(\dot{q}) + off_m \\ \tau_{fj} &= F_{vj}\dot{q} + F_{sj}sign(\dot{q}) + off_j \end{aligned} \tag{3.2}$$

where I_m is the rotor inertia of the motor, I_j is the corresponding link inertia expressed at the joint level, F_{vm} and F_{sm} are the motor viscous and dry frictions, F_{vj} and F_{sj} are the viscous and dry frictions at the joint level, off_m and off_j are the offsets of the torque motor and joint, MX and MY are the first moment of inertia expressed at the joint level and g is the gravity.

The motor torque τ_m is related to the current by:

$$\tau_m = RK_m i_m \tag{3.3}$$

where R is the gear ratio, K_m is the joint drive of the manufacturer and i_m is the input current of the motor. Moreover, we have a similar relationship between the real applied torque at the joint, τ_j, and the torque measured by the sensor ,τ_{js}:

$$\tau_j = K_j\tau_{js} \tag{3.4}$$

where K_j is the gain between the two quantities.

We can express the simple dynamic of our system as:

$$\begin{aligned} \ddot{q} \;=\; & \frac{1}{I}\Big(RK_m i_m - F_v\dot{q} - F_s sign(\dot{q}) \;+\; off\Big) \\ & - (MY\sin q + MX\cos q)\frac{g}{I} \end{aligned} \tag{3.5}$$

where $I = I_m + I_j$, $F_v = F_{vm} + F_{vj}$, $F_s = F_{sm} + F_{sj}$ and $off = off_m + off_j$.

3.2.2 Actuator Thermal model

The robot TALOS has temperature sensors for each of its joint. In this section is described the method used to obtain a model of the motor temperature evolution. Let us consider R the winding resistor which increases linearly with the temperature of the motor T:

$$R = R_{TA}(1 + \alpha_{Cu}(T - T_A)) \tag{3.6}$$

where T_A is the ambient temperature, R_{TA} is the armature resistance at ambient temperature and α_{Cu} is the temperature coefficient of copper resistance.

We can now consider the Joule power losses together with the actuator model presented in the previous section:

$$P_J = Ri_m^2 = R_{TA}(1 + \alpha_{Cu}(T - T_A))i_m^2 \tag{3.7}$$

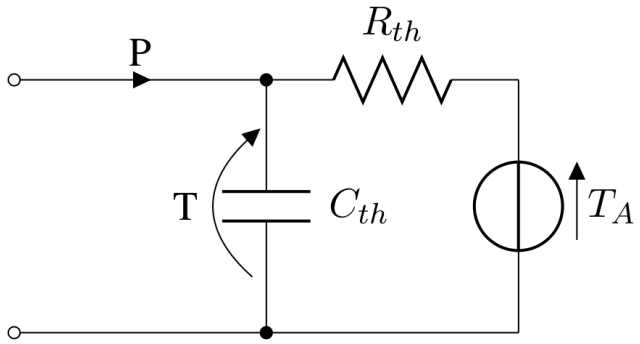

Figure 3.2: Equivalent electric model of the simple actuator thermal model.

In this model, some terms are neglected due to their small impact on the considered problem: The friction losses because the type of motor is DC brushless, the re-magnetization losses and the Eddy current losses.

We consider that casing conducts the temperature well enough. Therefore, the outer side of the casing (where we measure the temperature) is considered to have the same temperature as the windings, i.e. $R_{TA} = R_{th}$. This allows us to have a very simple model of the motor temperature evolution, depicted in Fig.3.2.2:

$$C_{th}\dot{T} = P_J - \frac{1}{R_{th}}(T - T_A) \tag{3.8}$$

with C_{th} is the thermal capacity, R_{th} is the thermal resistance. We set $R_{th} = R$.

3.3 Identification of the Rigid Chain Actuators

3.3.1 Identification of chain drive and segment inertial parameters

Usually the segment mass, center of mass and inertia are provided by the robot's manufacturer with a relatively good accuracy, whereas the friction and chain drive parameters are unknown. The provided current gain drive K_m is known to have 10 to 15% of inaccuracy Gautier and Briot [194]. Finally, the joint torque sensor should be calibrated prior to be used for control applications. Thus, we propose to use the inverse dynamics model and a total least square approach Gautier and Jubien [58] to identify the system described in Fig. 3.1 and its chain drive at once. To do so, two experiments were performed with and without using an additional payload.
When the inertial parameters of the dynamic model are expressed at the joint level, the model is linear, thus Eq.(3.1) becomes:

$$\begin{bmatrix}\tau_m \\ \tau_j\end{bmatrix} = \begin{bmatrix}RK_m i_m \\ K_j \tau_{js}\end{bmatrix} = \mathbf{W}\mathbf{\Phi} = \begin{bmatrix}cos(q) & sin(q) & \mathbf{D} & \mathbf{D} \\ cos(q) & sin(q) & \mathbf{0} & \mathbf{D}\end{bmatrix}\mathbf{\Phi} \tag{3.9}$$

where $\mathbf{D} = [\ddot{q} \quad \dot{q} \quad sign(\dot{q}) \quad 1]$ and $\mathbf{W}$ is the so-called regressor matrix. As quickly explained in Section 2.3.3, we need to create a vector containing all the parameters to be identified. This vector is ϕ, it contains the inertial parameters to be identified

and is defined as $\phi = [MX \quad MY \quad I_m \quad F_{vm} \quad F_{sm} \quad off_m \quad F_{vj} \quad F_{sj} \quad off_j]$. The TALOS robot embeds motor current and joint torque sensors, thus the regressor matrix $\mathbf{W}$ is full rank and the parameters can be identified separately.

To identify the current drive and the torque sensor gains, it is necessary to insert them into the vector ϕ Gautier and Jubien [58]. To do so, two regressor matrices that contain observations from two experiments have to be considered: $\mathbf{W}_0$ for the experiment without payload, and $\mathbf{W}_1$ for the one with a known payload at the end-effector. The Eq.(3.9) can be reformulated as:

$$\begin{bmatrix} -\mathbf{W}_0 & i_m & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -\mathbf{W}_0 & \mathbf{0} & \tau_{js} & \mathbf{0} & \mathbf{0} \\ -\mathbf{W}_1 & i_m & \mathbf{0} & -\mathbf{W}_{up} & -\mathbf{W}_{kp} \\ -\mathbf{W}_1 & \mathbf{0} & \tau_{js} & -\mathbf{W}_{up} & -\mathbf{W}_{kp} \end{bmatrix} \begin{bmatrix} \phi \\ RK_m \\ K_j \\ \phi_{up} \\ \phi_{kp} \end{bmatrix} = 0 \tag{3.10}$$

$$\mathbf{W}_{tot}\mathbf{\Phi}_{tot} = 0 \tag{3.11}$$

where $\mathbf{W}_{up}$ and $\mathbf{W}_{kp}$ are the observation matrices corresponding respectively to the unknown and known payload inertial parameters.

As noted by Gautier et al. Gautier and Jubien [58] $\mathbf{W}_{tot}$ is a full rank matrix because of the measurement perturbations. Therefore, the system described in Eq.(3.11) is modified to the closest compatible one with respect to the Frobenius norm:

$$\hat{\mathbf{W}}_{tot}\hat{\mathbf{\Phi}}_{tot} = 0 \tag{3.12}$$

where $\hat{\mathbf{W}}_{tot}$ is the closest rank deficient matrix from $\mathbf{W}_{tot}$ and is calculated using the singular value decomposition of $\mathbf{W}_{tot} = USV^T$.

The solution $\hat{\mathbf{\Phi}}_{tot}$ is given by the last column of V and is scaled Gautier and Jubien [58] using the known position of the center of mass of the payload along the Y-axis (see Fig.3.1).

The success of the inertial parameters identification relies on maintaining them excited using optimal motions Bonnet et al. [57]. An exciting joint trajectory is defined using a double S-curve spanning the whole range of motion of the elbow joint. The duration of each phase (acceleration, constant velocity and deceleration) of the S-curve is set to be equal. The velocity plateau is increased by steps of 20% up to its maximal value. A video illustrating the exciting joint trajectories of the arms is available at the following link: https://peertube.laas.fr/videos/watch/ae07428c-f25a-4637-a998-7e6885db3986.

Joint derivatives are obtained to fulfill the regressor matrix using centered differences from the measured joint positions. All the recorded data are low-pass filtered at 5Hz (Butterworth filter, zero-phase lag order 5) and the number of sample is decimated to reduce the noise influence Gautier [195]. The S-curve motions are considered exciting since the condition number of the base parameter regressor matrix is low ($cond(\mathbf{W}_b) = 36$).

3.4 Differential Dynamic Programming Optimal Control Scheme

This section presents the optimal control scheme used to find the control sequences required to perform the desired motion. In Section 2.8 is presented the DDP formulation (see Algorithm.1), we use a modified version here. In this chapter, we use the formulation given in Forget et al. [156], Kumar Hari Shankar Lal Das et al. [196], which maximizes the performance of the desired motion with respect to the control under the actuator physical limitations constraints.

3.4.1 DDP State space representation

In state space, we denote the state vector $x = [q, \dot{q}]$ and the command vector $u = [i_m]$. We then represent the direct dynamics model (Eq.3.5) as the following:

$$\dot{x} = f(x,u) = f(q,\dot{q},i_m) = \begin{bmatrix} \dot{q} \\ \frac{1}{I}\big(RK_m i_m - F_v\dot{q} - F_s sign(\dot{q}) \ + \ off\big) - \ (MY\sin q + MX\cos q)\frac{g}{I} \end{bmatrix} \tag{3.13}$$

3.4.2 Treatment of Constraints

In comparison to the work achieved in Forget et al. [156], we do not solve a box QP problem to bound the command vector. As introduced in Kumar Hari Shankar Lal Das et al. [196], we have instead considered the joint and actuator mechanical limits as constraints on the state and control space in the cost function formulation. The retained constraints functions are expressed by the following equations, referred as 'exponential barrier', at each time-step t:

$$\begin{aligned} max(c_t) &= 1 - \lambda(c_{max} - c_t) \\ min(c_t) &= 1 - \lambda(c_t - c_{min}) \\ \\ C_s(c_t) &= e^{\lambda max(c_t)} + e^{\lambda min(c_t)} \end{aligned} \tag{3.14}$$

where c_{min}, c_{max} are the corresponding lower and upper mechanical limits on angular position, velocity and torque, i.e. $c_t \in \{x(t), u(t)\}$. λ is a positive constant which defines the smoothness of the function. The higher it is, the quicker the cost will increase when approaching the limits (depending on the difference between the limit and the current state).

With these constraints, the total cost function will increase and reach a very high cost near the limits, keeping the system safe, in its mechanical bounds.

The control problem formulation is expressed as an open-loop control $u = u(t,x) \in U$ which can minimize or maximize a cost function along a given time interval $t \in [0,T]$ and from an initial state $x(0) = x_0$ value. The most generic expression for the cost function can be written as follows:

$$J(x_0) = C_f + C_r + C_s \tag{3.15}$$

where $C_f = h(x(T))$ is the final cost, $C_r = \int_0^T c(x(t), u(t, x(t)))dt$ is the integral of the running cost $c(x, u)$ which encapsulates the task objectives and C_s is the cost value imposed by the constraints.

For a non-linear dynamics Eq.(3.13) and a non-quadratic cost Eq.(3.15), optimal control solutions can be obtained using full DDP. However, as DDP is computationally expensive, an iterative LQR (iLQR) approach is considered Li and Todorov [158]. The iLQR method is relying on linearizing the dynamics and approximating the cost function to quadratic form along the x trajectory. This control approach is briefly summarized in the next section.

3.4.3 Iterative Linear Quadratic Regulator (iLQR)

To mitigate the DDP computational time it is possible to locally linearize the dynamics and approximate the cost function to quadratic form around the current point in state space (along the x trajectory). This method is called the iterative linear quadratic regulator (iLQR) approach Li and Todorov [158] and we use the same formulation as in Kumar Hari Shankar Lal Das et al. [196].

The iLQR is initialized with a nominal control sequence and the corresponding state trajectory (x_0, u_0). The dynamical system is then linearized as in Eq.(3.16) and the cost function is approximated by the quadratic form Eq.(3.17), eventually a local LQR problem is solved. Using this solution, the states and the control sequence are improved iteratively.

We denote the dynamic partial derivatives $f_u = \frac{\partial f(x,u)}{\partial u}$ and $f_x = \frac{\partial f(x,u)}{\partial x}$. And the cost partial derivatives $l_u = \frac{\partial J(x,u)}{\partial u}$, $l_{uu} = \frac{\partial^2 J(x,u)}{\partial u^2}$, $l_x = \frac{\partial J(x,u)}{\partial x}$, $l_{xx} = \frac{\partial^2 J(x,u)}{\partial x^2}$, $l_{xu} = \frac{\partial^2 J(x,u)}{\partial x \partial u}$, $h_x = \frac{\partial h(x,u)}{\partial x}$ and $h_{xx} = \frac{\partial^2 h(x,u)}{\partial x^2}$.

$$\dot{\delta x} = f_x \delta x + f_u \delta u \tag{3.16}$$

$$\begin{aligned} \Delta J \;=\; & h_x^T \delta x(T) + \delta x^T(T) h_{xx} \delta x(T) + \\ & \int_0^T \Big(l_x^T \delta x + l_u^T \delta u + \delta x^T l_{xx} \delta x \; + \delta x^T l_{xu} \delta u + \delta u^T l_{uu} \delta u \Big) dt \end{aligned} \tag{3.17}$$

At every iteration, Eq.(3.16) and Eq.(3.17) are solved and $(\delta x, \delta u)$ are deduced from the resolution of a modified Ricatti-type system. Then the new improved sequence is generated by $x \leftarrow x + \delta x$ and $u \leftarrow u + \delta u$. As for a classical DDP, it is a succession of backward and forward phases (see Section 2.8.1). When $\Delta J \approx 0$, the iLQR converges thus giving an optimal control sequence $u^* \in U$ and the corresponding optimal state trajectory x^*.

3.4.4 State space partial derivatives

This section details the state space partial derivatives needed by the iLQR algorithm, obtained from eq. (3.13). The $sign()$ function is not differentiable in 0, therefore we have chosen to approximate the $sign(\dot{q})$ term by an hyperbolic tangent: $\tanh(\mu \dot{q})$,

where $\mu = 1000$. We obtain the following new equation for the dynamic of the system:

$$f(x,u) = \begin{bmatrix} \dot{q} \\ \frac{1}{I}\big(RK_m i_m - F_v\dot{q} - F_s \tanh(\mu\dot{q}) + off\big) \\ -(MY\sin q + MX\cos q)\frac{g}{I} \end{bmatrix} \tag{3.18}$$

The partial derivative in u can be directly computed because the function is linearly dependent in i_m:

$$f_u = \begin{bmatrix} 0.0 & \frac{1}{I}RK_m \end{bmatrix}^T \tag{3.19}$$

Concerning the derivative in x, we have non-linear dependencies between $\ddot{q}$ and $\dot{q}$, and $\ddot{q}$ and q. Moreover the relationship between $\dot{q}$ and q is not explicit. Therefore we use the spatial finite difference discretization of the equation to obtain f_x at each iteration i:

$$x_{i+1} = f(x_i, u_i) \tag{3.20}$$

$$f_x = \begin{bmatrix} \frac{f(\begin{bmatrix} q_i + h/2 \\ \dot{q}_i \end{bmatrix}, u_i) - f(\begin{bmatrix} q_i - h/2 \\ \dot{q}_i \end{bmatrix}, u_i)}{h} \\ \frac{f(\begin{bmatrix} q_i \\ \dot{q}_i + h/2 \end{bmatrix}, u_i) - f(\begin{bmatrix} q_i \\ \dot{q}_i - h/2 \end{bmatrix}, u_i)}{h} \end{bmatrix} \tag{3.21}$$

3.4.5 Cost function

This section presents the cost function used in our system and the cost partial derivatives needed by the iLQR algorithm, obtained from the equation (3.13). Considering the actual state vector x, the desired state vector x^*, the actual command vector u and the actual torque on the elbow motor τ_m, we use the following cost function:

$$\begin{aligned} J &= (x-x^*)^T Q\,(x-x^*) + u^T R\,u \\ &\quad + C_s^T(x)\,W\,C_s(x) + C_s^T(\tau_m)\,P\,C_s(\tau_m) \end{aligned} \tag{3.22}$$

$$\begin{aligned} Q &= \begin{bmatrix} 40.0 & 0.0 \\ 0.0 & 0.01 \end{bmatrix} & W &= \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 0.1 \end{bmatrix} \\ R &= 0.0001 & P &= 10.0 \end{aligned} \tag{3.23}$$

The weighting matrix Q, W, R and P have been chosen to give a hierarchy from the most important error to control to the lesser one. Using the lexicographical order we have: $q \succ\succ \overline{\tau_m}, \underline{\tau_m} \succ\succ \overline{q}, \underline{q} \succ\succ \overline{\dot{q}}, \underline{\dot{q}} \succ\succ \dot{q} \succ\succ u$. Where $\overline{c}, \underline{c}$ are the corresponding lower and upper mechanical limits. The tracking of the trajectory (in position) is the prioritized task, followed by the one controlling the torque bounds.

We obtain the following cost partial derivatives $l_u = \frac{\partial J(x,u)}{\partial u}$, $l_{uu} = \frac{\partial^2 J(x,u)}{\partial u^2}$, $l_x = \frac{\partial J(x,u)}{\partial x}$, $l_{xx} = \frac{\partial^2 J(x,u)}{\partial x^2}$, $l_{xu} = \frac{\partial^2 J(x,u)}{\partial x \partial u}$:

$$l_u = 2Ru + 2\,\frac{dC_s(\tau_m)^T}{du}\,WC_s(\tau_m) \tag{3.24}$$

$$l_{uu} = 2R + 2\left[\frac{d^2C_s(\tau_m)^T}{du^2}WC_s + \frac{dC_s(\tau_m)^T}{du}W\frac{dC_s(\tau_m)}{du}\right] \tag{3.25}$$

$$\frac{dC_s(\tau_m)}{du} = \lambda(\tau_m)^2\, RK_m\Big(e^{\lambda(\tau_m)max(\tau_m)} - e^{\lambda(\tau_m)min(\tau_m)}\Big) \tag{3.26}$$

$$\frac{d^2C_s(\tau_m)}{du^2} = \lambda(\tau_m)^4\,(RK_m)^2\,C_s(\tau_m) \tag{3.27}$$

$$l_x = 2Q\,(x - x^*) + 2\,\frac{dC_s(x)^T}{dx}\,WC_s \tag{3.28}$$

$$l_{xx} = 2Q + 2\left[\frac{d^2C_s(x)^T}{dx^2}C_s + \frac{dC_s(x)^T}{dx}W\frac{dC_s(x)}{dx}\right] \tag{3.29}$$

$$\frac{dC_s(x)}{dx} = diag\Bigg(\begin{bmatrix} \lambda(q)^2\Big[e^{\lambda max(q)} - e^{\lambda min(q)}\Big) \\ \lambda(\dot{q})^2\Big(e^{\lambda max(\dot{q})} - e^{\lambda min(\dot{q})}\Big) \end{bmatrix}\Bigg) \tag{3.30}$$

$$\frac{d^2C_s(x)}{dx^2} = \begin{bmatrix} \begin{bmatrix} \lambda(q)^4\,C_s(q) & 0.0 \\ 0.0 & 0.0 \end{bmatrix} \\ \\ \begin{bmatrix} 0.0 & 0.0 \\ 0.0 & \lambda(\dot{q})^4\,C_s(\dot{q}) \end{bmatrix} \end{bmatrix} \tag{3.31}$$

with $\lambda(\tau_m) = 0.5$, $\lambda(q) = 10$ and $\lambda(\dot{q}) = 1$. These parameters have been chosen accordingly to the explanation on the λ of the section 3.4.2. The 'exponential barrier' on the position bounds will be sharper than the two others. It also depends on the difference between the limit and the current state. For instance the maximal difference in current is $12A$ whereas the one in position is $2.35rad$, then the $\lambda(\tau_m)$ do not need to be as big as the $\lambda(q)$.

3.5 Results

3.5.1 Identifying the thermal model

The room temperature (26°C during the experiment) is never the same as the ambient temperature for the motor. Indeed, the robot drives release a lot of heat, heating the motor to approximately 43°C even when it is unused. The experimental setup is the following: the robot is controlled in position, a sinusoidal command is sent to the elbow actuators and a heavy charge is put on the arm of the robot (Fig.3.6).

As represented in Fig.3.3, during the heating sequence of the experiment, the identified function fits the measured temperature evolution. On the other hand,

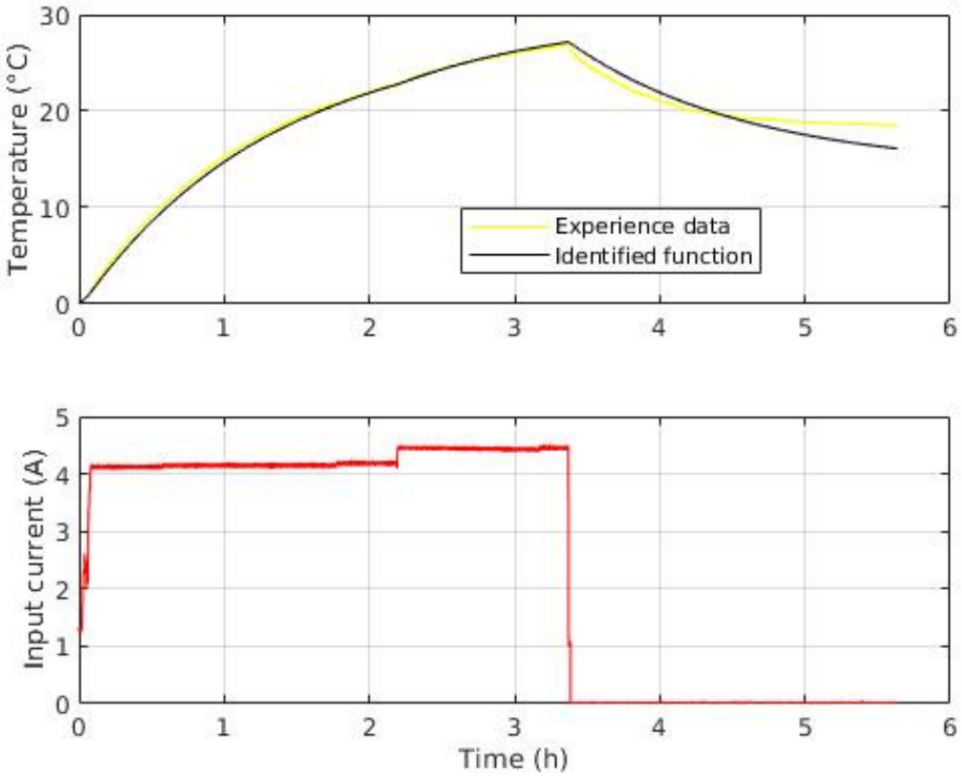

Figure 3.3: Thermal parameter identification

in the cooling part, the estimated and measured curves tend to diverge. The heat of the drives may cause the problem. Indeed, the temperature of the drives also increases during the experiment. But the drives and the motor are really close to one another, meaning that the temperature of the drives can not be set aside while studying the temperature of the motors: it makes the motors cool down slower. Nevertheless, results show that holding 20 kg at 5 cm of the elbow is not a problem for this specific joint in terms of temperature. As the chain actuators are similar for each joint of the robot TALOS, one can conclude that overheating the actuators of the robot during experiments is unlikely.

3.5.2 Identifying the chain drive and inertial parameters

Fig. 3.4 shows the results of the least square identification. It shows the fitting of the measured motor and joint torque when no payload was used. The corresponding Root Mean Square Difference (RMSD) was 0.6 N.m showing an excellent fitting. Fig. 3.6.c shows the estimate of joint torque from the current of the motor that is of importance for the proposed dynamic controller.
Table 1 details the comparison between the identified parameters and their values as provided by the manufacturer.

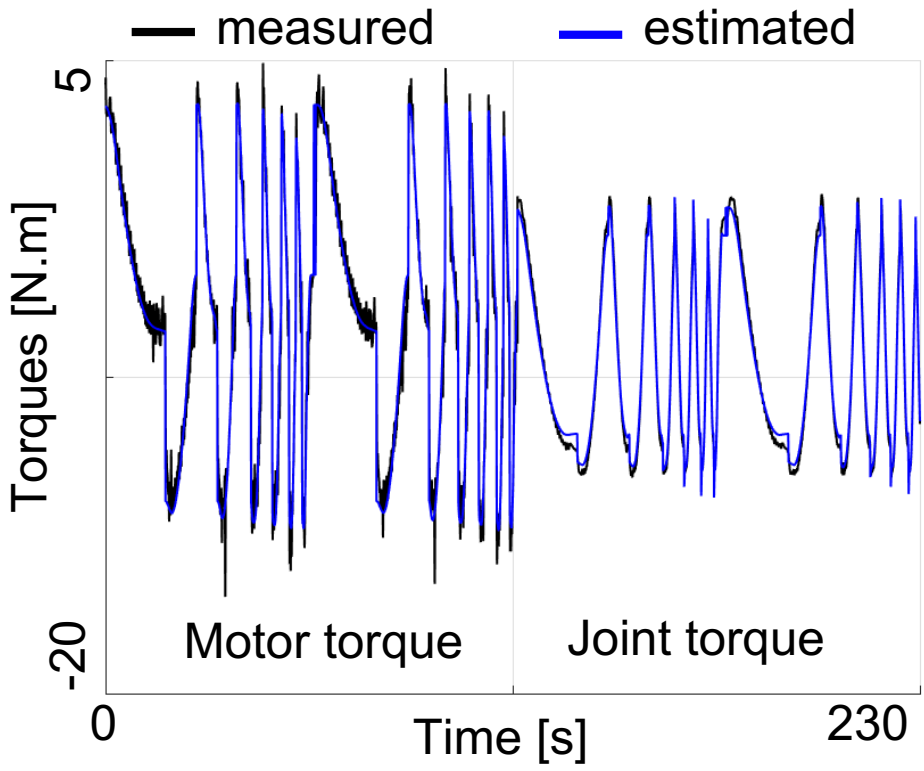

Figure 3.4: Results of the fitting of motor and joint torques used for the identification process.

Table 3.1: Results of the identification process and comparison with manufacturer data.

	CAD	ID	$\sigma_{\%}$
MX	-0.08	-0.11	2.1
MY	1.1	1.1	0.2
I_j	0.34	0.33	9.9
F_{Sj}	0	0.55	1.3
F_{Vj}	0	0.47	10.3
Off_j	0	-0.27	3.4
I_m	0.21	0.45	9.5
F_{Sm}	0	4.0	0.1
F_{Vm}	0	5.1	0.7
Off_m	0	0.86	7.9

Overall the identified segment inertial parameters are similar to the ones provided by the manufacturer. These parameters can be considered well identified due to their relatively low standard deviation $\sigma_{\%}$. The relative standard deviation gives in % a confidence index on the reliability of the identification of each parameter. See Presse and Gautier [197] for the detail of $\sigma_{\%}$ calculation. The joint torque sensor was well calibrated since its identified gain was K_j =1.015. The total joint drive gain

RK_m value is not disclosed, because of a confidentiality agreement with the robot manufacturer, but a difference of 16% with the value provided by the manufacturer was found.

3.5.3 Controlling the actuator

The iLQR is implemented as described in Forget et al. [156] and completed by the 'exponential barrier' constraints in the cost function. The cost functions and model dynamics are evaluated using the identified parameters (Table.3.1). This implementation is first validated in simulation with the use of Gazebo and then tested on the real TALOS robot. The noticeable contribution of our solution compared to Forget et al. [156] remains in this last point, the torque control is implemented on the robot and achieve a satisfying real time control (with more protections on the mechanical limits of the robot).

The experimental setup is the following: the robot is controlled in torque with high gains except for the elbow actuator. A sinusoidal command for the elbow actuator is sent to the iLQR algorithm which computes a resulting signal respecting the limits. This signal is then sent to the robot. A heavy charge is finally put incrementally on the arm of the robot, at 5cm of the elbow joint, until reaching 34kg (see Fig.3.6). The load moment arm when the load is perpendicular to the robot arm is equal to 16.67 N.m. Otherwise it is expressed similarly to the mass of the arm at the center of mass (see Eq.(3.13)): $LY \sin q + LX \cos q$ with LY and LX the first moment of inertia of the load.

The iLQR algorithm is executed on the robot with a $15ms$ preview window, a $3.3ms$ discretization and has an execution time of $300\mu s$ (using an Intel CPU i7-3612QE @ 2.1 GHz). On a standard laptop such as an Intel CPU i7-8850H @ 2.6 GHz it can be executed with a longer preview window ($100ms$) and with a faster control frequency ($1ms$ discretization) in a smaller execution time ($200\mu s$).

3.5.3.a Simulation

Fig.3.7 presents the action of the exponential barriers on the position joint limits (see Eq.3.14). A sinusoidal desired trajectory is given to track, but reaches the lower angular position limit: $-2.35rad$. The computed trajectory shows a plateau before reaching the limit, demonstrating the activation of the protection in the iLQR. We can notice the small oscillations at the top of the sinusoide, due to the dry frictions when the angular velocity reaches zero (see Fig.3.11). This causes a small delay in the computed trajectory.

Fig.3.8 depicts the tracking results on two cases: first without additional load and second by adding a 30 kg load to the forearm. In the first case, the trajectory follows the desired one with a small delay (∼0.1s), which may be causes by the dry friction as explain above. This bias can be removed by increasing the state constraint gains but the system will loose compliance. In the second case, the delay with the desired trajectory increases, in particular during the ascending phase of the sinusoidal command (the robot raises its arm and the load). Indeed, to match the desired trajectory quicker, the torques needed are high and the iLQR algorithm

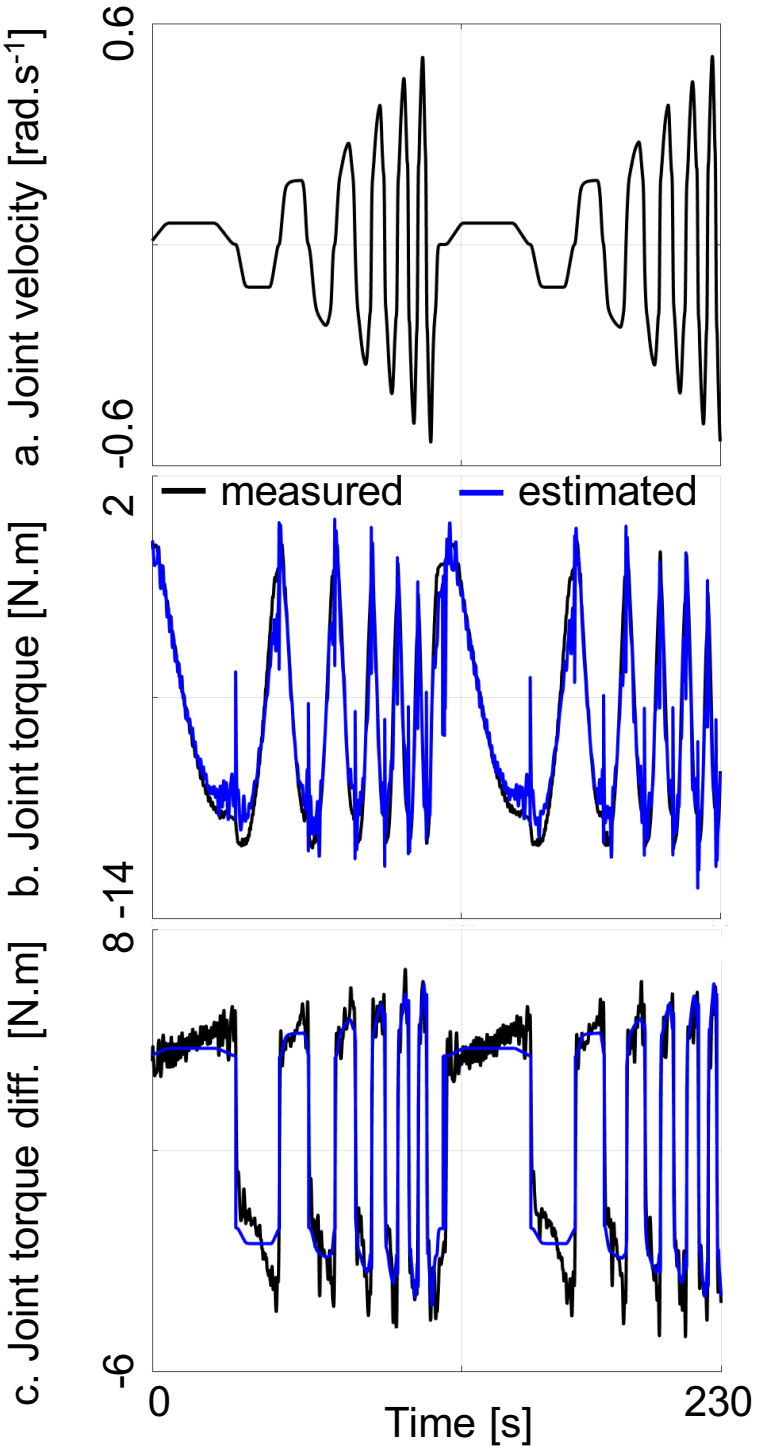

Figure 3.5: (a) Velocity profile used to excite the dynamics system. (b) Estimate of the joint torques from motor current and the identified model. (c) Estimation of the difference between the joint and motor torque.

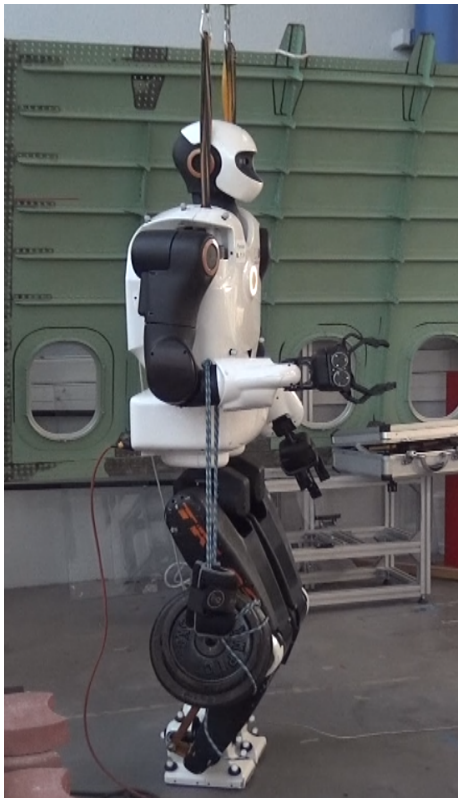

Figure 3.6: Experiment where TALOS is holding $34kg$ at $5\,cm$ of the elbow joint.

limits them. Nevertheless, in both cases the overall motion shape is respected.

The control trajectories depicted in Fig.3.9 show how the limitation of 6 A (set *a-priori*) is implemented with the exponential barrier given by Eq.3.14. The cyan trajectory represents the current command when there is no additional load and it does not reach the limit. The green trajectory represents the current command when the robot carries a 30kg load, it shows that the command reaches a plateau (around -5.6A) before the limit. It explains the delay of the computed angular trajectory of Fig.3.8, the tracking of the desired joint trajectory is degraded because the current is limited. This is due to the iLQR action to preserve the actuator, i.e. the exponential barrier is activated by the iLQR algorithm.

In comparison, Fig.3.10-bottom depicts the control going over the current limit in the loaded case when the current exponential barrier is not enabled in simulation. In this configuration the joint position tracking is better, as shown in Fig.3.10-up, but the actuator reaches its current limit.

3.5.3.b Experiments

Fig.3.11 presents the action of the exponential barriers on the position joint limits on the real robot. As in simulation, the desired trajectory reaches the lower angular position limit: $-2.35rad$. The computed trajectory shows a plateau before reaching the limit, demonstrating the efficiency of the protection in the iLQR. Notice the oscillations at the beginning of the sinusoidal movement (when the arm is raising), the arm of the robot has difficulties to perform a smooth trajectory. As thought in

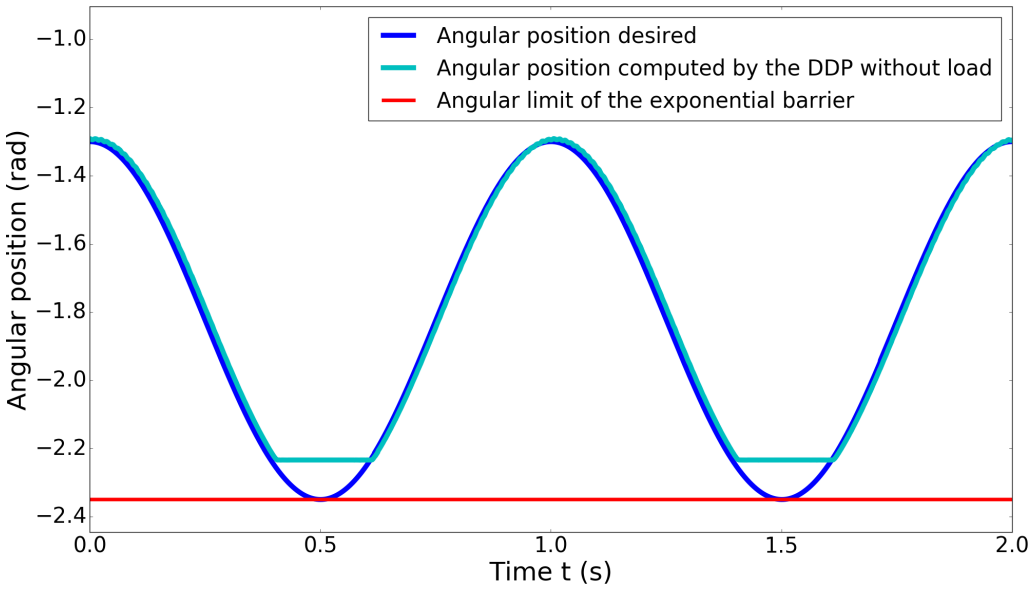

Figure 3.7: Simulated state trajectory illustrating the angular exponential barrier.

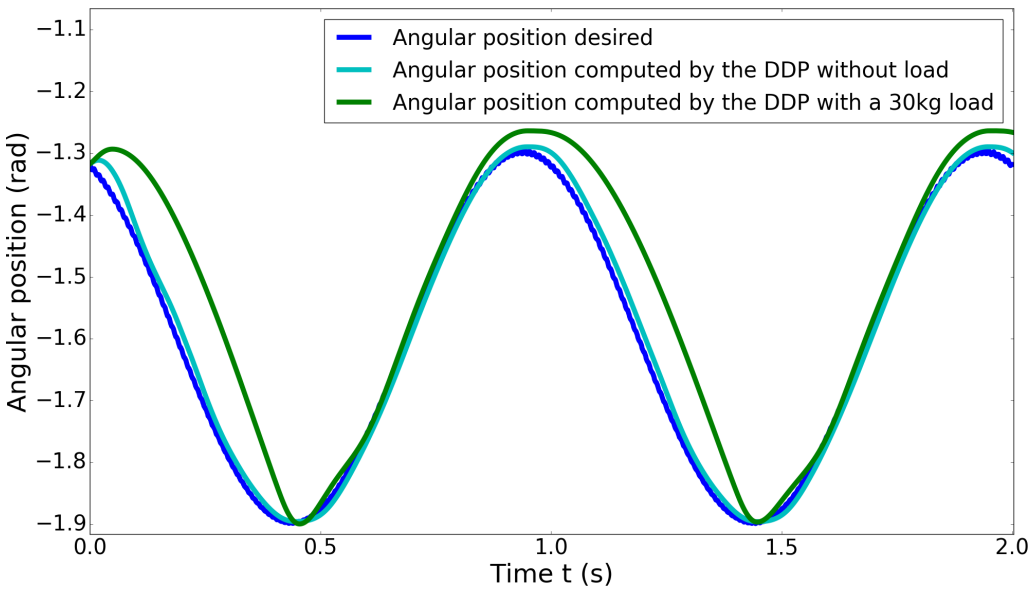

Figure 3.8: Comparison between simulated trajectories with and without load.

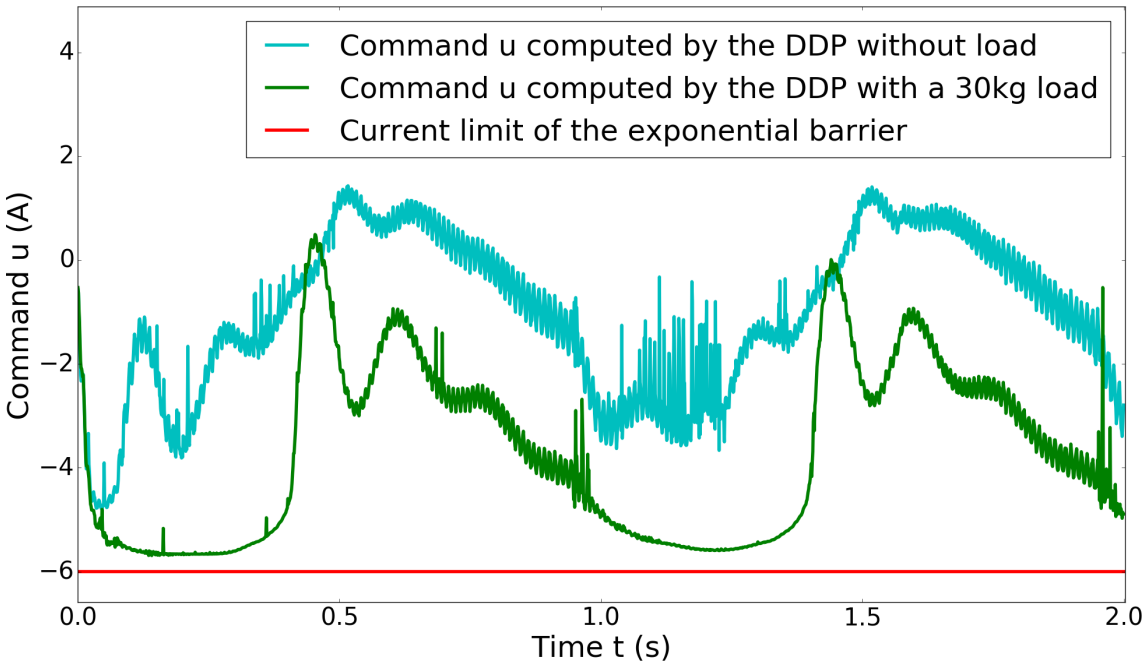

Figure 3.9: Simulated control trajectories with and without an additional load.

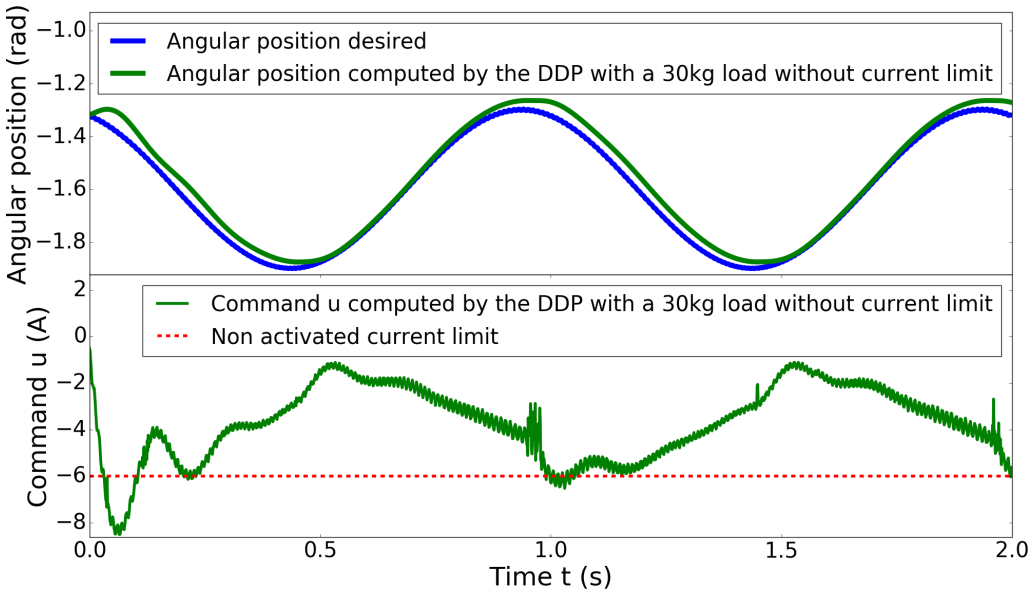

Figure 3.10: Simulated state and control trajectories in loaded case but without control input limitation.

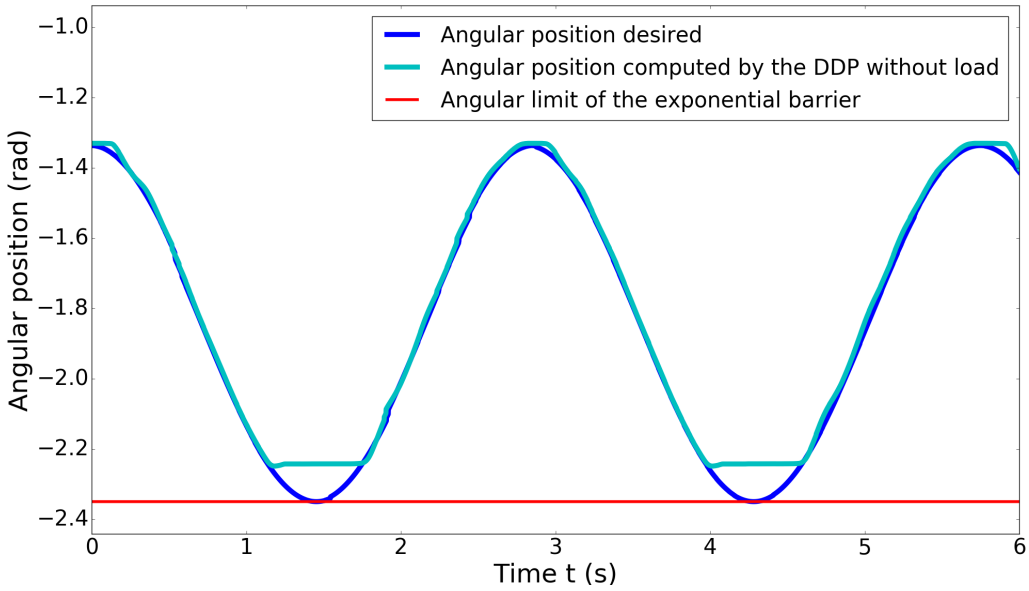

Figure 3.11: Experiments - State trajectory illustrating the angular exponential barrier.

the simulations, this is the consequence of the dry frictions, when the angular velocity $\dot{q}$ is approaching zero, the dry coefficient F_s increases to reach the Breakaway friction value (the sum of the Coulomb and static frictions). This behaviour can create the delay noticed in the simulations. This is not currently taken into account in our model but would be in the future.

Fig.3.12 shows a comparison between the desired joint trajectory and the ones obtained without and with a 34kg load (we increase the load to have a better demonstration of the activation of the current exponential barrier). As expected, the trajectory without any load has a small delay and oscillates when the angular velocity is around zero. The trajectory obtained when the robot is carrying the additional 34 kg displays greater oscillations due to the load movements. It also presents a bigger degradation of the trajectory than in simulation, the sinusoidal movement is reduced (stopped at $-1.7rad$ instead of $-1.9rad$) because large torques are necessary to achieve it (but are prohibited by the iLQR algorithm, see Fig.3.13).

The Fig.3.13 presents the control trajectories computed by the iLQR without and with the load. In the first case (cyan line), the command is far from the limit and do not activate the barrier. In the second case (green line), as in simulation, the command reaches a plateau before the limit, around -5.3 to $-5.6A$. The current is more reduced than in simulation, inducing a more degraded state trajectory in Fig.3.12. The plateau is not as smooth as in the simulation, due to the computation time which is quite high, leading to picks in the calculation. Indeed, in these experiments the computation time of the overall control system is around 1 to $1.5ms$. Compared to the control frequency of the robot, which is $1kHz$, this duration is large.

A video describing the experiments on the robot can be found at the following location: https://youtu.be/YNoSnU7w4FY.

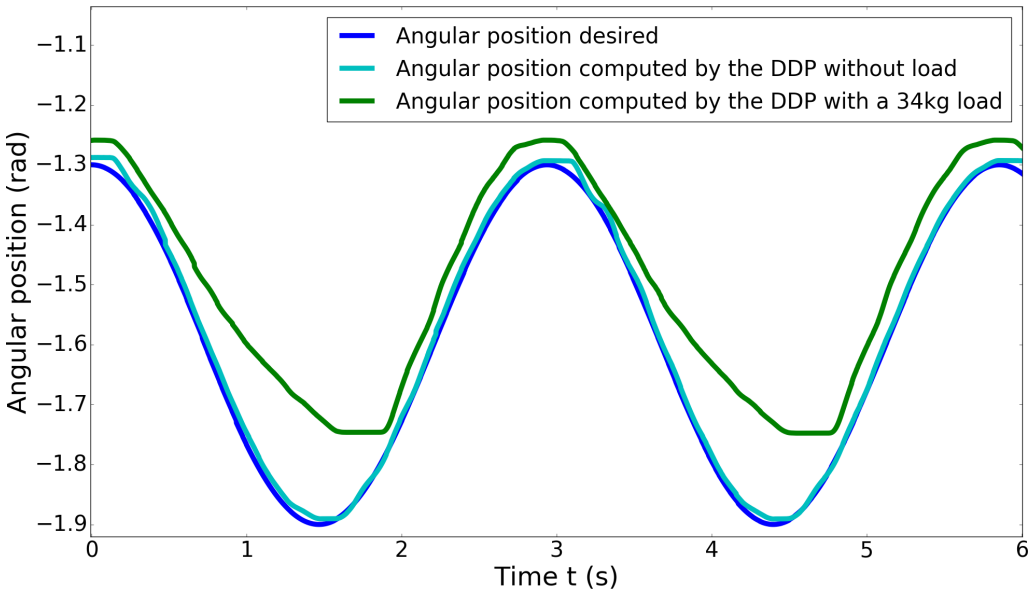

Figure 3.12: Experiments - State trajectories with and without a 34kg load.

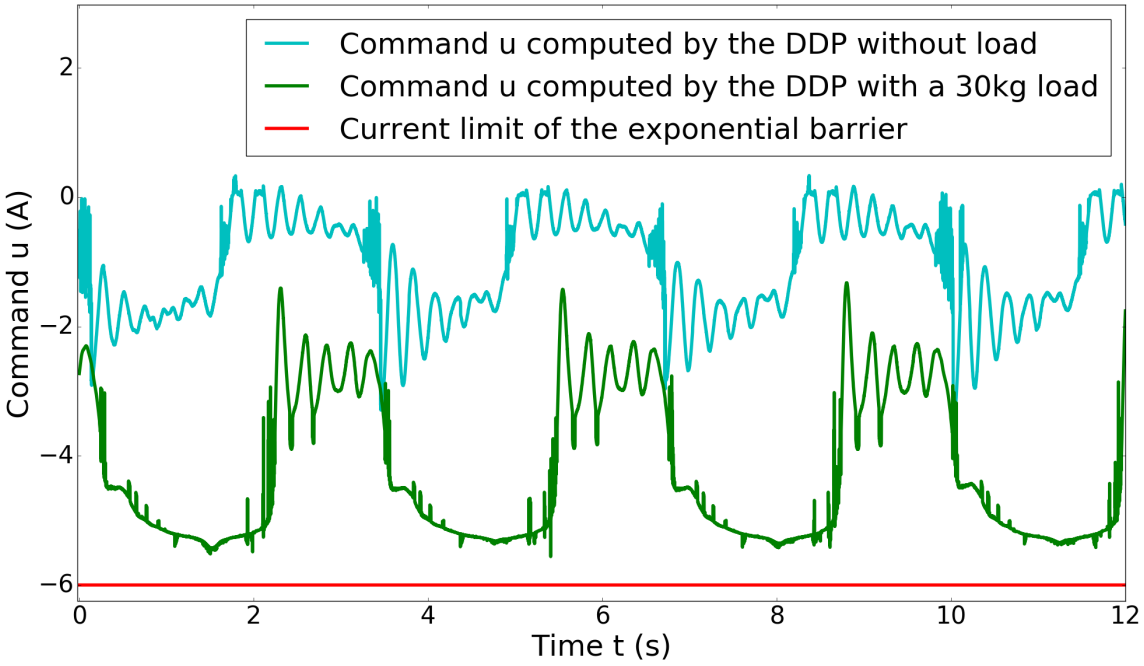

Figure 3.13: Experiments - Control trajectories with and without a 34kg load.

3.6 Conclusion

In this chapter we presented the actuator model, the identification and the control of the TALOS robot elbow joint. The identification results of the chain drive and of the inertial parameters at once proved to be accurate with low standard deviation and physical consistency of the parameters. Using the identified model and a iLQR approach to avoid reaching its limits, we have demonstrated that the robot is able to carry a load up to $34kg$ with a sinusoidal motion at low speed. As expected, it is not possible to use this algorithm in the main CPU as it takes $300\mu s$ for one actuator, nevertheless, we validated the efficiency of the solution. The extension of this work would be to identify all the actuators of the robot and to implement the solution over high-performance dedicated and embedded electronics board attached to each actuator. The work presented in this chapter details the publication Ramuzat et al. [22].

This work has focused on the identification and modeling of the rigid chain actuators of the robot and how to use these models to protect the system using a specific low level joint-controller. The following chapters must deal with the higher problematic of whole-body control which provides the desired joint-torque needed by the low-level controller. This whole-body controller must compute a command following reference trajectories given by a planning algorithm; regulating the errors coming from the robot model and environment.

Chapter 4

Whole Body Control

Contents

In this chapter are detailed the results obtained in the publications Ramuzat et al. [23] and Maroger et al. [24]. The second publication is a collaboration with the PhD Isabelle Maroger of the Gepetto Team. Not all this paper is detailed, my contribution on the whole-body controller and the simulations is the main part that is presented.

4.1 Introduction

One of the major issue involving the whole-body motion and control of humanoid robots is their locomotion. Bipedal locomotion of humanoid robots is still considered as a difficult problem, because of the complexity of the robots dynamics, the numerous constraints of the motion and the unknown environment. Indeed, to make a humanoid robot stand and walk it is necessary to find a solution involving balancing, contacts creation or removal and contacts stability while performing motions adapted to the environment. To achieve the locomotion of such complex systems, the problem can be decomposed in four stages (see Fig.2.7): guide planning, contacts sequence generation, centroidal trajectory optimization and whole-body control.

Some solutions for the planning stages have been presented in Section 2.4. Most of the trajectory planning methods use the centroidal dynamics to generate consistent behaviors for a legged robot. In addition, the concepts of Divergent Component of Motion (DCM) Englsberger et al. [45] and Capture Point (CP) Pratt et al. [43] associated to reduced dynamics models such as the Linear Inverted Pendulum (LIPM) Kajita et al. [47] allow to simplify the trajectory generation. Our work focuses on the last stage of the problem: the whole-body control. Most control architectures for legged locomotion are either *torque* or *position controlled*.

In the recent literature there is a growing number of implementation of torque control algorithms to solve locomotion problems Lee and Goswami [52], Koolen et al. [54], Englsberger et al. [97], Herzog et al. [153]. Indeed, due to the intrinsic compliance of the torque control formulation, it is more suitable for interactions with humans and for multi-contact problems where external interactions and several contact points are needed. However, the transition from the simulations to the real experiments are harder due to the necessity to take into account the model of the actuation chain of the robot Ramuzat et al. [22].

This chapter details the works realized to create whole-body controllers able to cope locomotion problems. Indeed, locomotion and manufacturing operations both consist in creating contacts and applying forces in a certain manner while balancing. The capabilities designed for the controllers to realize locomotion with the robot TALOS are the same one needed to achieve the force operations. Moreover, the Gepetto team core subject of study is the anthropomorphic motions, thus the team has already numerous solutions of complex locomotion trajectories. This allows us to quickly test our implementations on these problems. This chapter presents the different aspects and steps needed to create a complete solution. For instance, the DCM approach can be used in the control stage, for admittance control on the CoM Mesesan et al. [49], Caron et al. [50] to improve the stability of the robot (as described in Section 2.2.3.b). We successfully implement three controllers, two position controlled and one torque controlled.

We then investigate the differences and performances of our torque and position control schemes to choose the most appropriate one for the robot TALOS. We test the most promising controller on a specific application of human-robot collaboration for navigation. Finally, the results of the experiments performed on TALOS are presented.

The chapter is organized as follows: Sections 4.2 and 4.3 detail the formulations used for the position and torque QPs used by our whole-body controllers. Then, the

next sections explain the process implemented to benchmark the controllers: Section 4.4 details the three task-space whole-body control schemes to be compared. Section 4.4.2 presents the planning methodologies and the state estimator used for the simulations. Section 4.4.3 describes the energy criteria used for the controllers benchmarking. Section 4.4.4 presents the simulations results and Section 4.4.5 discusses them. In Section 4.5 is detailed the application of our torque controller in the scope of a human-robot co-navigation, this work was realized in collaboration with Isabelle Maroger. Finally, the last section describes the intermediate experiments realized on the real robot and the difficulties we faced, using the presented controllers.

4.2 Inverse Kinematics Quadratic Program

The first controller created during this thesis is a Hierarchical Quadratic Programming (HQP) (see Section 2.5.2.c). It solves a task-based inverse velocity kinematics described in Mansard et al. [16] and outputs a velocity command. It implements a strict hierarchy between the tasks and has been implemented using the package *SoT-Core* [20]. For the remaining of the manuscript, the inverse velocity kinematics is directly called inverse kinematics or IK.

4.2.1 Motion Task

In this controller, the task errors e to be reduced in the cost function are implemented as velocity-based tracking laws in the Lie group $SE(3)$. Having the robot coordinate vector q and its velocity v as control input, a task-function is a derivable function $x(q)$ whose space is named the task-space. And the task errors $e \in \mathbb{R}^l$ are expressed as in the Section 2.5.2.b:

$$\begin{array}{rcl} \dot{e}(q,t) & = & \dot{x}(q) - \dot{x}^*(t) \\ \dot{x}(q) & = & Jv \end{array} \tag{4.1}$$

with $x^*, \dot{x}^*$ the desired position and velocity of the task, l is the dimension of the task error (for instance, for the 3D acceleration error on the CoM $l = 3$), $J = [J_u\ J_a] = [\mathcal{T}(S,B)\ \frac{\partial e}{\partial q}] = [\mathcal{T}(S,B)\ \frac{\partial x}{\partial q}]$ the Jacobian according to the robot coordinates vector. J_u is the "Jacobian" of the under-actuated part: it is the transformation $\mathcal{T}$ between the task/error frame S and the base frame B (see Eq.2.6).

A first-order linear dynamics is imposed on these errors (see Section 2.5.2.b):

$$\begin{array}{rcl} \dot{e}(q,t) & = & K_P(x(q) \ominus x^*(q)) \\ \Leftrightarrow\ \dot{x}(q) & = & \dot{x}^*(t) + K_P(x(q) \ominus x^*(q)) \end{array} \tag{4.2}$$

with $\ominus$ the difference operator of Lie group.

4.2.2 Contact Constraints

In *SoT-Core*, the contact constraints are implemented as tasks with the higher priorities. The velocity of the task is null and the position is kept constant, i.e. for

a contact point x_c and its Jacobian J_c:

$$\begin{array}{rrcl} & x_c^i & = & x_c^{i+1} \\ & \dot{x}_c & = & 0 \\ \Leftrightarrow & J_c^T v & = & 0 \end{array} \tag{4.3}$$

4.2.3 Quadratic Programming Formulation

Using the similar formulation as Section 2.5.2.b and Section 2.5.2.c, with $g_i = \|w_i\|$ for each task $i \in N$, it is possible to formulate the QP formulation of our controller as follows:

$$\begin{array}{rrl} & g_i^* = \min\limits_{v,g_i} & g_i \\ \text{s.t.} & J_c^T v = 0 & \\ & g_j = g_j^* & \forall j < i \end{array} \tag{4.4}$$

with J_c the Jacobian associated to the c contact points $X_c = \{x_{c1}...x_{cc}\}$. Here we differentiate the contact tasks from the other one but they can be also put as function g_0 with the higher priority 0.

This strict hierarchy can be solved by using the orthogonal projection operator P on the null-space of the Jacobian J. The tasks with lower priorities are solved at best in the remaining space of the previous tasks solutions. With $J^\dagger$ the pseudo-inverse of J, one can write for N tasks such that $e_1 \prec e_2 \prec ... \prec e_N$:

$$\dot{q} = J_1^\dagger e_1 + \sum_{k=2}^{N} J_k^\dagger (\prod_{l=1}^{k-1} P_l) e_k \tag{4.5}$$

In *SoT-Core*, this QP is solved using the method presented in Mansard and Chaumette [78].

4.3 Inverse Dynamics Quadratic Program

The second and third controllers created during this thesis use a Weighted Quadratic Programming (WQP) called TSID [80]. It sums selected task functions in a general cost function using weights to define their priorities (see Section 2.5.2.c). It then optimizes this sum subject to several constraints. It has been successfully used on a HRP-2 robot to realize torque control in Prete et al. [21]. The controllers have been implemented in the package *SoT-Torque-Control* [79].

4.3.1 Motion Task

Most of the task-function errors of TSID are expressed as motion task errors e. They are implemented as acceleration-based tracking laws in the Lie group $SE(3)$. Having the robot coordinates vector q and the acceleration a as control input, a task-function is a second-order derivable function $x(q)$ whose space is named the task-space. And the motion task errors $e \in \mathbb{R}^l$ (with l the dimension of the error task for instance 3 for the CoM task) are expressed as in the Section 2.5.2.b:

$$\begin{array}{rcl} \ddot{e}(q,t) & = & \ddot{x}(q) - \ddot{x}^*(t) \\ \ddot{e}(q,t) & = & (Ja + \dot{J}v) - \ddot{x}^*(t) \end{array} \tag{4.6}$$

with $x^*, \ddot{x}^*$ the desired position and acceleration of the task, J the Jacobian according to the robot state vector as defined previously and in Eq.2.6.

The following dynamics is imposed on these errors (see Section 2.5.2.b):

$$\begin{array}{rrcl} & \ddot{e}(q,t) & = & K_P(x(q) \ominus x^*) + K_D(\dot{x}(q) - \dot{x}^*(t)) \\ \Leftrightarrow & \ddot{x}(q) & = & \ddot{x}^*(t) + K_P(x(q) \ominus x^*(t)) + \\ & & & K_D(\dot{x}(q) - \dot{x}^*(t)) \end{array} \tag{4.7}$$

with $\ominus$ the difference operator of Lie group. To simplify notations, in the following, we will use the usual difference operator instead of $\ominus$ and we will not write the state or time dependencies of the terms. Using Eq.4.6 and Eq.4.7 leads to the generic formulation for a motion task-function:

$$\underbrace{J}_{O}\ \underbrace{a}_{y} = \underbrace{\ddot{x}^* - \dot{J}v + K_P e + K_D \dot{e}}_{o} \tag{4.8}$$

4.3.2 Angular Momentum Task

The Angular Momentum (AM) dynamics is expressed in TSID using the equation defined in Section 2.2.4, recalled here:

$$\dot{k}_c = \dot{k}_c^* + K_{Pam}(k_c^* - k_c) \tag{4.9}$$

with $\dot{k}_c$ the commanded rates of change of the angular momentum and K_{Pam} the proportional gain for the AM task. The linear component follows the motion task defined in Eq.4.8, called CoM task.

4.3.3 Contact Constraints

The inverse dynamics of the robot is solved in rigid contact with the environment Herzog et al. [153], thus, these contacts constrain the motion. They are implemented in TSID as nonlinear functions, which are differentiated twice (with J_c the Jacobian at the contact point). The Eq.2.52 is recalled here for clarity:

$$\begin{array}{rcl} c(q) = 0 & \Leftrightarrow & \text{Contact point does not move} \\ J_c^T v = 0 & \Leftrightarrow & \text{Contact point velocity is zero} \\ J_c^T a + \dot{J}_c v = 0 & \Leftrightarrow & \text{Contact point acceleration is zero} \end{array} \tag{4.10}$$

The equation of the dynamics is formulated as an equality constraint with τ_{ext} replaced by the forces applied at each contact, and these forces are optimized (see Eq.4.14). In this chapter, we use the unilateral rectangular plane contact formulation of TSID. The contact is defined as a 6D motion task (Eq.4.8), an inequality constraint on the force bounds and a direct force task:

$$T^T f = f^* \tag{4.11}$$

with the force-generator matrix $T^T \in \mathbb{R}^{6\times 12}$ mapping the forces at the vertices of the contact surface, $f \in \mathbb{R}^{12}$ (four 3D forces), to a 6D resultant force at the contact point (center of the surface). $f^* \in \mathbb{R}^6$ the desired force for the contact. T creates the map between the two representations: $\tau_{contact} = J^T T^T f$. However in our formulation we do not use this direct force task, letting the QP optimize freely the forces at the contact, except for the Cartesian Force-Contact case.

4.3.4 Cartesian Force-Contact task

The context of my thesis is the validation of a solution for manufacturing operations. These operations, mainly drilling or tightening, need precision while applying a certain amount of force on a structure. To achieve these operations with our developed whole-body controllers we need to implement a new force task. The targeted application is the drilling of a metal plate, requiring the application of huge forces ($\sim$ 600N). Following the study of the state-of-the art realized on the force control for manipulations (see Section 2.7), and based on the formulation of the tasks in TSID, we choose to implement a parallel force/position control. This task is composed of two parts: the force application and the position control of the contact. Thus, it combines two tasks, a motion task defined in Eq.4.8 for the control of the position, and a force task for the application of the force.

The force task is defined as a Proportional Integral with a feed-forward (PI+) and an anti-windup, inspired by Shahriari et al. [113]:

$$\begin{aligned} e_f &= F^* - F_{ext} \\ \underbrace{T_F^T}_{O}\underbrace{F_F}_{y} &= \underbrace{F^* + P_F e_f + I_F \int_0^t e_f ds}_{o} \end{aligned} \tag{4.12}$$

with $F^*, F_{ext} \in \mathbb{R}^6$, respectively the desired and measured or estimated external wrenches and $P_F, I_F \in \mathbb{R}^{6\times6}$ the task gains. T_F is the force-generator matrix defined as in the previous paragraph and F_F is the constrained wrench.

In TSID, as explained in Eq.2.52,4.11, the forces in the equation of the dynamics are expressed at the contacts. Thus, the force task is linked to its appropriate contact task (see previous paragraph, Section 4.3.4). In the library, the task is called *task-contact-force*, it implements the force task of Eq.4.12 and the position control part of the task is done by the 6D motion task of the associated contact (Eq.4.8).

Thus, we implemented a parallel force/position control which also respects the coulomb constraints as the position control is done by the contact constraint of the previous section (Section 4.3.3). Considering that our system has c contacts and the Cartesian Force-Contact task, we can replace τ_{ext} by:

$$\tau_{ext} = \sum_{k=1}^{c} J_k^T T_k^T f_k - J_F^T T_F^T F_F \tag{4.13}$$

which is in fact the $J_c^T F$ term defined in Eq.2.4.

4.3.5 Quadratic Programming Formulation

TSID sums all the task functions $g_i(y)$ using user-defined scalar weights λ_i, $\forall i \in \mathcal{N}$ tasks, subject to constraints such as the rigid contacts and the dynamics. It leads to the same formulation defined in Eq.2.57 replacing ω_i with the tasks functions (Eq.4.8):

$$
\begin{aligned}
& g_i(y) = \|O_i y - o_i\|^2 \\
\min_{y=[a,f]} \quad & \sum_{i=0}^{\mathcal{N}} \lambda_i \, g_i \\
\text{s.t.} \quad & \begin{bmatrix} J_c & 0 & 0 \\ M & -J_c^T & -N^T \end{bmatrix} \begin{bmatrix} a \\ f \\ \tau \end{bmatrix} = \begin{bmatrix} -\dot{J}_c \, v \\ -h \end{bmatrix} \\
& \tau_{min} \leq \tau \leq \tau_{max} \\
& q_j^{min} \leq q_j \leq q_j^{max} \\
& \dot{q}_j^{min} \leq \dot{q}_j \leq \dot{q}_j^{max} \\
& f_{min} \leq f \leq f_{max}
\end{aligned}
\tag{4.14}
$$

Where the free variables are the acceleration a and the force F. The QP is formulated to solve the acceleration and the forces and to then retrieve the torque using Eq.2.4. It has only two strict layers: the constraint layer (priority 0) and the cost layer (priority I).

The solver used in TSID is eiquadprod [198], it implements the algorithm of Goldfarb and Idnani Goldfarb and Idnani [199] for the solution of a convex QP problem by means of a dual method.

For $x = [a, F]$, the problem is of the form :

$$
\begin{aligned}
\min_x \quad & \tfrac{1}{2} x^T G x + g_0 x \\
\text{s.t.} \quad & CE^T x + ce_0 = 0 \\
& CI^T x + ci_0 \geq 0
\end{aligned}
\tag{4.15}
$$

with G the symmetric Hessian of the Cholesky factorization of the cost function, g_0 the Lagrange multipliers of the cost function, CE the generalized matrix of the equality constraints and ce_0 their associated vectors and similarly CI the generalized matrix of the inequality constraints and ci_0 their associated vectors.

4.4 Comparison of the Control Schemes

This section focus on the implementation and comparison of three real-time whole-body controllers using the task-function approach Samson et al. [37], Escande et al. [38]. The objectives to be performed by the robot are expressed in their respective task spaces, using reference trajectories given by the planning. Complex motions combine several nonlinear tasks and constraints. Quadratic Programming problems (QP) are instantaneous optimization techniques used to solve such nonlinear problems, employing the whole-body kinematics or dynamics of the robot. In this section two types of QP are compared, a Hierarchical QP which imposes a strict hierarchy between the tasks Hoffman et al. [88], Henze et al. [89], Herzog et al.

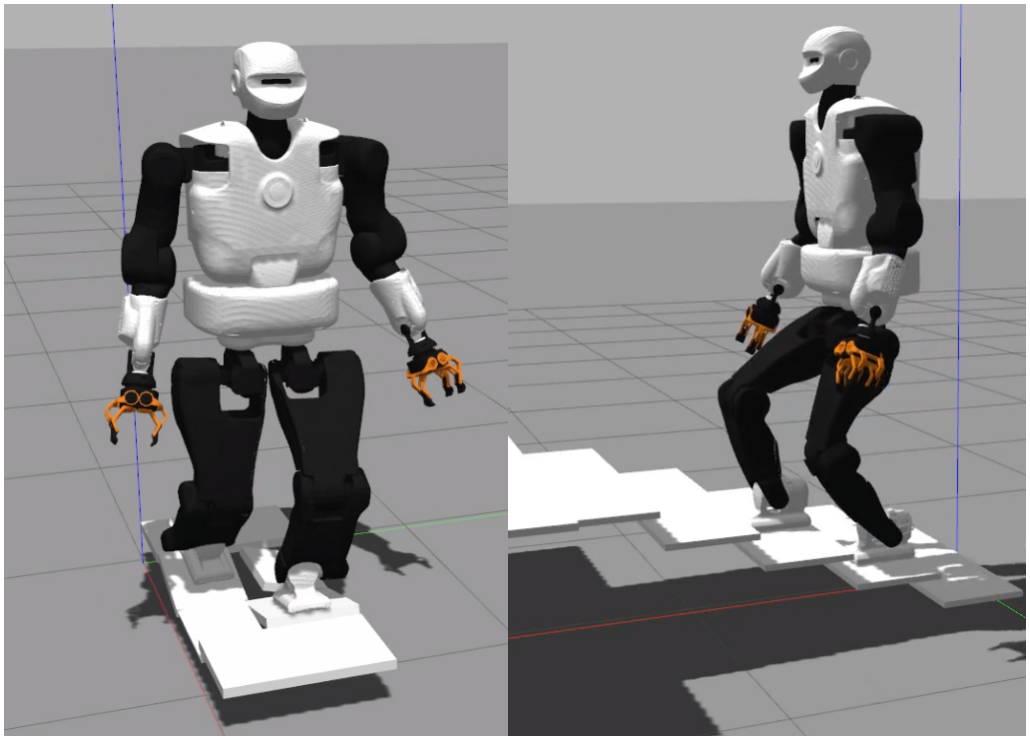

Figure 4.1: Walking on Uneven Terrain and Climbing Stairs.

[153], and a weighted QP which sets weights to prioritise the tasks Koolen et al. [54], Bouyarmane et al. [84], Cisneros et al. [103].

This section intends to follow the benchmarking of humanoid robots control architectures Romualdi et al. [200]. It contributes toward the implementation and comparison of three whole-body control schemes: two using position control associated with DCM and CoM admittance controls and one using torque control. The first one is based on a Hierarchical QP using Inverse Kinematics (denoted IK in the section), the second and the third one use a weighted QP with Inverse Dynamics and an Angular Momentum (AM) regularization task (denoted respectively TSID position and TSID torque). They are evaluated in Gazebo simulations on three locomotion problems: walking on flat terrain, walking on uneven terrain and climbing stairs, on the criteria of trajectory tracking, energy consumption, passivity and computational cost. In particular, the proposed torque control implementation allows the TALOS robot to reach a 0.6m/s walk on flat floor, which is the highest walking velocity achieved on the TALOS robot.

4.4.1 Control Schemes

4.4.1.a Lexicographical Quadratic Programming

The first controller used is a Hierarchical Quadratic Programming (HQP) task-based inverse kinematics described in Mansard et al. [16] (see Sectin 4.2).

Inverse Kinematics QP: IK - This control scheme is based on a DCM controller (Eq.2.30), a CoM admittance controller (Eq.2.31) and a Lexicographic QP solving the inverse kinematics of the robot. The authors have implemented this scheme in

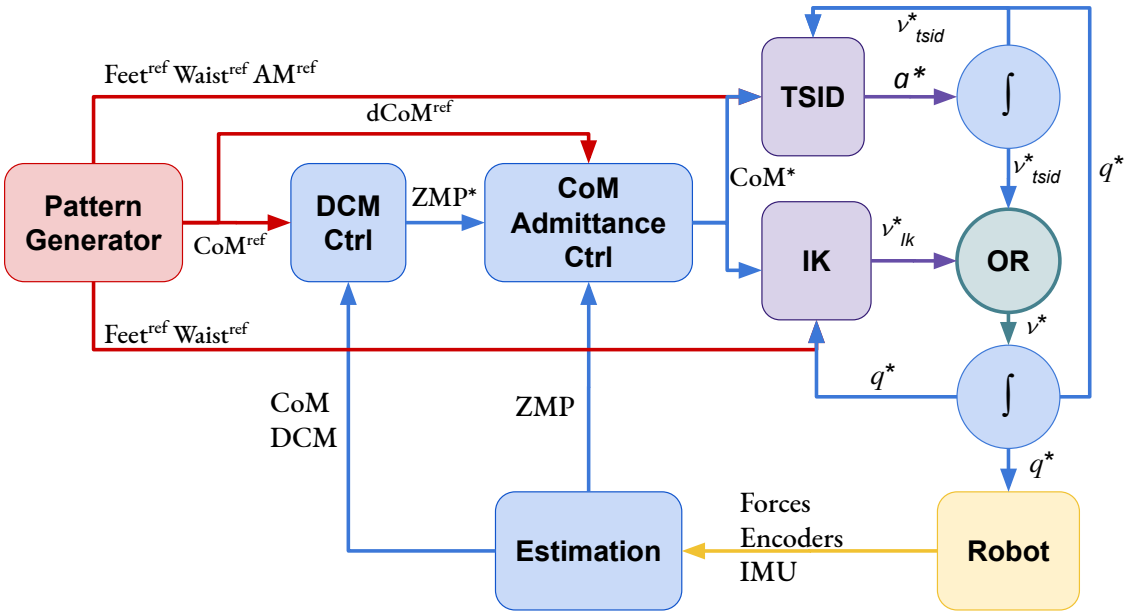

Figure 4.2: Position control schemes: IK and TSID. The *OR* block is used to activate only one controller at a time.

Tasks	Priority
Feet tracking	0
CoM height tracking	I
CoM lateral-sagittal tracking	II
Waist orientation	III
Posture regularization in half-sitting	IV

Table 4.1: Set of tasks used for the IK control scheme.

an open-source package [201], based on the QP in Mansard et al. [16], adding the DCM and CoM admittance controllers.

The tasks used during the simulations are defined in Table.4.1 (the priority 0 is the highest one). The respective task gains are defined in Table 4.3. The weights and gains have been chosen through trials and errors with an a priori heuristic.

This control scheme is presented in the Fig.4.2, the *IK* block in purple represents the Lexicographic QP solving the inverse kinematics of the robot. The Eq.2.30 is implemented in the *DCM Ctrl* blue block and the Eq.2.31 in the *CoM Admittance Ctrl* one. A complete scheme of the framework is presented in Appendix 3, in the Fig.6.5.

4.4.1.b Task Space Inverse Dynamics (TSID)

In this section are described the two schemes based on TSID (see Section 4.3) compared on locomotion challenges on the robot TALOS.

Inverse Dynamics WQP: TSID Position - This control scheme is based on a DCM controller (Eq.2.30), a CoM admittance controller (Eq.2.31) and a WQP solving the inverse dynamics of the robot. Compared to the previous controller, this one

Tasks	Priority	Weight
Feet tracking	0	-
Feet contacts	0	-
CoM height tracking	I	10^3
CoM lateral-sagittal tracking	I	10^3
Waist orientation	I	1
Posture regularization in half-sitting	I	0.1
AM velocity-acceleration regularization	I	2×10^{-2}

Table 4.2: Set of tasks used for the TSID position and torque control schemes.

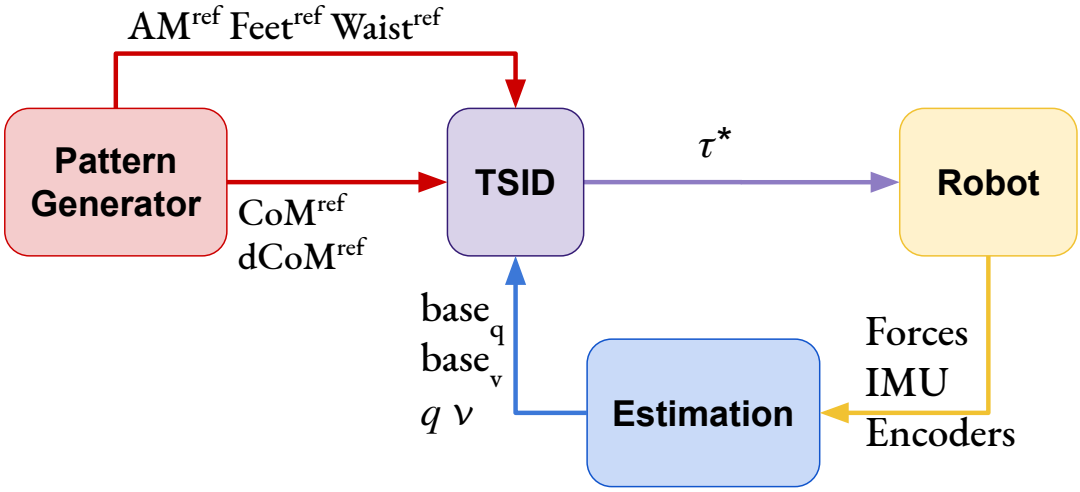

Figure 4.3: TSID torque control scheme.

implements an AM task, which regulates the angular momentum to 0, using the formulation of Eq.2.32. The authors have implemented this controller using the TSID [80] library in the same package than the controller *TSID Torque*, with the DCM and CoM admittance controllers.

The tasks considered during the simulations are defined in Table.4.2. The priority 0 corresponds to a constraint and the priority I to a task in the cost function. The respective task gains are defined in Table 4.3. The weights and gains have been chosen through trials and errors with an apriori heuristic.

This control scheme is presented in the Fig.4.2, the *TSID* block in purple represents the WQP solving the inverse dynamics of the robot. The Eq.2.30 is implemented in the *DCM Ctrl* blue block and the Eq.2.31 in the *CoM Admittance Ctrl* one. A complete scheme of the framework is presented in Appendix 3, in the Fig.6.5.

Inverse Dynamics WQP: TSID Torque - This control scheme is based on a WQP solving the inverse dynamics of the robot (with an AM regularization task, using the formulation of Eq.2.32), as shown in Fig. 4.3. From the desired acceleration computed by the QP, TSID retrieves the associated torque by using the robot equation of the dynamics. The authors have implemented this controller using the TSID [80] library in the open-source package [202].

The tasks considered in the simulations are the same as *TSID position*, with

different gains (see Table 4.3). A complete scheme of the framework is presented in Appendix 3, in the Fig.6.5.

4.4.1.c Remark on the state feedback

For position control, it is needed to integrate the result of the QP (one time for *IK* and two times for *TSID position*, see Fig.4.2) to obtain the desired command. To avoid instabilities, the control loop of both QP use these integrated values in the next iteration instead of the measured ones. The measured position and velocity of the robot are only used to compute the CoM, DCM and ZMP for the admittance control in the position schemes. In contrary, the torque control scheme uses the measured values at each iteration of the QP (see Fig.4.3) and in particular the position and velocity of the robot base (or free-flyer).

4.4.2 Locomotion Planning and Estimator

In this section the two locomotion planning methods presented in Section 2.4 have been used. First, the WPG has computed the trajectories for the walk on flat floor simulations (20cm and 60cm), with pre-defined set of foot steps. The WPG is plugged to the whole body controller using the dynamic-graph package in order to get the next reference point of the trajectories in real-time. Secondly, the *multicontact-locomotion-planning* is used to obtain the trajectories for the platforms and stairs simulations. Here the references are computed offline and then a file containing the trajectories is read during the simulation.

Finally, for every simulations, the estimation of the base (free-flyer) of the robot has been realized using the approach presented in Flayols et al. [29]. The free-flyer information are estimated using robot joints configurations and velocities, the IMU and the force sensors of the robot. It allows the whole-body controller to have access to the whole state vector, including the position and velocity of the base, necessary to compute the CoM on an under-actuated system.

4.4.3 Energetic Comparison Criteria

4.4.3.a Energy cost

Based on Torricelli et al. [203], a relevant criterion to compare the energy consumption of the control schemes is the cost of transport. It can be computed as the energetic cost of transport C_{et} using the whole mechanical work of the actuation system E_m or as the mechanical cost of transport C_{mt} using only the positive one E_{m+}.

$$C_{et} = \frac{E_m}{mgD} \qquad C_{mt} = \frac{E_{m+}}{mgD} \qquad E_m = \int_0^T \sum_{i=0}^{N} |\tau_i(t)\,\omega_i(t)|dt \tag{4.16}$$

with m the mass of the system, g the gravity constant, D the distance traveled by the system and τ_i, ω_i the respective torque and velocity of each robot joint for all (N) joints.

Tasks Gains	IK (20cm|stairs)	TSID position (20cm|stairs)	TSID torque (20-60cm|stairs)
$K_{\mathrm{P}com}$	100	1000	20|12
$K_{\mathrm{D}com}$	-	300	3
$K_{\mathrm{P}comH}$	100	1000	-
$K_{\mathrm{D}comH}$	-	300	-
$K_{\mathrm{P}waist}$	300	100	100
$K_{\mathrm{D}waist}$	-	20	20
$K_{\mathrm{P}contacts}$	1000	30	30-100|30
$K_{\mathrm{D}contacts}$	-	11	11-0|11
$K_{\mathrm{P}feet}$	1000	2000	1200|500
$K_{\mathrm{D}feet}$	-	20	12
$K_{\mathrm{P}am}$	-	10	10
$K_{\mathrm{P}posture}$	100	see below	see below
$K_{\mathrm{D}posture}$	-	$2\sqrt{K_{\mathrm{P}posture}}$	$2\sqrt{K_{\mathrm{P}posture}}$
$K_{\mathrm{P}comAdm}$	15|45	12	-
$K_{\mathrm{P}dcm}$	8|25	8	-
$K_{\mathrm{I}dcm}$	1	1	-
$K_{\mathrm{Z}dcm}$	1	1	-

TSID Gains	Legs	Torso
$K_{\mathrm{P}posture}$	[10, 5, 5, 1, 10, 10]	[100, 100]
	Arms	Head
$K_{\mathrm{P}posture}$	[50, 10, 10, 10, 50, 10, 10, 10]	[100, 100]

Table 4.3: Tasks gains of the control schemes. *tilted platforms* and *stairs* simulations use the same gains.

4.4.3.b Passivity Gait Measure

Another interesting energetic criterion is the ability to minimize joint torques to increase the passivity of the walk Torricelli et al. [203]. The Passivity Gait Measure (PGM)Mummolo and Kim [204] quantifies the passivity of a biped walking motion:

$$PGM = 1 - \frac{RMS(\tau_{sa})}{RMS(\tau_{tot})} \tag{4.17}$$

$$RMS(\tau_{tot}) = \sqrt{\frac{\int_0^T \left[\sum_{i=0}^{N} \tau_i(t)^2\right] dt}{T}} \tag{4.18}$$

where RMS is the Root Mean Square along the period of time T, τ_{sa} stands for the torque on the stance ankle joint and τ_{tot} for the torque on all robot joints.

Control Scheme	Axis	Average	Standard deviation	Peaks
IK	x-axis	0.019m	0.022m	0.131m
	y-axis	0.022m	0.026m	0.150m
TSID	x-axis	0.028m	0.025m	0.142m
position	y-axis	0.025m	0.027m	0.138m
TSID	x-axis	0.026m	0.021m	0.078m
torque	y-axis	0.011m	0.014m	0.078m

Table 4.4: ZMP error of the 20 cm step walk simulation.

4.4.4 Simulation Results

The simulations presented in this section have been made using Gazebo. A video illustrating the simulations is available at the following link: https://peertube.laas.fr/videos/watch/4b5d3a5b-2355-47a0-8197-f41ed4f885c6. The chosen simulations are walking on flat or uneven terrains and stair climbing. Based on Torricelli et al. [203], they cover different aspects of locomotion skills for a stationary environment with and without unexpected disturbances.

4.4.4.a Straight walk of 20 cm steps

In the simulation, the robot executes 6 steps forward at 0.2m/s and a final step (traveled distance of 1.2m). The time distribution is 0.9s for single support phase and 0.115s for double support phase (leading to steps of approx. 0.20m). The controllers have also been successfully tested on a faster walk with single/double support time of 0.711/0.089s. The Fig. 4.4 presents a comparison of the three control schemes on their estimated ZMP, on the sagittal (x-axis, top curves on the figure) and lateral (y-axis, bottom curves) planes only, because the desired height of the CoM is constant. Similarly Fig. 4.5 provides the results of the DCM estimation and tracking. Fig. 4.6 shows the forces applied on the ground along the z-axis on the left foot. The tracking of the CoM and the feet are accurately followed by the three controllers (tracking error lesser than 1cm).

The two position controllers achieve similar results, tracking correctly the ZMP reference of Eq. 2.30 and thus the DCM, with an average error of 2cm (see Table 4.4). Noticeably, the torque control presents a ZMP which is close to the position control results in Fig. 4.4 even though there is no explicit control on the ZMP nor the DCM. In the Tables presenting the error on the ZMP, for the torque scheme, the estimated ZMP is compared to the desired ZMP (from the planning). In particular, in the lateral plane, the error is quite low, $1cm$ in average.

The Fig. 4.6 illustrates the ground impacts problem in position control compared to the better foot landing observed in torque control. Indeed, each time the left foot comes into contact with the ground (1.5s, 3.5s,...), the *IK* and *TSID position* schemes show peaks in the foot force ($\sim$ 400N) which are avoided in *TSID torque*. This explains also the peaks in the ZMP errors (around 15cm) because during an impact the foot bounces on the ground.

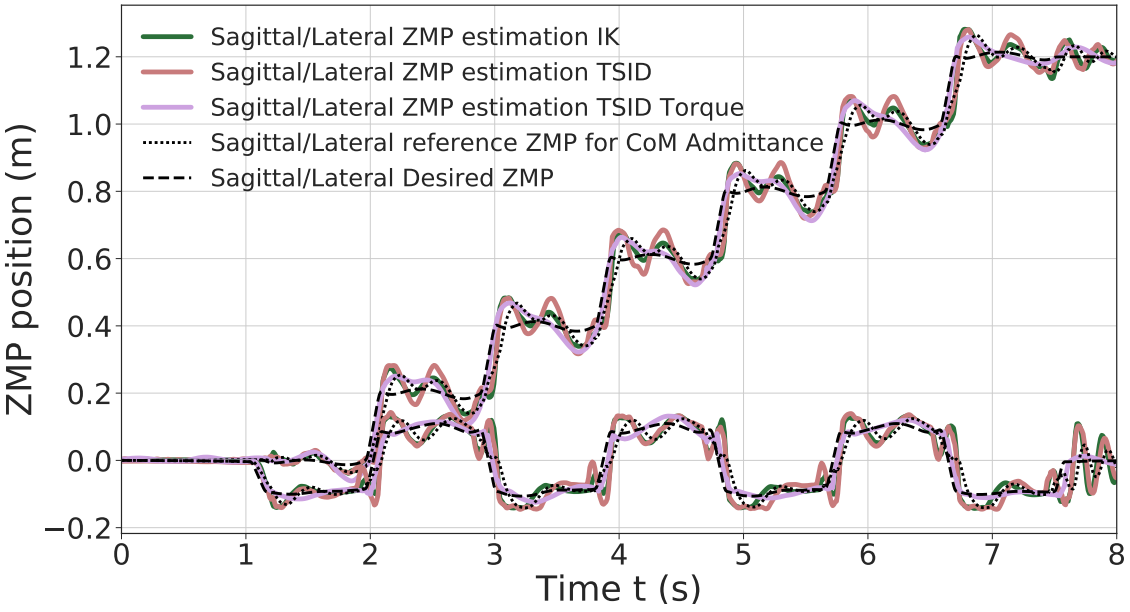

Figure 4.4: ZMP estimation of the 20 cm step walk.

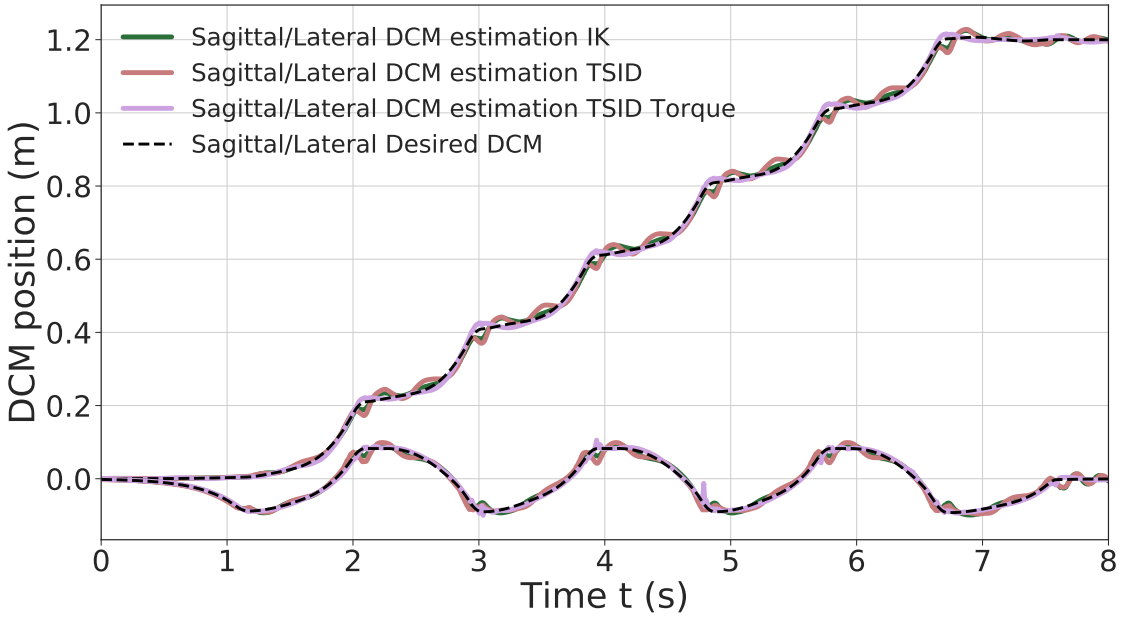

Figure 4.5: DCM estimation of the 20 cm step walk.

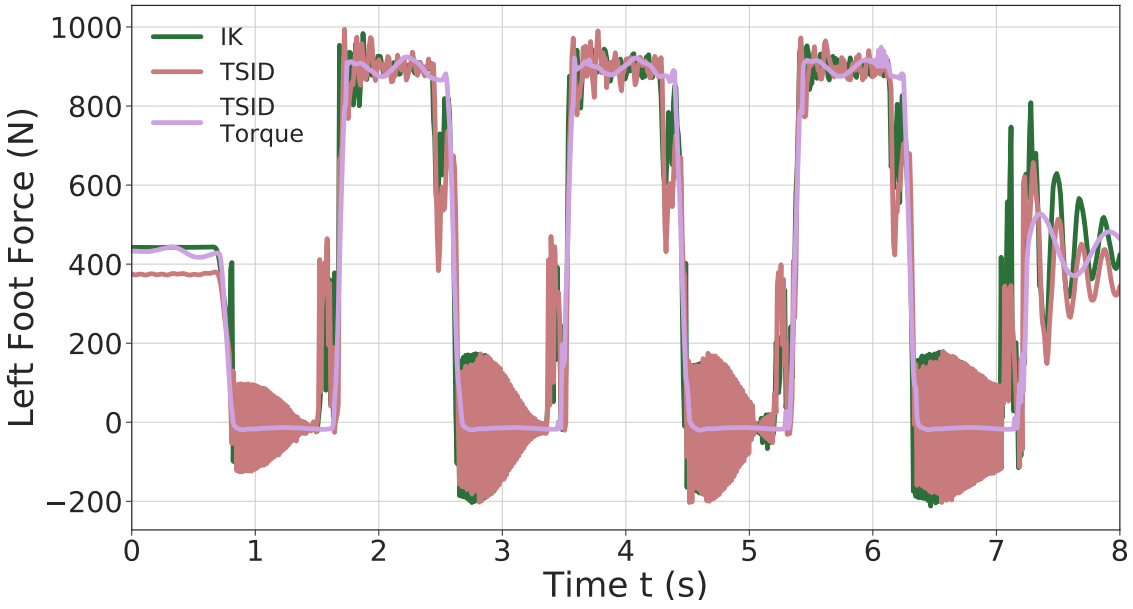

Figure 4.6: Z-axis left foot force of the 20 cm step walk.

The force oscillations of the *IK* and *TSID position* controllers when the foot is in the air are due to the high control gains on the ankle (PID gains of the low-level position control in Gazebo), it is mainly noises.

4.4.4.b Straight walk of 60 cm steps in torque control

In Mesesan et al. [49] the humanoid robot TORO successfully performed a walk on flat terrain with a step length of 55cm (single/double support time of 1.1/0.4s). In the following simulation, the torque controller is pushed to its limits to show its capability to achieve a similar result. The robot TALOS executes 6 steps forward of 0.6m/s and a final one to go back to the initial position. The time distribution used is of 0.9s for single support phase and 0.115s for double support phase (leading to steps of approx. 60cm).

Figure 4.7 presents the results obtained on the tracking of the feet and the CoM (see Table 4.5); the ZMP and DCM estimations. The feet tracks well the desired trajectories along the y-axis (maximum error of 6mm) however, along the x-axis, they show some delay (maximum error of 6cm). Thus, it induces greater tracking errors on the x-axis for the CoM (peaks of 5cm along the x-axis and 1.5cm along the y-axis). One can notice that the DCM and ZMP along the x-axis are more stable, whereas along the y-axis they present large oscillations (which are caused by the feet impacts on the ground when landing).

In Fig. 4.8, the AM behavior is shown along the three axes. The AM task minimizes the momentum to zero. The x and y momentum components are the most solicited, leading to the inclination of the torso forward and backward and to important moves of the arms to compensate the delay of the CoM and succeed the 60cm steps. The authors observed that without this AM task, the walk cannot be achieved.

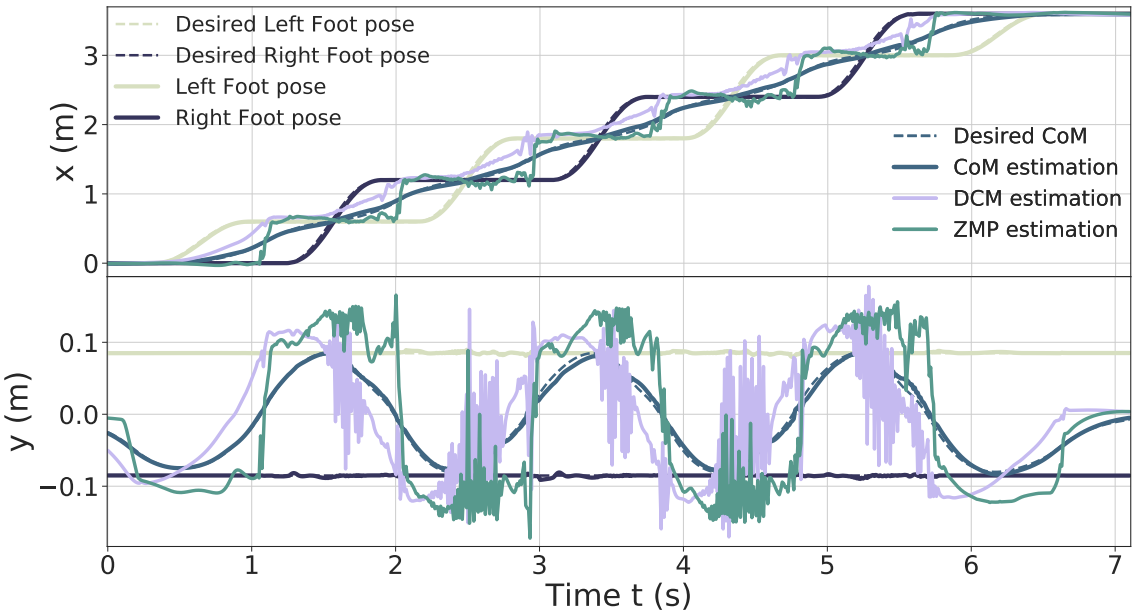

Figure 4.7: Feet, CoM, DCM and ZMP of the 60 cm step walk.

	Axis	Average	Standard deviation	Peaks
CoM	x-axis	0.018m	0.013m	0.050m
	y-axis	0.004m	0.003m	0.015m
Left Foot	x-axis	0.014m	0.013m	0.063m
	y-axis	0.001m	0.001m	0.005m
Right Foot	x-axis	0.016m	0.016m	0.063m
	y-axis	0.001m	0.001m	0.006m

Table 4.5: CoM and Feet error of the 60 cm step walk.

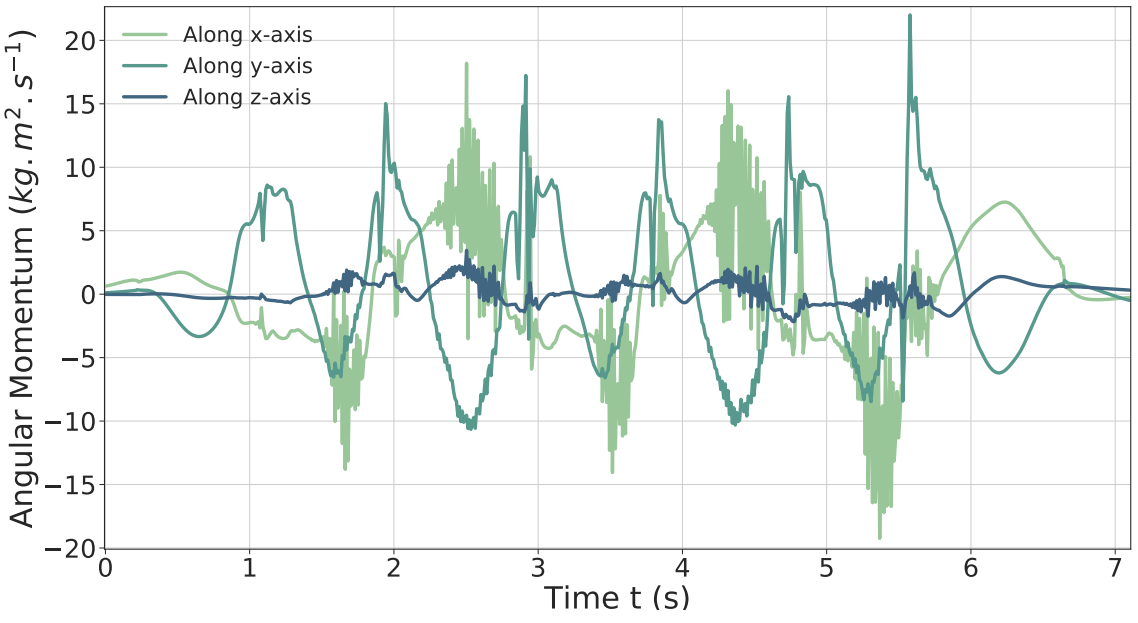

Figure 4.8: AM behaviour during the 60 cm step walk in torque.

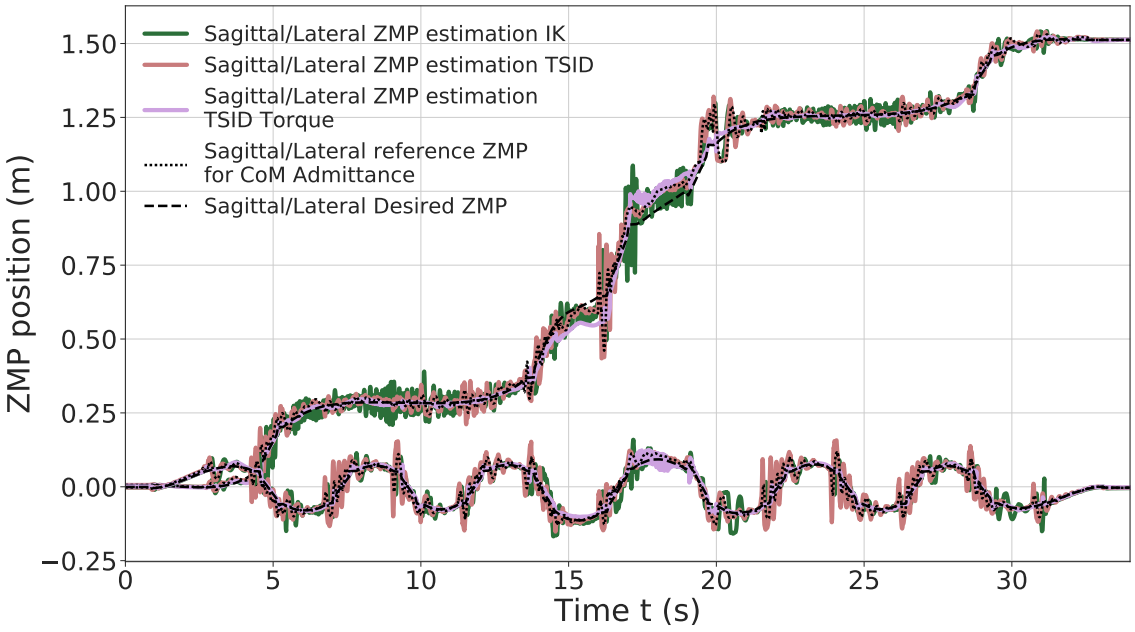

Figure 4.9: ZMP estimation of the *tilted platforms* simulation.

4.4.4.c Walk on the tilted platforms: Uneven terrain

In this third simulation, the robot walks on tilted platforms which represent uneven terrain (Fig. 4.1). This walk is achieved using the *multicontact-locomotion-planning* trajectories (see Section 2.4.2). The framework ensures the stability of the controllers on non-flat terrain when the feet are tilted.

Figure 4.9 illustrates the tracking performance of the controllers. The ones in position present the largest oscillations as *TSID torque* is the most stable (see Table 4.6). Both the *IK* and the torque control show oscillations at $t \approx 18$s; it corresponds to the worst case where the robot has its two feet tilted to keep its balance on two opposite platforms leading to small slippages of the feet (this behavior can be observed in the linked video). These oscillations are larger in the case of the *IK* scheme. Figure 4.9 illustrates the tracking performance of the controllers. The ones in position present the largest oscillations as *TSID torque* is the most stable (see Table 4.6). Both the *IK* and the torque control show oscillations at $t \approx 18$s; it corresponds to the worst case where the robot has its two feet tilted to keep its balance on two opposite platforms leading to small slippages of the feet (this behavior can be observed in the linked video). These oscillations are larger in the case of the *IK* scheme.

Finally the same result of the *20cm* walk on the contact forces is obtained in this simulation, as described by Fig.4.10, there are impacts on the platforms in position control. Due to the high gains on the DCM to avoid the slippage of the robot, the IK control presents bigger peaks of force. Increasing the gains on the feet only generates more instability, but raising the ones on the DCM and admittance control lessen the oscillations (at the cost of a more rigid behavior).

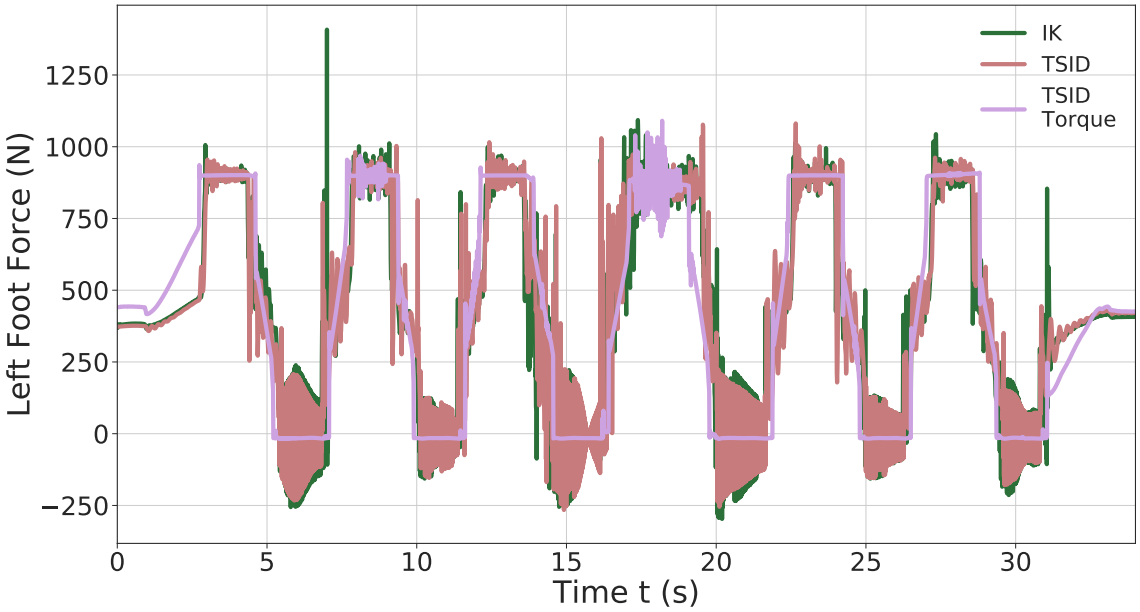

Figure 4.10: Z-axis left foot force of the *tilted platforms* simulation.

Control Scheme	Axis	Average	Standard deviation	Peaks
IK	x-axis	0.021m	0.024m	0.278m
	y-axis	0.016m	0.018m	0.118m
TSID	x-axis	0.012m	0.017m	0.197m
position	y-axis	0.015m	0.019m	0.127m
TSID	x-axis	0.013m	0.021m	0.107m
torque	y-axis	0.005m	0.006m	0.058m

Table 4.6: ZMP error of the *tilted platforms* simulation.

4.4.4.d Climbing Stairs

In the last simulation the robot is climbing 6 stairs of 10cm height and 30cm long (see Fig. 4.1). The trajectories are planned with the *multicontact-locomotion-planning*. Fig. 4.11 shows the ZMP evolution of each controllers, where the result is similar to the *uneven terrain* simulation. The *TSID torque* scheme behave significantly better than the others, with a ZMP matching the one planned (errors lesser than 1cm, see Table 4.7). Noticeably, the *IK* scheme presents higher oscillations at the end of the move in the lateral plane. The robot ends displaced on the right compared to the desired trajectories, due to slippages of the feet when it finishes to climb a stair (shown in the linked video).

4.4.4.e Energy cost and Passivity Gait Measure

The results obtained for the cost of transport of the four simulations are presented in the Table 4.8, depending on the control scheme. The results obtained for iCub in Romualdi et al. [200] are also presented for comparison (computed using Eq.4.16), as the human ones. The lower the energy consumption is, the better, and similarly,

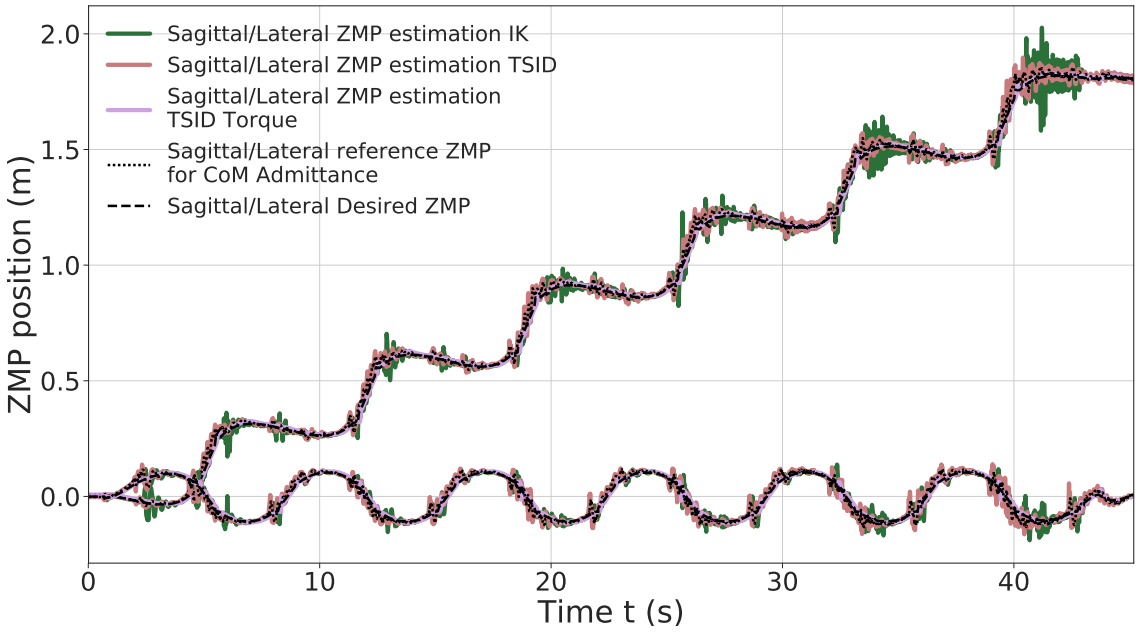

Figure 4.11: ZMP estimation of *stairs* climbing.

Control Scheme	Axis	Average	Standard deviation	Peaks
IK	x-axis	0.022m	0.026m	0.257m
	y-axis	0.015m	0.017m	0.151m
TSID	x-axis	0.009m	0.013m	0.151m
position	y-axis	0.012	0.015m	0.119m
TSID	x-axis	0.008m	0.006	0.049m
torque	y-axis	0.006m	0.005m	0.047m

Table 4.7: ZMP error of the *stairs* simulation.

getting closer to the human cost of transport is an improvement.

Compared to the results obtained on iCub, the control in torque has a similar cost for the *20*cm steps simulation. However, the cost of the position controllers presented in this section is higher, because of their higher gains. The human efficiency is closer to the torque control, walking with a C_{et} around $0.2J/kg/m$ Collins et al. [205]. Noticeably, the energy costs in torque for the *tilted platforms* and *stairs* trajectories are still less important than the simpler walk in position; the C_{mt} never exceeds 1, even for the *60 cm* walk. Overall, the controller *TSID position* consumes less energy than the *IK*.

The Passivity Gait Measure comparison of the different simulations is reported in Table 4.9 for three gait stages: Single Support (Single S. corresponding to the stance ankle), Double Support (Double S.) and Flying Foot (Flying F. where the foot has no contact with the ground). The human results is given as an indicator Mummolo and Kim [204], the robot behavior is expected to be similar during double support and flying foot phase where the ankle should be passive.

The results of the position control schemes show a behavior which is the opposite of the human one. The passivity of the ankle is higher during the stance phase

Control Scheme	Simulation	E_m [J]	E_{m+} [J]	C_{et} [J/kg/m]	C_{mt} [J/kg/m]
Human	-	-	-	0.2	0.05
iCub					
position	20cm	-	-	-	0,49
torque	20cm	-	-	-	0.26
	20cm	1983.9	1359.3	1.68	1.15
IK	platforms	5418.7	3769.2	3.7	2.6
	stairs	7249.5	2145.3	4.1	1.2
TSID	20cm	2324.5	764.1	1.97	0.65
position	platforms	5377.5	1413.6	3.6	2.0
	stairs	6812.6	2059.6	3.8	1.2
	20cm	521.8	259.3	0.44	0.22
TSID	60cm	3147.2	1583.8	0.89	0.45
torque	platforms	1378.6	668.5	0.93	0.45
	stairs	1861.1	1205.5	1.1	0.68

Table 4.8: Results of the specific cost of transport.

	Simulation	Double S.	Single S.	Flying F.
Human	50cm	1.0	0.6	~ 1.0
	20cm	0.35	0.89	0.24
IK	platforms	0.27	0.85	0.31
	stairs	0.46	0.86	0.36
TSID	20cm	0.37	0.74	0.37
position	platforms	0.27	0.86	0.30
	stairs	0.55	0.86	0.34
	20cm	0.93	0.87	1.0
TSID	60cm	0.87	0.79	1.0
torque	platforms	0.87	0.8	0.91
	stairs	0.97	0.89	1.0

Table 4.9: Results of the PGM on three gait stages.

Control Scheme	Simulation	20cm (60cm)	Platforms	Stairs
IK	Average	0.5ms	0.7ms	0.6ms
	Peaks	2ms	4ms	4ms
TSID	Average	1.2ms	1.2ms	1.2ms
position	Peaks	4.5ms	4.3ms	4.2ms
TSID	Average	1ms (1.4ms)	1.2ms	1.1ms
torque	Peaks	2.8ms (6ms)	5ms	5.5ms

Table 4.10: Comparison of the execution time.

because of the control of the ZMP which minimizes the ankle torque. And it is weaker during the double support and flying phases, due to the high PID gains of the low-level position control.

The control scheme in torque shows much more passive behavior (except on the stance foot), with a completely passive foot during the flying phase. During the double support phase, the ankle is almost passive ($PGM \sim 0.9$) which is close to the human result. These results are better than the one expected in Mummolo and Kim [204], where the torque controlled robot has a higher control on its stance ankle ($PGM = 0.2$).

Finally, on the *uneven terrain*, the double support phase corresponds to the worst case where the robot has its two feet tilted to keep its balance on two opposite platforms. This leads to a greater actuation than on flat floor (decreasing the passivity). Similarly, the stance phase corresponds to the left support phase on the final platform (highest slope), also leading to a bigger actuation of the ankle.

4.4.4.f Execution time of the control schemes

The computational time obtained during the execution of one control loop of the three schemes are presented in Table 4.10, according to the simulations.

The computational time of the *IK* is better due to the computational efficiency of the null space projectors of the tasks. Exploiting this specific structure allows it to keep its control frequency higher than 1kHz in average with 4 hierarchy levels. In *TSID* this method can only be used once because it is composed of two strict layers: the constraints and the cost.

4.4.5 Discussion

For the PGM results of the position schemes, the authors believe that adding an admittance control Caron et al. [50] on the ankle orientation may improve the results. If added, one can expect an increase of the actuation of the ankle during the stance phase, leading to a smaller PGM value.

In general the *IK* scheme presents higher oscillations and slippages when adding contacts. The authors think that this issue is mitigated in *TSID position* because it separates the feet task into a contact task and a tracking task. It allows to have different gains depending on the context (contact or not), indeed, the TSID schemes have higher gains for tracking tasks than for the contact ones.

One major point to discuss is the transition from the simulations to the real experiments. For torque control, in the Gazebo simulator, the joint torque control is almost perfect because the dynamics of the motor is completely neglected. However, not taking these dynamics into account will lead to unrealistic and dangerous behaviors on the real robot. The authors are currently testing the simulations on the simulator of the TALOS constructor, PAL robotics, which models the actuator dynamics of the robot. To do so, the torque control scheme is plugged to the constructor low-level controller, which computes a new command respecting the actuator dynamics. Only some tuning of the task gains seems to be needed to achieve good results.

Another concern is that the real robot is subject to imperfections such as errors in the chain actuation model, sensor noise, limited torque bandwidth or delays. In the presented implementation some actions against the possible problems have been implemented Englsberger et al. [74], such as adding filters on the signals retrieved from the robot and using the force sensors on the feet to improve the robustness of the base-estimator. Moreover, a PD+ controller has been implemented for the torque controller, to stiff the ankles, improving the quality of the feet landing. However, in simulation, this PD+ only increases the rigidity because of the feedback in position and velocity, and decreases the PGM, thus it has not been presented.

For position control, the simulations are dependent of the low-level control PID of the Gazebo plugin. These gains however have been modified to fit the real behavior of the motors. Thus, the position controllers should not need important modifications to be tested on the real robot. Yet, because the robot TALOS presents a flexibility in the hips which is not measured by the encoders, the simulations cannot be entirely realized on the real robot. The Appendix 4 describes the solutions developed to compensate this non-measurable hip flexibility of the robot TALOS (mentioned in Section 1.4). This flexibility leads to errors in the landing positions of the feet which cannot be compensated without a proper identification and modeling of the flexibility. In torque control however, this issue is mitigated because the flexibility is considered by the control system as external disturbances. Nonetheless, to achieve the experiments, it will be necessary control this flexibility; the solution presented in Villa et al. [206] is currently tested on the robot TALOS and our controllers will be adapted to take it into account. It is important to mention that the final real robot implementations will require slightly different gains and weights.

4.4.6 Conclusion

Three whole-body control implementations are compared in this chapter. Two of them are position based (with DCM and CoM admittance control): a Lexicographic QP using inverse kinematics and a WQP using TSID with an AM task. The last one is a WQP using TSID in torque with an AM task. They are evaluated in Gazebo on flat, uneven terrains and stairs climbing; on the criteria of trajectory tracking, energy consumption, passivity and computational cost.

In general, both position control schemes present the same results, with less energy consumption and higher passivity for the *TSID position* controller. A better tuning of the tasks gains may improve its results on the ZMP tracking.

On the other hand, the *TSID torque* controller shows better results in terms of

smoothness of the trajectory tracking, energy consumption, passivity of the walk - without impacts and can achieve a 60cm walk with steps of 1s in simulation. This confirms the high capabilities of a torque control scheme coupled with an angular momentum regularization (see for instance Atlas in DARPA robotics challenge Koolen et al. [54]). In average, the *TSID* controllers reach the 1kHz of control loop, necessary for real-time control, nonetheless, the *IK* scheme has the best computational time.

For our future work, we plan to control the hip flexibility of TALOS based on the identification work done in Villa et al. [206], so that we can evaluate the three controllers on the real robot.

4.5 Application on Human-Robot Collaboration for Navigation

4.5.1 Introduction

This section exposes the efficiency of the previous whole-body torque controller in a context of human-robot collaboration. It presents the results obtained during a collaboration with another PhD student of the Gepetto team, Isabelle Maroger, where the robot proactively walks alongside a human. The resulting paper Maroger et al. [24] is part of a French project, called ANR-COBOT, which aims at a collaboration between a human and a TALOS humanoid robot to carry and move a table. To smoothly achieve such a handling task, a good knowledge of the human dynamics while walking with a table is needed to allow the robot to anticipate and proactively follow the human movements. For now, the problem of walking with a table has been put aside and the issue has been reduced to the proactive tracking of a walking human by a humanoid robot. Thus, the paper focuses on the real-time prediction of human walking trajectory and its coupling with a Walking Pattern Generator (WPG) for a TALOS humanoid robot.

The paper presents a prediction model of human trajectory during gait and its coupling with a WPG which generates a TALOS humanoid robot CoM and footsteps along the predicted trajectory. This coupling allows a proactive tracking of a walking human by a humanoid robot, instead of a passive one, in order to ease every potential interactions during gait. The whole framework is assessed in simulation on Gazebo, as Fig.4.12 shows, and is described on Fig.4.13. This framework is designed to help a humanoid robot to predict the spontaneous human trajectory during gait according to its real-time CoM observation. This aims to allow the robot to move in accordance with its human partner. Thus, if the human accelerates, slows down or stops, the robot should act similarly while following the same trajectory as its partner.

This section focuses on the part of the paper corresponding to my work. It consists in the adaptation of the torque whole-body controller presented in Section 4.4.4 to successfully simulate the coupling of the WPG with the prediction model

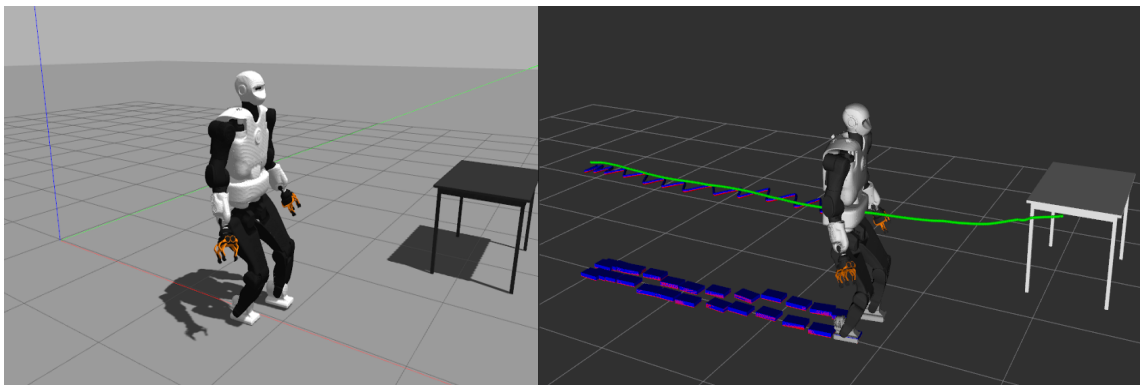

Figure 4.12: Simulation on Gazebo (*left*) of the robot executing a predicted trajectory. The CoM, footsteps (desired in *red*, real in *blue*) and human trajectories (in *green*) are also displayed in RViz (*right*) for comparison.

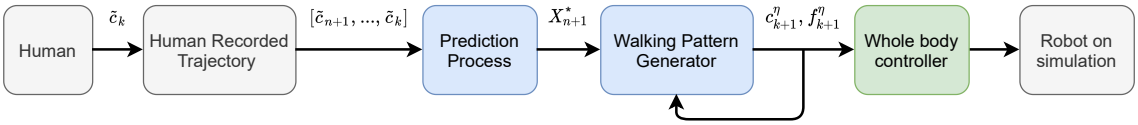

Figure 4.13: Description of the whole framework presented in Maroger et al. [24]

on the robot TALOS. It is represented by the green box. To achieve the simulations with the human-like trajectories, tasks gains and weights tuning were necessary and some of the trajectories cannot be performed because of their particularities. Some of them include wide lateral steps which lead to important swing of the torso, this behavior imbalance the robot and is the blocking point preventing our controller to achieve all the trajectories in simulation.

4.5.2 Scenarios

Three scenarios are considered in the paper:

1. The robot walks behind the human at a constant distance d_{behind}.

2. The robot and the human act synchronously so that no distance divides them. Experimentally, this scenario can be achieved by putting the robot and its human partner side by side to show that the robot is mimicking the human motions without any delay.

3. The robot walks ahead of the human at a constant distance d_{ahead}.

In all those scenarios, the robot is trying to follow the same trajectory as the human. Those scenarios are represented on Fig.4.14: the *green* and *yellow* curves display the whole past human trajectory, in *stippled green* the unknown future human trajectory and in *purple* the predicted trajectory.

4.5.3 Framework Description

4.5.3.a Prediction Model

To make the robot anticipate the human behaviour during gait, a model which allows to predict where a human is going using its past trajectory is needed. The model used in Maroger et al. [24] is based on the OCP model described in Maroger et al. [187] which had shown to generate paths which fit well human CoM trajectories. The OCP model is solved with a DDP solver Tassa et al. [151] from the Crocoddyl library Mastalli et al. [207].

The prediction process needs a human trajectory as an input. We used trajectories of 10 healthy subjects walking from 40 starting positions to one goal position in front of a table collected as part of a study of human walking trajectories Maroger et al. [186] Maroger et al. [187]. To simulate a human walking, the recorded trajectory was sent at a given rate, simulating the human velocity, to the prediction process thanks to a ROS framework Stanford Artificial Intelligence Laboratory et

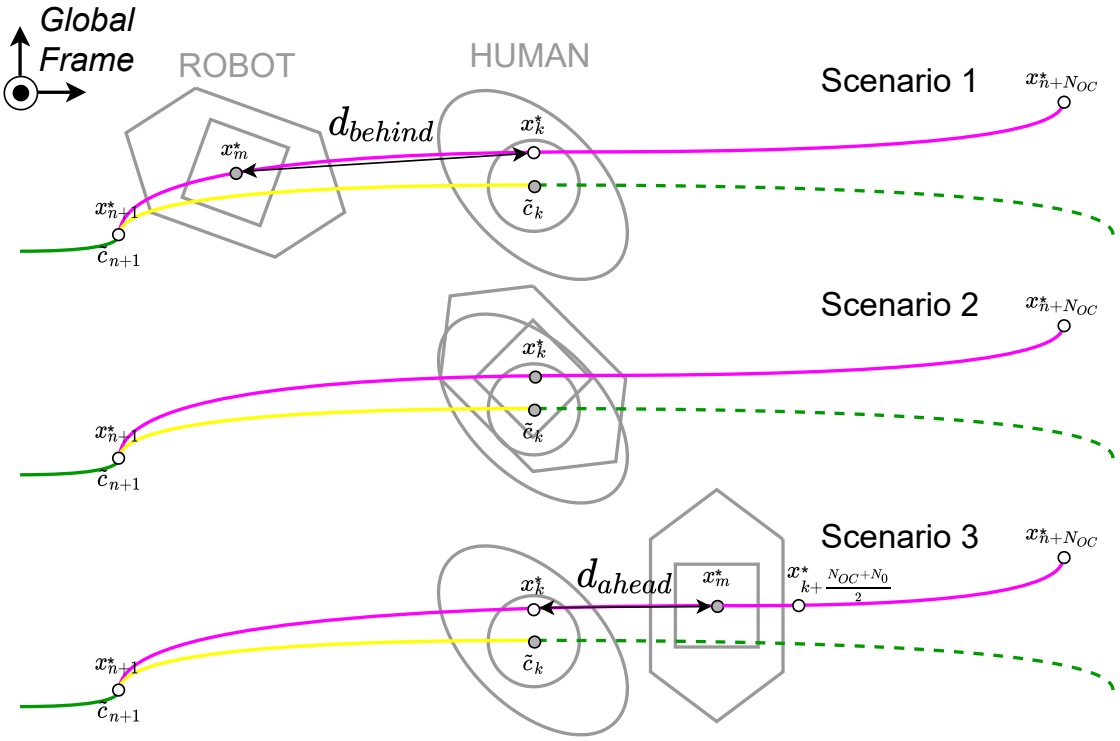

Figure 4.14: Representation of the 3 possible scenarios of Maroger et al. [24]: at the top, the robot walks behind the human, on the middle they are synchronized, at the bottom the robot walks ahead of the human

al. [7]. According to the chosen rate, the recorded trajectory could be sped up or slowed down.

Two parameters are fundamental in the prediction process: the amount of measurements needed before being able to correctly predict the human future trajectory and the sliding window size where the prediction process is done.

4.5.3.b Walking Pattern Generator

Then, as the future trajectory of the human can now be computed thanks to the prediction model described in the previous section, only the robot gait remains to be generated to perform a proactive tracking of a human. In Maroger et al. [24], a new WPG is designed to generate the CoM trajectory and the footsteps for a TALOS humanoid robot along the predicted trajectories. It is based on the Non-linear Model Predictive Control (NMPC) developed in Naveau et al. [69] which simultaneously solves the CoM and footsteps position and orientation problems. There are two main differences between the two NMPC:

- ▷ The controller of Maroger et al. [24] is designed to track a reference trajectory rather than a reference velocity which implies a different cost function in comparison with Naveau et al. [69].
- ▷ A terminal constraint on the Capture Point (CP) has been added in Maroger et al. [24] to avoid the internal instability issue in the problem formulation.

4.5.3.c Coupling of the Prediction Process and the WPG

The purpose of the paper Maroger et al. [24] is to design a whole architecture that allows a humanoid robot to smoothly follow a human who walks to an a priori unknown goal while predicting his future trajectory. This tracking is aimed to be proactive as a prediction process estimates the human future trajectory so that the future steps of the robot planned by the WPG are already in the right direction. This may cancel the reaction time of the robot and make the tracking smoother. The coupling of the prediction process and the WPG for a TALOS humanoid robot described in Sec.1.4 is a first step towards this purpose.

This coupling can be described as follows, it corresponds to the blue boxes represented in Fig.4.13. First, a pre-recorded human trajectory is streamed and retrieved by the prediction process. Then, it sends the predicted trajectory to the WPG which computes, thanks to a NMPC, the robot CoM and feet positions and orientations over the preview horizon with respect to the human velocity (the CoM orientation is in fact the robot base one). All those data are shared through a ROS framework and displayed on the RViz software as shown on Fig.4.15 for a specific trajectory. In this figure, the chosen scenario is the second one on Fig.4.14, namely the synchronization of a robot and a human. In the next section, the last steps of the process are detailed, namely the whole-body controller (the green box on Fig.4.13) and the simulations on the TALOS robot on Gazebo are presented.

All the results and simulations resulting of this coupling are reproducible as all the libraries are open-source and the source code and the data are available on: https://github.com/imaroger/human_walking_trajectory_prediction.

4.5.3.d Whole-body controller

The whole-body controller computes a stable command from the reference trajectories and the actual state of the robot. In the paper Maroger et al. [24], we used the controller described in Section 4.4.1.b: it is a Weighted Quadratic Program (WQP), which solves the inverse dynamics of a robot in rigid contact with the environment Herzog et al. [153]. It has been successfully tested on the humanoid robot TALOS in simulations in Ramuzat et al. [23], the results are presented in Section 4.4.4. The controller takes as inputs the CoM position, the base (free-flyer) orientation and the feet positions and orientations reference trajectories of the WPG. It implements task functions with them, as acceleration-based tracking law (see Section 4.3). The controller optimises its cost function using these tasks, prioritised with weights, while respecting constraints such as the robot dynamics and feet contacts. The weights and gains used in the paper are the same as the one of the torque controller presented in Section 4.4.4, except for the CoM and Angular Momentum (AM) tasks gains: the CoM proportional gains were raised to 800 and the AM ones decreased to 2.5. This gains tuning were needed to make the controller successfully perform the trajectory wide side steps found by the WPG.

Moreover, the orientation of the free-flyer given by the WPG was changing a lot and impacted the walking. As it was computed to be the mean of the feet orientations on each component in the WPG of Maroger et al. [24], we implemented it directly as this mean in our torque controller given the reference feet values. This solved the problem and we had a smoother orientation of the free-flyer, however the

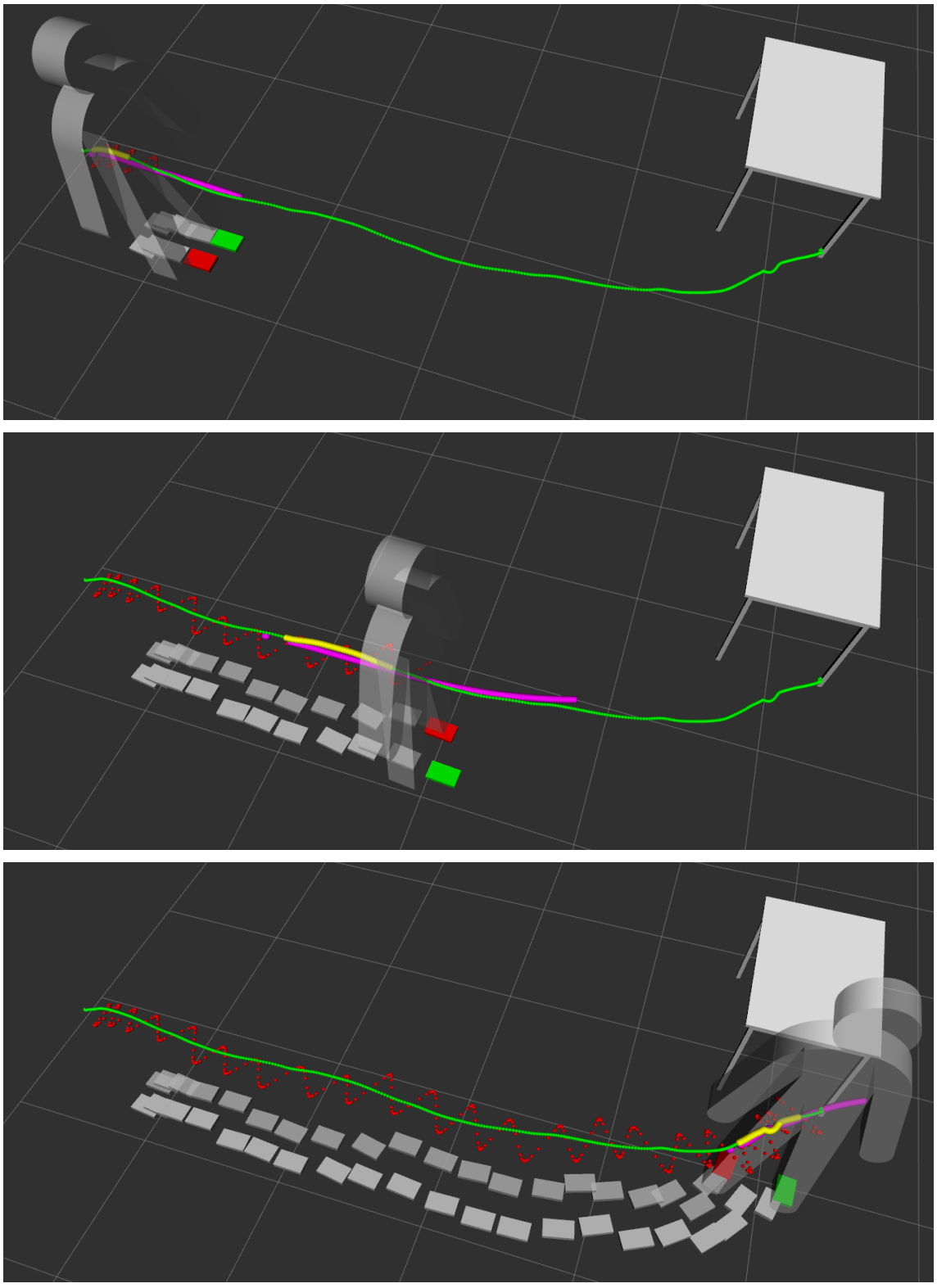

Figure 4.15: The robot CoM (in *red*) and footsteps (past steps in *grey*, current support foot in *red* and future support foot in *green*) are generated from the current predicted human trajectory (in *purple*)

robot was thus less pivoting its waist than the human during the simulations.

The interest of this controller is that it implements an AM regularisation task, which allows to control the angular momentum part generated by the contact transition Kajita et al. [51]. In particular, this task is fundamental here because the WPG feet reference trajectories contain wide lateral steps. These steps cannot be dynamically performed without the AM task in torque with our controller because the robot structure is not rigid, as opposed to position control. The wide lateral steps (in particular the ones followed by other lateral steps) lead to swings of the torso which are difficult to compensate without arms motions. Introducing the AM task solves this problem because to reduce the centroidal angular momentum the arms are naturally solicited to create motions around the torso to avoid the swings.

4.5.4 Simulation Results

The simulations have been realized using Gazebo on a standard laptop (Intel CPU i7-8850H @ $2.6GHz$). The reference trajectories from the WPG are registered in files which are read by the whole-body controller during the simulation. The controller computes the desired torque for all the joints of the robot at $1kHz$ and sends this command to the simulated robot in Gazebo.

The simulation result of the same trajectory presented in Fig.4.15 is shown in Fig.4.12, where the robot is synchronized with the human. The CoM, base and feet references are well followed by the controller, their tracking are presented in Fig.4.16, allowing the robot to successfully perform the motion. In particular, as explained in the previous paragraph, there are wide lateral steps to perform while keeping balance. There are 3 steps from 2.75m to 3m on the x-axis (with a total length of 25cm for the 3 steps) which have lengths of 20cm, 10cm and again 20cm on the y-axis (lateral plane). And there is a backward step from 3.9m to 3.75m on the x-axis which has a length of 22cm on the y-axis close to the end of the motion. These steps of around 20cm on the lateral plane are the ones needing the AM task and which are difficult to succeed for our torque controller.

The Figure.4.17 illustrates the results obtained for the first scenario, where the robot walk behind the human, here with a distance of 30cm. Similarly to the previous simulation, the references trajectories are correctly followed as presented in Figure.4.18. In this predicted trajectory there are less wide lateral steps, leading to a more stable motion. The whole prediction process and other simulations (in particular one with the scenario 3 where the robot walks ahead of the human) are shown in the video available at https://youtu.be/hu-cuUYl-58.

4.5.5 Discussion

Human walking trajectory model. The model used to predict the human trajectory depends a lot on the chosen human trajectory model. The model has been optimised in Maroger et al. [187] to generate smooth trajectory of the human CoM. That is to say that it does not take into account the oscillations of the CoM due to the steps. So, another model of human gait taking the steps into account could be consider in case of the prediction model needs to be more precise. Moreover, the chosen human trajectory model has been shown to fit well the average human behaviour

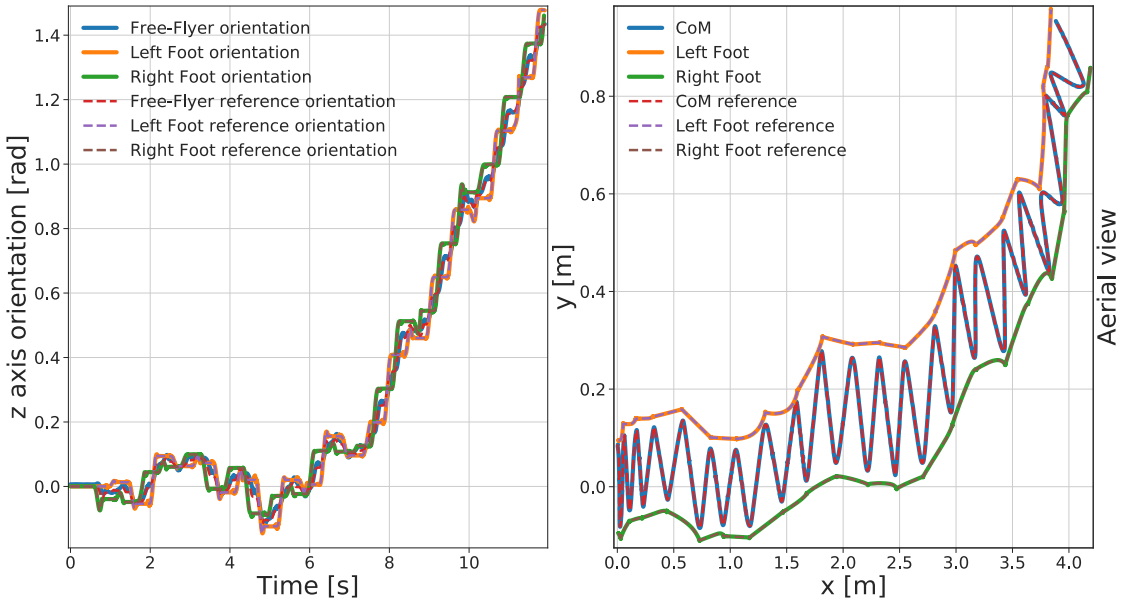

Figure 4.16: Tracking of the CoM and Feet trajectories in the Gazebo simulation where the robot is synchronized with the human (scenario 2).

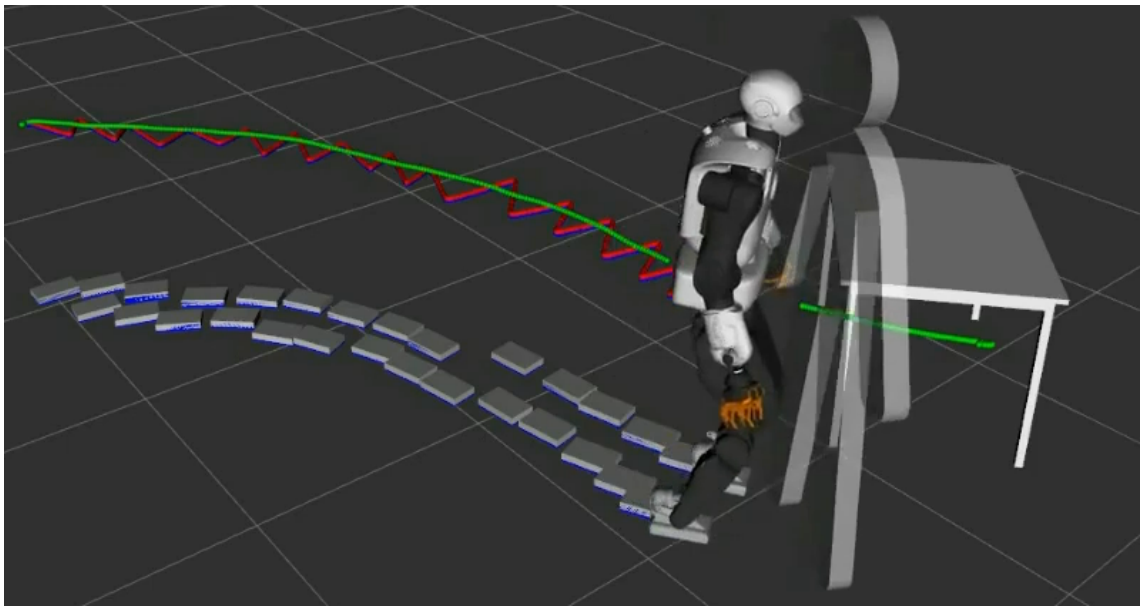

Figure 4.17: Simulation of the robot executing a predicted trajectory behind the human. The CoM, footsteps (desired in *red*, real in *blue*) and human trajectories (in *green*) are displayed in RViz but have also been realized in Gazebo.

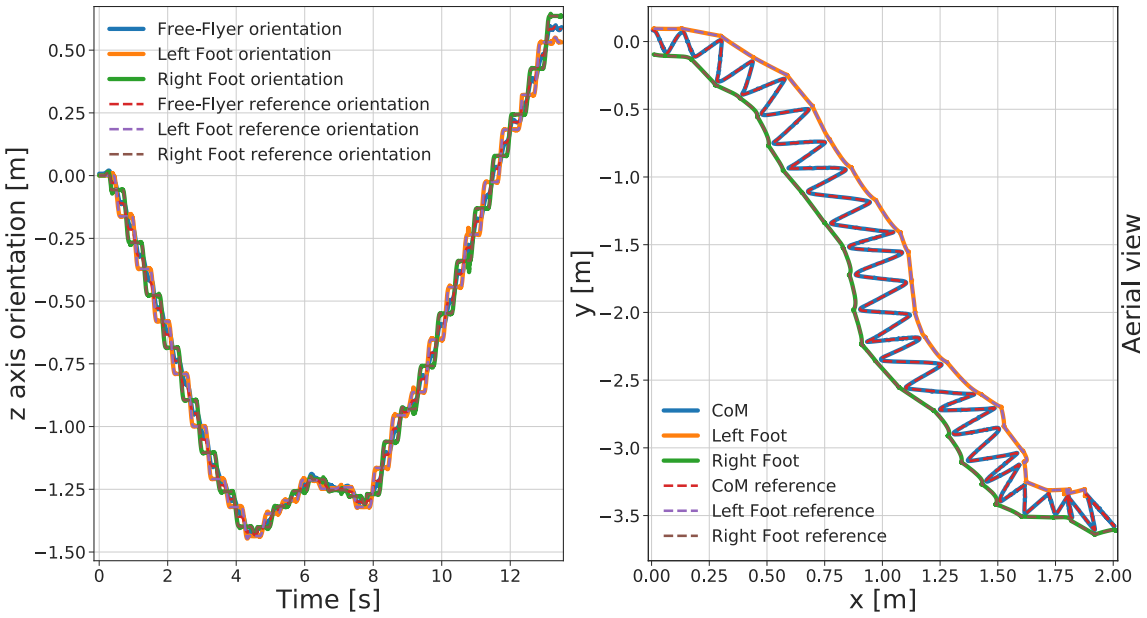

Figure 4.18: Tracking of the CoM and Feet trajectories in the Gazebo simulation where the robot is behind the human (scenario 1).

while its performance could be poorer when apply to individual subjects. However, when testing the prediction model over trajectories of numerous subjects, it seems to achieve its goals whoever the subject is Maroger et al. [24].

Role of the parameters in the whole process performance. The prediction process efficiency depends a lot on the sliding window size where the prediction process is done and on the amount of measurements needed to reach a good prediction. Indeed, a greater amount of measurements and a smaller window size lead to a more precise prediction. However, the process is still able to predict the human behaviour when the amount of measurements is low, even if this prediction is less accurate Maroger et al. [24]. Yet, for the whole-body controller using a large sliding window size leads to a more stable motion, indeed small a window size creates more variations. A numerical assessment of the prediction model is presented in Maroger et al. [208]. On the other hand, the precision of the human tracking performed by the WPG relies on the chosen scenario and on the potential chosen distance to the human. The greater the distance is when the robot walks behind the human, the cleaner the footsteps are, because the part of the predicted trajectory used by the WPG is mostly in the the measured part. On the opposite, the greater distance is when the robot is ahead of the human, the messier the footsteps are, because all those generated footsteps are based on the prediction which can vary a lot according to the measurements.

Toward a full online application on the real robot. The test on the TALOS robot is an ongoing work. Torque-based walking has been successfully realized. However this walking pattern generator has some limited speed. Developments are under way to reach a sufficient speed to realize a human-humanoid robot interaction.

4.5.6 Conclusion

In this section is presented the work realized in collaboration with Isabelle Maroger in a context of human-robot collaboration. The aim is to ease the tracking of a human by a humanoid robot by helping the robot to predict, and thus anticipate, its partner behaviour. To this end, three contributions have been made in Maroger et al. [24]: First, a prediction OCP model is designed based on a human trajectory model, which generates the future trajectory of a human from its recent past trajectory. Then, a WPG is designed to generate a TALOS humanoid robot CoM and footsteps along the predicted trajectory in real time. Finally the CoM and footsteps trajectories resulting from the WPG are given to a torque whole-body controller which computes a stable command for the robot to follow. This whole framework has been successfully tested in real-time simulation. Thus, in this work, we achieve to develop and assess the capabilities of a whole framework aiming to control a simulated TALOS humanoid robot in order to make him proactively walk along with a human partner.

4.6 Experiments realized on the real robot using the controllers

In this section are presented the results we succeeded to achieve on the real robot TALOS and the difficulties we encountered. These experiments are intermediate steps toward transferring the whole simulated results on the real robot. We detail the blocking points preventing us to successfully achieve these complete experiments.

4.6.1 Position Control

4.6.1.a Static stabilization

Using the whole body admittance control and the stabilizer described as the *IK* scheme in the Section 4.4, the team obtained good results for balancing during quasi-static moves and standing position. Indeed, the admittance control at the CoM allows a quick reaction when applying external perturbations such as pushing the robot. Fig.4.19 presents the reactive balancing of the TALOS humanoid robot when it is pushed from the front and from the side while standing on one foot. A video about this experiment is available at the following link: https://peertube.laas.fr/videos/watch/2dec7dba-cc57-4df4-8f10-a7d387404301

In the video push-recovery experiments are performed while the robot is standing on both feet and with one foot raised. One can notice that the robot is more stable with both feet on the ground, nonetheless, the *IK* scheme allows a good stabilization at the CoM level. The stabilizer correctly achieves the balance of the robot: it controls the DCM such that the CoM does not diverge and applies correct contact wrenches to avoid falling (neither slipping, nor presence of too much forces on one foot which imbalance the robot). It is important to underline that the admittance control is only implemented on the CoM, thus the robot is stiff on its upper parts while being more compliant on its lower parts (in particular the hips and ankles). This is why in the video pushing the robot arm produced motions on the whole robot and in particular its CoM.

The robot can achieve tasks with its upper body while external perturbations occur and keep its balance. It can also stabilize itself when non-dynamic trajectories are asked to the legs, or with no contact with the ground (for instance execute a swing on its foot). The difficulties appear when dynamic tasks are asked and involve the creation of contacts with the ground, typically during walking.

4.6.1.b Dynamic stabilization

The dynamic stabilization of the robot is an ongoing work. The actual implementation of the stabilizer should allow the robot to achieve this goal, however this is compromised by the flexibility in the hip of the robot TALOS (see Appendix 4). By tuning the gains of the admittance controller, the team managed to achieve once a straight walk of 20cm using the WPG reference trajectories of Section 4.4.4.a. The video of this successful motion is available at the following link: https://peertube.laas.fr/videos/watch/b56d80ed-7c6c-46a7-8750-fdb7ea6d1636

Latter on, we successfully achieved a repeatable on spot walking which is

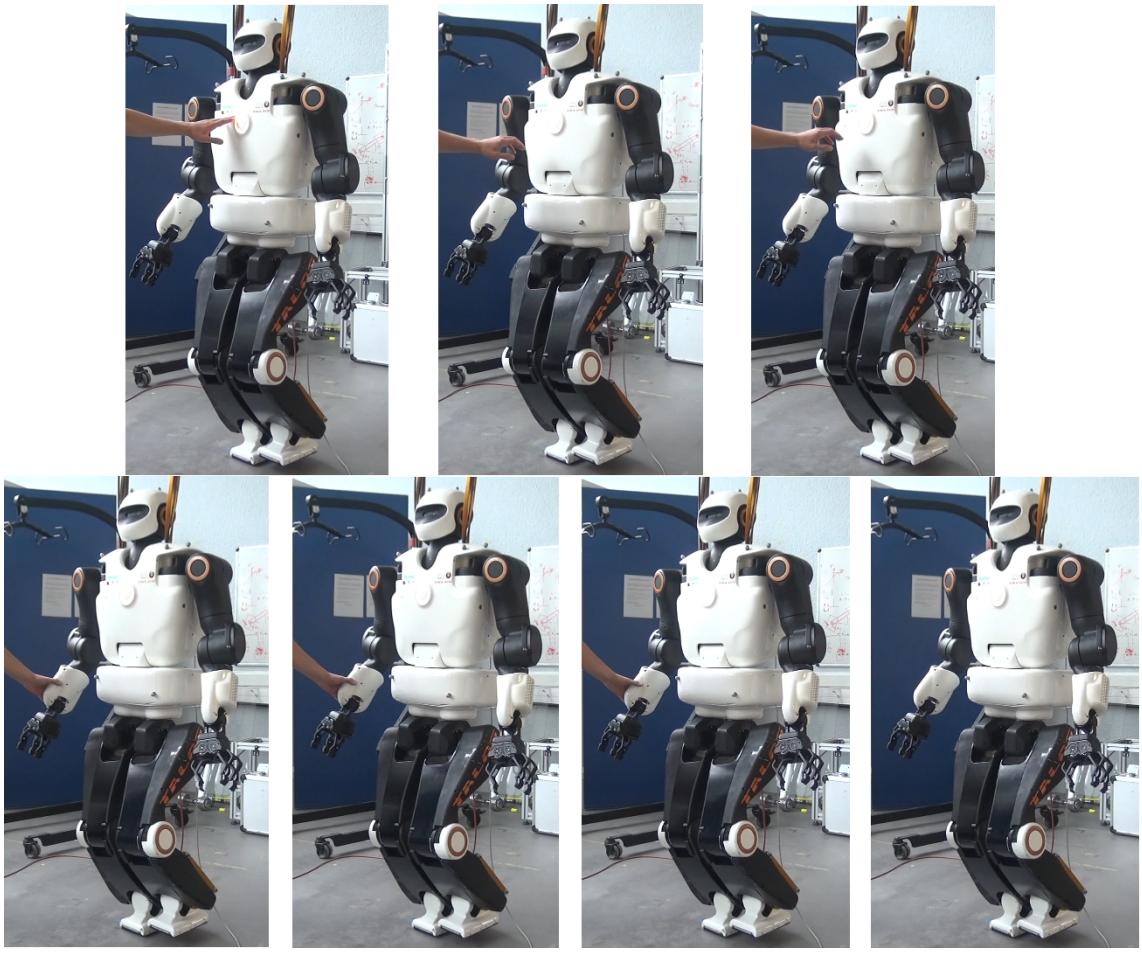

Figure 4.19: Experiments - Push recovery of the TALOS robot with one foot raised.

quite stable (See Fig.4.20). The video of this on spot walking is available at the following link: https://peertube.laas.fr/videos/watch/1a920902-c75f-4fb0-a638-33bb9b48d649. One can notice that the left wrist of the robot is tilted, indeed, its absolute and relative encoders did not send the same value. Thus, when controlling its position, the wrist had an abnormal behavior as its returned position was not the good one. We had to deactivate its control for the experiment and then fix the offset of the relative encoder.

In both videos the impacts on the ground are large and lead to instabilities, in particular slippage (which can also be caused by the flexibility in the hip). The robot has to move its upper body to compensate for them, because of that, at the end of the 20cm walk the robot almost fall. These impacts are partly due to the wrong positions of the feet when making contacts with the ground. The flexibility in each hip of the robot cannot be measured by the encoders, then it is creating an error between the positions given by the encoders and the real ones. These displacements at the hips are small, but when transferred at the feet positions it can lead to errors of up to 5cm. Thus, the controller is assuming a false position of the feet, and the robot enters in contact with the ground at a wrong position (even at the wrong moment, sooner if the displacement is in the direction of the walk or later in the opposite case). This is creating the large impacts and slippage, which prevented us to achieve a successful walking; this is why compensating this flexibility is necessary.

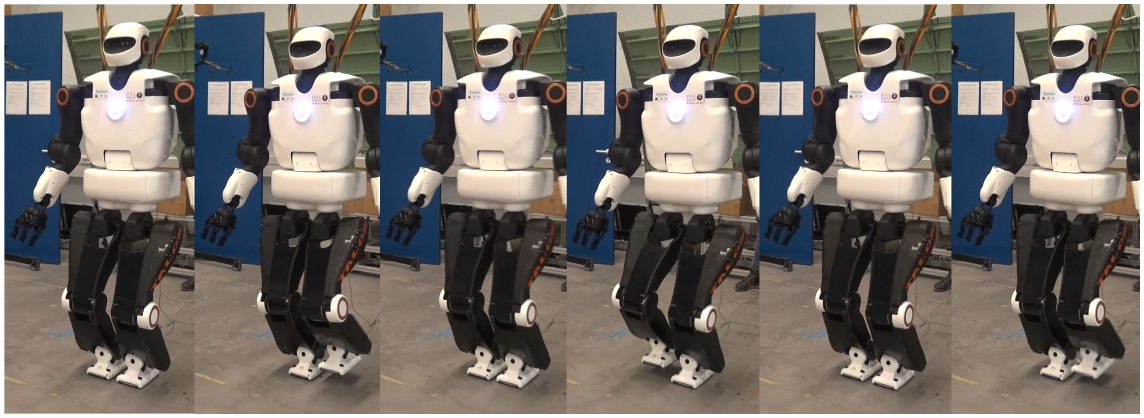

Figure 4.20: Experiments - On spot walking with the TALOS robot.

In the next subsection, the experiment realized to compensate it with a fixed value is presented.

An additional way to cope with the stabilization problem would be to reschedule the footsteps and their location according the landing time.

4.6.1.c Fixed compensation of the flexibility

As described in the Appendix 4 we first tried to compensate the flexibility by using a feed-forward on the commanded position of the hip taking into account the torsional stiffness and the measured torque. However, because of the noises on the torque sensors, we had to filter it, which leads the compensation to be applied with delays. We also tried to activate this compensation only on single support phases and not on double support ones to avoid accumulation of internal efforts (on double support the robot will try to correct its hip position while having its feet in contact with the ground and thus not moving, leading to this accumulation of energy). Even with such modifications the results were not enough to successfully perform repeatable walk.

Thus, we then tried to impose a fix compensation of the flexibility without taking into account the measured torque. With a leg of 1m weighting 20kg, we fixed the compensation on the hip to $\Delta q^{hip} = \frac{20}{K} \approx \frac{20}{973} = 0.021$ rad. Only a repeatable one step froward walk in position control has been successfully achieved with this method, see the video at the following link: https://peertube.laas.fr/videos/watch/08db3177-372b-43cc-85da-2009a267b5c9.

As the robot locomotion was not the core subject of my thesis and that we suspect the flexibility would be a less important issue for the robot during torque control; we stopped the work on the flexibility until the postdoctoral researcher Nahuel Villa came in the team. Then, he properly modeled this flexibility, as described in Appendix 4 and has recently achieved walking results on the robot using a CoP whole-body controller.

4.6.2 Torque Control

4.6.2.a PAL robotics low-level controller

To achieve torque control on the real robot, as explained in the Section 2.3.3 and in conclusion of the Section 4.4.4, it is needed to transform the joint torque commands to motor currents. We decided to use the PAL robotics constructor low-level controller, which computes new commands respecting the robot actuators dynamics. This low-level controller is a proprietary black-box, which use a *ros-control* hardware interface to communicate with the robot (see Fig.1.10). To interface our control scheme (based on the SoT with the WPG), we had to create a new version of the *roscontrol-sot* package, presented in Section 1.6.2.b. Indeed, our control scheme needs no more to communicate directly with the robot but with the PAL robotics controller, which implements different functions and formulations. One of the major difficulty is that the proprietary code source is not available, we only had access to its C++ headers and some basic tutorials. Developing this interface to keep all the functionalities implemented in the *roscontrol-sot* package (for instance to keep the recording of the logs and creating all the necessary signals needed by the SoT in the *dynamic-graph* structure), take us months of work (including the following remark).

Moreover, the robot has a modified operating system called ferrum (which is an equivalent to ubuntu). In order to have exactly the same environment as the one on the robot to test our codes, we created a Docker [209] container. Installing the SoT packages on this environment was not trivial as some packages had conflicting dependencies with the PAL robotics packages. Finally, we succeeded to test in this Docker container our interface and our torque controller using the PAL simulator available on ferrum. An additional difficulty is that the simulator (based on RViz) is slowed down when using our control scheme; i.e. the displayed behavior is estimated as five times slower than the real one. Therefore, a small and slow oscillation seen in the simulator is in reality a high frequency one on the robot and can lead to dangerous behaviors.

One has to note that, in Dantec et al. [147], the MPC is not embedded on the robot and is interfaced with the PAL robotics low-level controller via ROS topics. This simpler architecture choice was made because it is a stand-alone package (no SoT or *dynamic-graph* framework) and does not imply sending commands at high frequency (200Hz). ROS topics may induce latency and do not allow to send high frequency commands leading to real-time issues. To avoid the troubles I encounter to interface my control scheme with the PAL robotics low-level controller, the CoP whole-body controller of Nahuel Villa (not relying on the SoT nor *dynamic-graph* framework) compensating the flexibility was directly implemented in the PAL robotics control scheme.

Tasks	Priority	Weight
Feet contacts	0	-
Posture regularization in half-sitting	I	10

Table 4.11: Set of tasks for the torque control scheme on the posture task.

Experiments	Gains	Legs	Torso
Fail	$K_{Pposture}$	[800, 800, 800, 800, 800, 800]	[1000, 1000]
Success	$K_{Pposture}$	[50, 50, 50, 50, 50, 50]	[100, 100]
		Arms	Head
Fail	$K_{Pposture}$	[800, 800, 800, 800, 800, 800, 800, 800]	[100, 100]
Success	$K_{Pposture}$	[50, 50, 50, 50, 50, 50, 50, 50]	[10, 10]

Table 4.12: Tasks gains of the torque control scheme for the posture task.

4.6.2.b Experiment Results on a Posture Task

Once we achieved satisfying results on the PAL robotics simulator, we tested the classical formulation of our torque controller using inverse dynamics on the real robot on a simple postural task. The tasks weights and gains used are presented in the Tables.4.11 and 4.12, as the "Fail" experiment.

After few repetitive sinusoidal motions on the robot arm, the system diverged brutally and blocked two of its harmonic drives: the waist and the right shoulder (we pushed the emergency button but the robot reached anyway the harmonic drive blocks). The Fig.4.21 presents the result of the fail. After investigation it seems that the gains tuned in simulation (which simulates the actuation chains) were too high for the real robot. Thus, tuning the gains even on a proper simulator with the model of the actuators is not enough to ensure a safe solution. We know that some tuning is always necessary on the real robot, but we wrongly assumed that the solution would remain quite stable. Thus, to provide a safe and reliable interaction with the environment and possibly humans, we have looked for a way to ensure the system stability. The video of the failed experiment is available at this link: https://peertube.laas.fr/videos/watch/31fa2562-ba13-4043-a996-c2b8d5b21f4a. Unfortunately, it was not possible to repair the robot at the laboratory because the right shoulder and the torso were preventing the back cover of the robot to be removed (which needed to be removed to access the shoulder harmonic drive). Thus the robot had to be send back to PAL robotics premises for repair.

By lowering the gains value on the posture task, as presented in the Table.4.12, we succeeded to have a stable and compliant behavior of the robot. A video showing this compliant behavior is available at the following link: https://peertube.laas.fr/videos/watch/e9d8948d-08d5-4de9-8f42-2986fbbf0242, and depicted by the Fig.4.22. At the end of the video, the robot is falling because the contacts on the feet have been disturbed (the feet moved) breaking the constraint of the QP.

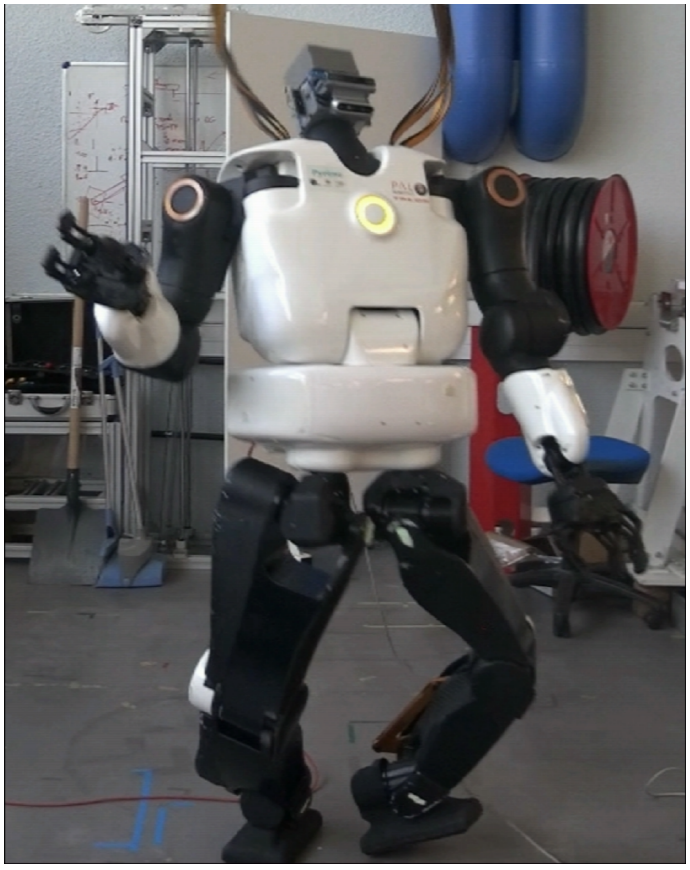

Figure 4.21: Failed experiment using torque control for a postural task.

This small success encourage us to add the CoM task for further tests. Unfortunately, this task does not work on the real robot, instead of correcting the CoM error the QP seems to make it diverge. It is a behavior that is not appearing in the simulator, where the experiment is working. After investigations, it may be due to incorrect torque offsets used in the simulator and not up-to-date PAL robotics packages in our Docker container. However, after updating these packages, our interface with the PAL robotics low-level controller did not work anymore. As we were working on the passivation of our control scheme (see the next Chapter), we put the experimental parts in standby and will try to fix this problem by the end of my thesis.

Figure 4.22: Successful experiment using torque control for a postural task.

4.7 Conclusion of the Chapter

In this chapter we presented the evaluations realized to achieve an efficient whole-body controller for the humanoid robot TALOS, able to follow correctly the reference trajectories given by the planning in real-time while keeping a low energy consumption. We detailed the force task implemented to achieve manufacturing operations. Finally, we prove the adaptability our torque controller by using it in different locomotion contexts such as walking in collaboration with a human, using a similar gait.

However, these evaluations have been made only on locomotion problems in simulations. Because the torque controller has proven to be adaptable, energy efficient and capable of quite high speed locomotion we choose to focus on this solution. Yet, when we tried our torque implementation on the real robot the system eventually diverges and the robot reaches its limits abruptly, blocking its harmonic drives. The investigation of the problem leads to decrease the controller tasks gains, however the question of the stability of our scheme was raised. Recalling the results obtained by Henze et al. [30] and other passive controllers, we decide to consider a way to make our controller respect the passivity of the system.

Thus, the following chapter presents the works realized in order to analyse the energy flow and passivity of our system. We detail the achieved results which modify the previous torque controller with a global energy tank to force the passivity of the system. The new scheme is tested in a multi-contact scenario employing the designed force task of Section 4.3.4 as a first step for manufacturing operations.

Chapter 5

Energy Analysis and Passivity of the Robotic System

Contents

In this chapter are detailed the results obtained in the submission Ramuzat et al. [26].

5.1 Introduction

Humanoid robots are complex systems designed to operate in similar environment than humans. One of their interesting purposes is to be able to achieve tasks that can be dangerous or too demanding for a person. This aim often relies on two main capabilities of the humanoid robots: the locomotion and manipulation. However, to achieve such a goal, these robots must interact with their environments (workspace, tools, other robots, even humans), which are mostly unknown and/or subject to disturbances. Even performing a simple task in a known and/or fixed environment can lead to instabilities. As stated in the Section 4.6 of the previous chapter, the authors encountered this problem while testing a classical torque control scheme using inverse dynamics on the robot TALOS on a simple postural task. After some repetition of a sinusoidal motion on the robot arm, the system diverged brutally and blocked some of its harmonic drive. Thus, to provide a safe and reliable interaction with the environment and possibly humans, we have looked for a way to ensure the system stability.

As presented in the Section 2.6, the two most common ways to prove the stability of a system are the Lyapunov or the Passivity analysis. Because of the results presented in Henze et al. [30] and of the proof on the passivity as a necessary condition for stability during robot interactions Folkertsma and Stramigioli [96], Camlibel et al. [106]; we chose the passivity analysis. This chapter details the analysis and results obtained toward the passivation of the torque controller presented in the previous Section The developed scheme is based on the mathematical adaptation of the passivity method introduced by Dietrich et al. [112] and is adapted for under-actuated humanoid robots, using the non-hierarchical multi-objectives controller TSID. To the best of the authors knowledge, no proof of passivity on an ID controller using energy tank on an under-actuated robot has been published or implemented yet.

Our contribution to the literature consists in a passive whole-body controller using inverse dynamics (as opposed to Henze et al. [119] which uses a PD+ structure) with energy tank (as opposed to Englsberger et al. [98]) which regulates the task gains. The concept of energy tank is rarely used on humanoids robot: it is used in Garofalo and Ott [108] with a strict hierarchy of two tasks and the proof of stability is only done on fully actuated robots. Our control scheme takes advantage of the energy tank to modulate our tasks gains, in order to respect the passivity of the system. It is similar to what is done in Englsberger et al. [98], where the task gains are functions of the current task inertia. However, our final control scheme has a constraint to respect the passivity inequality, not implemented in Englsberger et al. [98]. Indeed, because humanoid robots are under-actuated, the passivity is no longer guaranteed when constraints become active in the QP (such as the torque limits for instance). It is the case for the works presented in Garofalo and Ott [108] and Englsberger et al. [98], but in our scheme, this case is dealt with the imposed passivity constraint.

Therefore, our contribution is two-fold: the task gains of our whole-body controller are non-constant and regulated by an energy tank, and our scheme implements a constraint to respect the passivity even when other constraints become active in the QP. The classical inverse dynamics formulation is transformed in a non-linear problem which first computes the energy tank and regulating coefficients

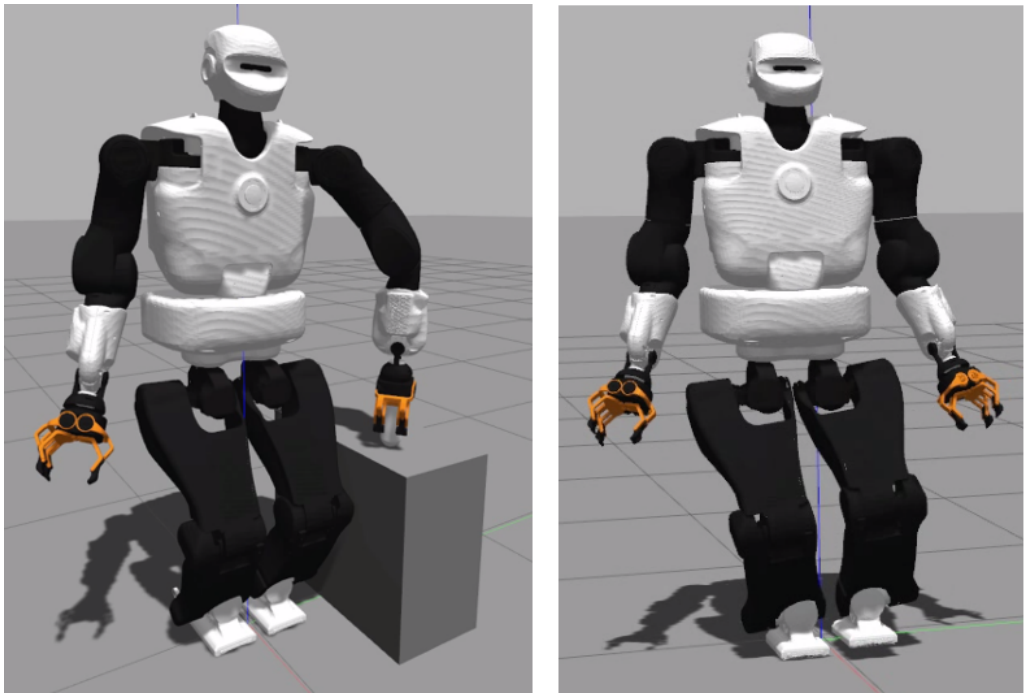

Figure 5.1: Simulations with Passivity: Left: Contact-Force Task - Right: Walk of 20cm steps.

and then solves the optimization problem. It is a sort of alternating minimization problem. Moreover we propose the implementation of this framework in an open-source package [80] and demonstrate its validation through a multi-contact scenario simulation and a locomotion one (see Fig.5.1). The multi-contact scenario is the first step toward a manufacturing operation such as drilling. It consists in applying a definite amount of force on a block (representing a part to manufacture) while maintaining a fix position; then this block is removed and the robot must keep its balance (see Fig.5.1). The removing of the block simulates a sudden lost of contact which can be due to slippage or, more importantly in the case of drilling, when the drill bit exits the hole (this can occur latter or sooner than what is planned and is introduced by a rapid decrease of the interaction force going down to 0).

We organize the chapter as follows: Section 2.6.3 recalls briefly the passivity theory presented in Section 2.6.3 while applying it to our system. Then, Section 5.3 describes the mathematical process to add a global energy tank and obtain the passivity proof in TSID. Finally, Section 5.4 presents the simulations used to validate the robustness of the controller and Section 5.5 discusses the obtained results.

5.2 Passivity Theory

5.2.1 Recall of the definitions

Section 2.6.3 details the definition of the passive system, which is a dissipative Willems [107] system respecting the following definition. We just recall here the Property 1 on the derivative of the storage function Schaft [104] (we remove the state or time dependencies of the terms for clarity):

A system with the state space model of $\dot{x} = f(x, y)$ *with initial state* $x(0) = x_0 \in \mathbb{R}^n$*, input vector* $u \in \mathbb{R}^m$ *and output* $y = h(x, u)$ *is said to be passive, if there exists a positive semi-definite function* $H : \mathbb{R}^n \to \mathbb{R}^+$*, called storage function, such that:*

$$\dot{H} \leq y^T u \quad \forall (x, y) \tag{5.1}$$

The passivity is a stability criterion based on the power flow exchanged by the components of the system. $H(x)$ is often chosen as the energy of the system, then, the system internal power ($\dot{H}$: derivative of the energy in the system) is lesser or equal to the power transferred to the system through its port ($y^T u$).

This property will be used in the following sections to prove the passivity of our control scheme. Then, let us recall the definition of a virtual energy tank, as the control framework will be augmented by a global one to store the energy dissipated by the tasks. The tank can be used to transfer the stored energy when a task becomes active (the task needs more energy to be completed). In opposition, when the tank is empty, it can be used to deteriorate the tracking of this task to ensure the passivity of the system. The energy tank is defined as a virtual storage element with flow variable $\dot{s}$ and effort variable s such that Giordano et al. [110], Dietrich et al. [111, 112]:

$$E_{tank} = \frac{1}{2} s^2 \tag{5.2}$$

5.2.2 Analysis of our System

Our system is composed of three components: the robot, the controller and the environment. As in Section 2.6.3, we assume that the environment of the robot can be described by a passive mapping ($v \to -\tau_{ext}$), in connection with impedance control Garofalo and Ott [108], Ott et al. [109]. Then, the group robot-environment is passive, so we have to make the group controller-robot passive. As in Ott et al. [109], the input of our controller is the velocity of the robot, with the joint angular velocity measured by the encoders and the free-flyer velocity calculated by the base-estimator. And the output of the controller is the control joint torque τ. Because the interconnection of passive systems leads to a general passive system we have two solutions: we can impose the passive mapping ($-Nv \to \tau$) between the robot and the controller, then because the group robot-environment is passive the overall system will be passive. Or, we can impose the passive mapping ($v \to \tau_{ext}$) between the system {robot+controller} and the environment, see Fig.5.2.

Note that because $N^T \in \mathbb{R}^{n \times n_j}$ is the selector matrix we have (taking the pseudo-inverse) $N^\dagger \in \mathbb{R}^{n_j \times n} = N^T$, $N^{\dagger T} = N$ and $N^{\dagger T} v = Nv = \dot{q}_j$ by definition (but $v \neq N^T \dot{q}_j$).

Then, as described in Section 2.6.3 (see Garofalo and Ott [108]), we have to find a storage function H such that:

$$\dot{H} \leq -(N^{\dagger T} v)^T \tau = -\dot{q}_j^T \tau = -v^T N^T \tau \tag{5.3}$$

Or a storage function H_{ext} such that:

$$\dot{H}_{ext} \leq v^T \tau_{ext} \tag{5.4}$$

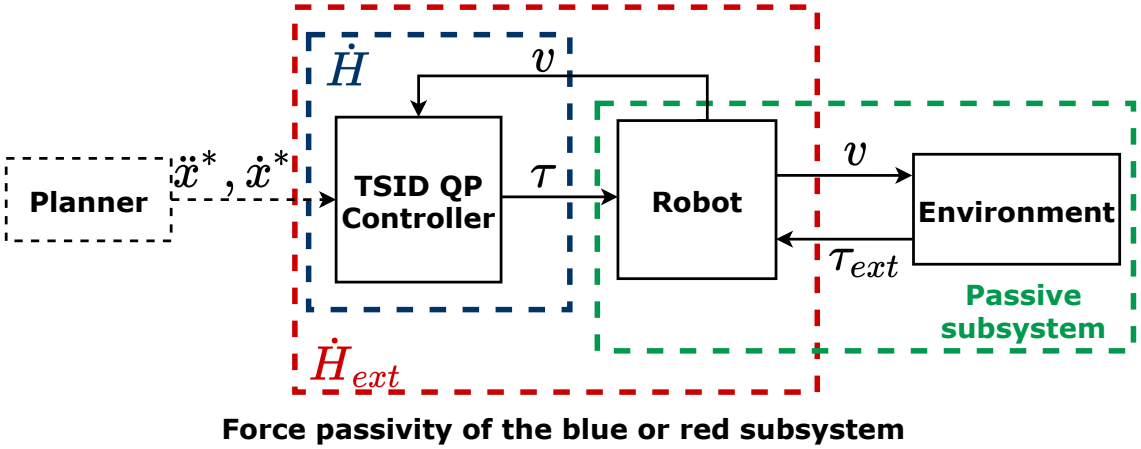

Figure 5.2: Port-based modeling of the subsystems and their relative power variables.

In the following, we prove the passivity of the system by building $\dot{H}$. Then we augment this function to obtain a $\dot{H}_{ext}$ satisfying Eq.5.4.

5.3 Formulation using Energy Tank in TSID

5.3.1 Choosing E_{tank} for a set of tasks

In this part we look at the proof of the passivity for a set of $\mathcal{N}$ motion tasks (see Eq.4.8). In this set of tasks, there must be at least a postural task regularization and one contact task with the environment. We consider here only one level of hierarchy (the cost, priority I), and assume that several tasks at this level can be put together as if it was only one task.

For one motion task x (see Eq.4.8), we define S the potential energy of the task:

$$\begin{aligned} S &= \tfrac{1}{2} e^T \Lambda K_P e \\ \dot{S} &= \dot{e}^T \Lambda K_P e + \tfrac{1}{2} e^T \dot{\Lambda} K_P e \\ \Lambda &= (J M^{-1} J^T)^{\dagger} \\ J^{\dagger} &= U D^{-1} V^T \end{aligned} \tag{5.5}$$

where K_P is the proportional gain of the task regularization in the controller (Eq.4.8), $J^{\dagger}$ is the Moore-Penrose pseudo-inverse of J, with U, D and V respectively the left singular vectors, the diagonal matrix of singular values and the right singular vectors of J (using the Singular Value Decomposition), $e = (x - x^*)$ and Λ is the semi-positive definite Cartesian space inertia matrix (multiplication property of the positive semi-definite matrix M, see Prop.2).

ΛK_p is semi-positive definite if K_p has equal positive elements (see Table.4.3) or under some specific conditions described in an open access document Ramuzat [210] which have been added in the Appendix 5 of the manuscript. Then, because $S = e^T \Lambda K_P e$ is a quadratic form of ΛK_p, S is always positive.

We then define the storage function for one task as:

$$H = S + E_{tank} \tag{5.6}$$

Then, for a set of tasks we define $\mathbf{H} \in \mathbb{R}^{(\sum_0^{\mathcal{N}} 1)\times 1}$ as:

$$\mathbf{H} = \begin{bmatrix} H_0 \\ ... \\ H_{\mathcal{N}} \end{bmatrix} \quad , \quad \dot{\mathbf{H}} = \begin{bmatrix} \dot{H}_0 \\ ... \\ \dot{H}_{\mathcal{N}} \end{bmatrix} \tag{5.7}$$

And accordingly, the vectors and matrix $\mathbf{J} \in \mathbb{R}^{L\times n}$, $\mathbf{K_P}$, $\mathbf{K_D} \in \mathbb{R}^{L\times L}$, $\mathcal{E} = \mathbf{X} - \mathbf{X}^* \in \mathbb{R}^L$, the respective stack of vectors and matrix of their non-bold value for the $\mathcal{N}$ tasks, with $L = \sum_{i=0}^{\mathcal{N}} l_i$.

From Eq.4.8 we obtain the following formulation for the set of motion tasks:

$$\begin{array}{rcl} \dot{\mathbf{X}} & = & \mathbf{J}v \\ \mathbf{J}a & = & \ddot{\mathbf{X}}^* - \dot{\mathbf{J}}v - \mathbf{K_P}\mathcal{E} - \mathbf{K_D}\dot{\mathcal{E}} \end{array} \tag{5.8}$$

It is important to notice here that $\mathbf{J} \in \mathbb{R}^{L\times n}$ is a *skinny* matrix, i.e. $L \geq n$ and is full rank. Indeed, as said before, the set of tasks contains at least a postural (acting on n_j DoF) and a contact task (acting on 6 DoF). As $n = n_j + 6$ in the case of a humanoid robot with a 6 DoF floating base, we have $L \geq n$. Thus, the generalized task space is greater than or equal to the joint space, leading to an over-determined problem. Moreover, the pseudo-inverse of $\mathbf{J}$ can then be written as Ben-Israel and Greville [211]: $\mathbf{J}^\dagger = (\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T$. And then by definition:

$$\mathbf{J}^\dagger\mathbf{J} = (\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T\mathbf{J} = \mathbf{I} \tag{5.9}$$

Finally, we can rewrite $\mathbf{\Lambda} = (\mathbf{J}M^{-1}\mathbf{J}^T)^\dagger$. By definition: $(AB)^\dagger = B^\dagger A^\dagger$ if A is full column rank and B full row rank Greville [212]. As $\mathbf{J}$ is *skinny*, it is a full column rank matrix and $\mathbf{J}^T$ is a full row rank matrix. Moreover, M is square and invertible, thus a full rank matrix. Then, $M^{-1}\mathbf{J}^T$ is also a full row rank matrix, leading to the fulfillment of the definition, we obtain:

$$\mathbf{\Lambda} = (\mathbf{J}M^{-1}\mathbf{J}^T)^\dagger = \mathbf{J}^{\dagger T}M\mathbf{J}^\dagger \tag{5.10}$$

Then, we can define the potential energy for the set of tasks $\mathbf{S}$ as:

$$\begin{array}{rcl} \mathbf{S} & = & \frac{1}{2}\mathcal{E}^T\mathbf{\Lambda}\mathbf{K_P}\mathcal{E} \\ \dot{\mathbf{S}} & = & \dot{\mathcal{E}}^T\mathbf{\Lambda}\mathbf{K_P}\mathcal{E} + \frac{1}{2}\mathcal{E}^T\dot{\mathbf{\Lambda}}\mathbf{K_P}\mathcal{E} \\ \mathbf{\Lambda} & = & \mathbf{J}^{\dagger T}M\mathbf{J}^\dagger \\ \mathbf{J}^\dagger & = & (\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T \end{array} \tag{5.11}$$

Following the same reasoning as for one task, $\mathbf{\Lambda K_P}$ semi-positive definite as $\mathbf{K_P}$ has equal positive elements along the tasks; thus $\mathbf{S}$ is always positive. Using the equation of the dynamics (Eq.2.4) and the formulation of TSID for τ_{ext} (Eq.4.13), we have:

$$-v^T N^T \tau \;=\; -v^T\Big[Ma + Cv + g - \textstyle\sum_{k=1}^{c} J_k^T T_k^T f_k - J_F^T T_F^T F_F\Big] \tag{5.12}$$

As the problem is over-determined, we can replace a by the Eq.5.8. The joints accelerations of the robot can be determined from the tasks accelerations. We obtain:

$$\begin{array}{rcl} -v^T N^T \tau & = & -v^T\Big[M\mathbf{J}^\dagger\Big[\ddot{\mathbf{X}}^* - \dot{\mathbf{J}}v - \mathbf{K_P}\mathcal{E} - \mathbf{K_D}\dot{\mathcal{E}}\Big] \\ & & +Cv + g - \sum_{k=1}^{c} J_k^T T_k^T f_k - J_F^T T_F^T F_F\Big] \end{array} \tag{5.13}$$

We can notice that, using the formulation of $\mathbf{J}^\dagger$ and $\mathbf{\Lambda}$ of Eq.5.11 and Eq.5.9, we have:

$$\begin{aligned} \mathbf{J}^T\mathbf{\Lambda} &= \mathbf{J}^T\mathbf{J}^{\dagger T}M\mathbf{J}^\dagger \\ &= (\underbrace{\mathbf{J}^\dagger\mathbf{J}}_{\mathbf{I}})^T M\mathbf{J}^\dagger = M\mathbf{J}^\dagger \end{aligned} \tag{5.14}$$

And thus the Eq.5.13 becomes:

$$\begin{aligned} -v^TN^T\tau &= -v^T\Big[\mathbf{J}^T\mathbf{\Lambda}\Big[\ddot{\mathbf{X}}^* - \dot{\mathbf{J}}v - \mathbf{K_P}\mathcal{E} - \mathbf{K_D}\dot{\mathcal{E}}\Big] \\ &\quad + Cv + g - \textstyle\sum_{k=1}^{c} J_k^T T_k^T f_k - J_F^T T_F^T F_F\Big] \end{aligned} \tag{5.15}$$

By definition $\dot{\mathbf{X}} = \mathbf{J}v$ and thus $\dot{\mathbf{X}}^T = v^T\mathbf{J}^T$. The previous equation is simplified in:

$$\begin{aligned} -v^TN^T\tau &= -\dot{\mathbf{X}}^T\mathbf{\Lambda}(\ddot{\mathbf{X}}^* - \dot{\mathbf{J}}v) + \overbrace{\dot{\mathbf{X}}^T\mathbf{\Lambda}\mathbf{K_P}\mathcal{E}}^{\mathbf{\Delta}} \\ &\quad + \dot{\mathbf{X}}^T\mathbf{\Lambda}\mathbf{K_D}\dot{\mathcal{E}} - v^TCv - v^Tg \\ &\quad + v^T\textstyle\sum_{k=1}^{c} J_k^T T_k^T f_k + v^T J_F^T T_F^T F_F \\ \mathbf{\Delta} &= \dot{\mathbf{S}} + \dot{\mathbf{X}}^{*T}\mathbf{\Lambda}\mathbf{K_P}\mathcal{E} - \tfrac{1}{2}\mathcal{E}^T\dot{\mathbf{\Lambda}}\mathbf{K_P}\mathcal{E} \end{aligned} \tag{5.16}$$

We know that $\mathbf{S}$ is positive, thus we want to choose the dynamics of our tank to contain the remaining terms (of which we do not know the sign):

$$\begin{aligned} \dot{\mathbf{E}}_{tank} &= \overbrace{\dot{\mathbf{X}}^T\mathbf{\Lambda}\mathbf{K_D}\dot{\mathcal{E}}}^{\text{Damping of the tasks}} + \dot{\mathbf{X}}^{*T}\mathbf{\Lambda}\mathbf{K_P}\mathcal{E} - \tfrac{1}{2}\mathcal{E}^T\dot{\mathbf{\Lambda}}\mathbf{K_P}\mathcal{E} - \dot{\mathbf{X}}^T\mathbf{\Lambda}(\ddot{\mathbf{X}}^* - \dot{\mathbf{J}}v) \\ &\quad + v^T\Big[-Cv - g + \sum_{k=1}^{c} J_k^T T_k^T f_k + J_F^T T_F^T F_F\Big] \end{aligned} \tag{5.17}$$

With this definition, $\dot{\mathbf{H}} = \dot{\mathbf{S}} + \dot{\mathbf{E}}_{tank} = -v^TN^T\tau$, ensuring the passivity of the system for the power-port $\{-Nv, \tau\}$. □

5.3.2 Including protection and valve

Once we have the desired formulation of the dynamics of the tank, we define α, β, γ, the coefficients that connect the tank to the system and regulate the tasks. They are used to bound the energy in the tank and to limit the power transferred to the system.

Let $E_{tank}^{max}, E_{tank}^{min}$ be the upper and lower bounds of our tank and P_{low} the power transfer limit. We have $\mathbf{E}_{tank} = \frac{1}{2}s^2$, thus, $\dot{\mathbf{E}}_{tank} = s\dot{s}$ and we build $\dot{s}$ to take into account the valves and satisfy $\dot{\mathbf{S}} + \dot{\mathbf{E}}_{tank} \leq -v^TN^T\tau$.

$\dot{s}$ is separated in two components: $\dot{s}^\sigma$ that we can control, linked to the tasks and $\dot{s}^\phi$ linked to the external components and thus cannot control (gravity, Coriolis and forces terms).

$$\dot{s}^\sigma = \frac{1}{s}\beta\gamma \underbrace{\Big[\dot{\mathbf{X}}^T\mathbf{\Lambda}\mathbf{K_D}\dot{\mathcal{E}} + \dot{\mathbf{X}}^{*T}\Lambda\mathbf{K_P}\mathcal{E} - \frac{1}{2}\mathcal{E}^T\dot{\mathbf{\Lambda}}\mathbf{K_P}\mathcal{E} - \dot{\mathbf{X}}^T\mathbf{\Lambda}(\ddot{\mathbf{X}}^* - \dot{\mathbf{J}}v) + v^TJ_F^TT_F^TF_F\Big]}_{B^\sigma} \tag{5.18}$$

$$\dot{s}^{\phi} = -\frac{1}{s}\underbrace{v^T\Big[Cv + g - \sum_{k=1}^{c} J_k^T T_k^T f_k\Big]}_{B^{\phi}} \tag{5.19}$$

$$\dot{s} = (1-\alpha)(\dot{s}^{\sigma} + \dot{s}^{\phi}) \tag{5.20}$$

Like in Shahriari et al. [113], to control the flow transferred from the tank to the tasks, we choose:

$$\gamma = \begin{cases} \dfrac{P_{low}}{B^{\sigma}} & \text{if } B^{\sigma} < P_{low} \leq 0 \\ 1 & \text{else} \end{cases} \tag{5.21}$$

$$\beta = \begin{cases} 0 & \text{if } (E_{tank} \leq E_{tank}^{min}) \ \& \ (\gamma B^{\sigma} - B^{\phi} < 0) \\ 1 & \text{else} \end{cases} \tag{5.22}$$

And like in Dietrich et al. [112], we choose α to be:

$$\alpha = \begin{cases} 1 & \text{if } (E_{tank}^{max} \leq E_{tank}) \ \& \ (\beta\gamma B^{\sigma} - B^{\phi} > 0) \\ 0 & \text{else} \end{cases} \tag{5.23}$$

α is the overflow valve, meaning that when $\alpha = 1$ the tank can be filled, otherwise the tank is already full and it cannot be filled more Dietrich et al. [112]. β is the opposite valve, it controls the output flow of the tank, if $\beta = 0$ the tank is empty (or at its lowest value) and the controller cannot use it to fulfill the tasks Dietrich et al. [112]. Finally, γ regulates the energy flow transferred from the tank to the controller. It avoids sharp release of the tank energy to the system, P_{low} is an empirical power threshold defining when the transfer should be regulated. When B^{σ} (the part of the tank derivative energy we can impact) is lesser than this threshold (negative: the system takes energy from the tank), γ takes the value of $\frac{P_{low}}{B^{\sigma}}$ Schindlbeck and Haddadin [125]. Then $0 \leq \gamma \leq 1$. To avoid discontinuities in the coefficient equations, the transitions between their two states are smoothen using the same approach as in Kronander and Billard [213]. The impact of the energy tank is propagated to the tasks with the parameters β, γ. They are introduced in the task functions such that we have new references $a^{\beta\gamma}$ and $F_F^{\beta\gamma}$:

$$\begin{aligned} Ja^{\beta\gamma} = J\beta\gamma a &= \beta\gamma\Big[\ddot{x}^* - \dot{J}v - K_P e - K_D\dot{e}\Big] \\ T_F^T F_F^{\beta\gamma} = T_F^T\beta\gamma F_F &= \beta\gamma\Big[F^* + P_F e_f + I_F\int_0^t e_f ds\Big] \end{aligned} \tag{5.24}$$

And thus the parameters appear naturally in the previous equations when replacing the acceleration and the force by $a^{\beta\gamma}$ and $F_F^{\beta\gamma}$ in the equation of the dynamics. The Eq.5.16 is then transformed in:

$$\begin{aligned} -v^T N^T\tau = \ & \underbrace{\begin{aligned}&\beta\gamma\Big[\dot{\mathbf{X}}^T\mathbf{\Lambda}\mathbf{K_D}\dot{\mathcal{E}} + \dot{\mathbf{X}}^{*T}\mathbf{\Lambda}\mathbf{K_P}\mathcal{E} + v^T J_F^T T_F^T F_F \\ &-\frac{1}{2}\mathcal{E}^T\dot{\mathbf{\Lambda}}\mathbf{K_P}\mathcal{E} - \dot{\mathbf{X}}^T\mathbf{\Lambda}(\ddot{\mathbf{X}}^* - \dot{\mathbf{J}}v)\Big]\end{aligned}}_{s\dot{s}^{\sigma}} \\ & \underbrace{-v^T\Big[Cv + g - \sum_{k=1}^{c} J_k^T T_k^T f_k\Big]}_{s\dot{s}^{\phi}} \\ & +\beta\gamma\dot{\mathbf{S}} \end{aligned} \tag{5.25}$$

And then, the terms of the energy tank derivative appear ($\dot{E}_{tank} = s(1-\alpha)(\dot{s}^{\sigma}+\dot{s}^{\phi})$), as the one of the potential energy of the tasks. One can notice that E_{tank} is only obtained by integrating $\dot{E}_{tank}$ over time, which can add drift. However, note that E_{tank} is only used to check if the tank is empty or full, its value is never used to change the scheme, as opposed to $\dot{E}_{tank}$ which is used (partially) in γ and in the passivity constraint. Thus, we think that the drift issue on the energy tank is not critical.

5.3.2.a Proof with valves for the power port $\{-Nv, \tau\}$

The role of α is clear when analysing the derivative of the storage function H in order to prove the passivity of the system. Choosing $\mathbf{H} = \mathbf{S} + \mathbf{E}_{tank}$ we have:

$$\begin{aligned}
\dot{\mathbf{H}} &= \dot{\mathbf{S}} + \dot{\mathbf{E}}_{tank} \\
\dot{\mathbf{H}} &= \beta\gamma\Big[\dot{\mathcal{E}}^T\mathbf{\Lambda}\mathbf{K_P}\mathcal{E} + \tfrac{1}{2}\mathcal{E}^T\dot{\mathbf{\Lambda}}\mathbf{K_P}\mathcal{E}\Big] + s\dot{s} \\
\dot{\mathbf{H}} &= \beta\gamma\Big[\mathbf{\Delta} - \dot{\mathbf{X}}^{*T}\mathbf{\Lambda}\mathbf{K_P}\mathcal{E} + \tfrac{1}{2}\mathcal{E}^T\dot{\mathbf{\Lambda}}\mathbf{K_P}\mathcal{E}\Big] + \\
&\quad s(1-\alpha)(\dot{s}^{\sigma} + \dot{s}^{\phi}) \\
\dot{\mathbf{H}} &= \beta\gamma\Big[\mathbf{\Delta} - \dot{\mathbf{X}}^{*T}\mathbf{\Lambda}\mathbf{K_P}\mathcal{E} + \tfrac{1}{2}\mathcal{E}^T\dot{\mathbf{\Lambda}}\mathbf{K_P}\mathcal{E}\Big] + \\
&\quad (1-\alpha)\Big[\beta\gamma\Big[\dot{\mathbf{X}}^T\mathbf{\Lambda}\mathbf{K_D}\dot{\mathcal{E}} + \dot{\mathbf{X}}^{*T}\Lambda\mathbf{K_P}\mathcal{E} - \\
&\quad \tfrac{1}{2}\mathcal{E}^T\dot{\mathbf{\Lambda}}\mathbf{K_P}\mathcal{E} - \dot{\mathbf{X}}^T\mathbf{\Lambda}(\ddot{\mathbf{X}}^* - \dot{\mathbf{J}}v) + v^T J_F^T T_F^T F_F\Big]\Big] \\
&\quad +(1-\alpha)\Big[- v^T\Big[Cv + g - \textstyle\sum_{k=1}^{c} J_k^T T_k^T f_k\Big]\Big]
\end{aligned} \tag{5.26}$$

Replacing the Eq.5.25 in Eq.5.26 we finally obtain:

$$\begin{aligned}
\dot{\mathbf{H}} &= -v^T N^T\tau - \alpha\Big[- v^T\Big[Cv + g - \textstyle\sum_{k=1}^{c} J_k^T T_k^T f_k\Big]\Big] \\
&\quad -\alpha\Big[\beta\gamma\Big[\dot{\mathbf{X}}^T\mathbf{\Lambda}\mathbf{K_D}\dot{\mathcal{E}} + \dot{\mathbf{X}}^{*T}\Lambda\mathbf{K_P}\mathcal{E} - \\
&\quad \tfrac{1}{2}\mathcal{E}^T\dot{\mathbf{\Lambda}}\mathbf{K_P}\mathcal{E} - \dot{\mathbf{X}}^T\mathbf{\Lambda}(\ddot{\mathbf{X}}^* - \dot{\mathbf{J}}v) + v^T J_F^T T_F^T F_F\Big]\Big] \\
&= -v^T N^T - \alpha\Big[- B^{\phi} + \beta\gamma B^{\sigma}\Big] \\
&\leq -v^T N^T\tau
\end{aligned} \tag{5.27}$$

Indeed because of Eq.5.23, if $\alpha \neq 0$ then $\beta\gamma B^{\sigma} - B^{\phi} > 0$, thus: $-\alpha\Big[\beta\gamma B^{\sigma} - B^{\phi}\Big] \leq 0$

Then from Eq.5.27 we can conclude the passivity of the system for a set of $\mathcal{N}$ tasks with respect to the power port $\{-Nv, \tau\} = \{-\dot{q}_j, \tau\}$. □

5.3.2.b Proof with valves for the power port $\{v, \tau_{ext}\}$

We can also prove the passivity of the overall closed-loop dynamics with respect to the power port $\{v, \tau_{ext}\}$. Let us augment the storage function $\mathbf{H}$ with the kinetic energy of the tasks and the gravity potential to obtain $\mathbf{H_{ext}}$:

$$\mathbf{H_{ext}} = \mathbf{S} + \mathbf{E}_{tank} + \frac{1}{2}\dot{\mathbf{X}}^T\mathbf{\Lambda}\dot{\mathbf{X}} + V_g(q) \tag{5.28}$$

We have by definition, $g(q) = \left(\frac{\partial V_g(q)}{\partial q}\right)^T$, leading to:

$$\frac{\partial V_g(q)}{dt} = \frac{\partial V_g(q)}{\partial q}\frac{\partial q}{dt} = \left(\frac{\partial q}{dt}\frac{\partial V_g(q)}{\partial q}\right)^T = v^T g(q) \tag{5.29}$$

Let us write the formulation of the equation of the dynamics for one task in the Cartesian space:

$$\Lambda\ddot{x} + \mu\dot{x} = J^{\dagger T}(-g + N^T\tau + \tau_{ext}) \tag{5.30}$$

With $\mu = J^{\dagger T}CJ^{\dagger} - \Lambda\dot{J}J^{\dagger}$. Then, for a set of $\mathcal{N}$ tasks we have, $\mathbf{Z}$ designating the stack of μ:

$$\mathbf{\Lambda}\ddot{\mathbf{X}} + \mathbf{Z}\dot{\mathbf{X}} = \mathbf{J}^{\dagger T}(-g + N^T\tau + \tau_{ext}) \tag{5.31}$$

By taking the derivative of Eq.5.28 and substituting Eq.5.31, we can write:

$$\begin{aligned}\dot{\mathbf{H}}_{\mathbf{ext}} &= \dot{\mathbf{S}} + \dot{\mathbf{E}}_{tank} + v^T g + \tfrac{1}{2}\dot{\mathbf{X}}^T\dot{\mathbf{\Lambda}}\dot{\mathbf{X}} + \\ &\quad \underbrace{\dot{\mathbf{X}}^T\left[\mathbf{J}^{\dagger T}\right.}_{v^T}(-g + N^T\tau + \tau_{ext}) - \left.\mathbf{Z}\dot{\mathbf{X}}\right]\end{aligned} \tag{5.32}$$

Using the fact that the Coriolis matrix is defined such that: $\frac{1}{2}\dot{M}(q) = \frac{C(q,v)+C(q,v)^T}{2}$, we have $\dot{X}^T(\frac{1}{2}\dot{\Lambda} - Z)\dot{X} = 0$ (skew-symmetric property, recall of the eq.2.5).
Finally we obtain, using the Eq.5.27:

$$\begin{aligned}\dot{\mathbf{H}}_{\mathbf{ext}} &= \underbrace{\dot{\mathbf{S}} + \dot{\mathbf{E}}_{tank}}_{\leq -v^T N^T\tau} + v^T N^T\tau + v^T\tau_{ext} \\ \dot{\mathbf{H}}_{\mathbf{ext}} &\leq -v^T N^T\tau + v^T N^T\tau + v^T\tau_{ext} \\ \dot{\mathbf{H}}_{\mathbf{ext}} &\leq v^T\tau_{ext}\end{aligned} \tag{5.33}$$

Delivering the proof of passivity for the port $\{v, \tau_{ext}\}$. □

5.3.3 Modified passive QP formulation

The classical inverse dynamics formulation is transformed in a non-linear problem with two stages. It is a sort of alternating minimization problem. The first stage (1) computes the energy tank dynamics and the regulating coefficients and then the second stage (2) solves the QP modified by these coefficients. One can notice that the first stage finds the coefficients for a torque which does not take into account the constraints of the QP. Thus as in Englsberger et al. [98], if a constraint became active, the passivity is no longer guaranteed because the computation used for the energy tank is not the same as the one of the QP. This is why we add a new constraint in the QP formulation, similarly to Joseph et al. [105], enforcing the inequality $\dot{\mathbf{H}} \leq -v^T N^T\tau$ to maintain the passivity in any situation. The general

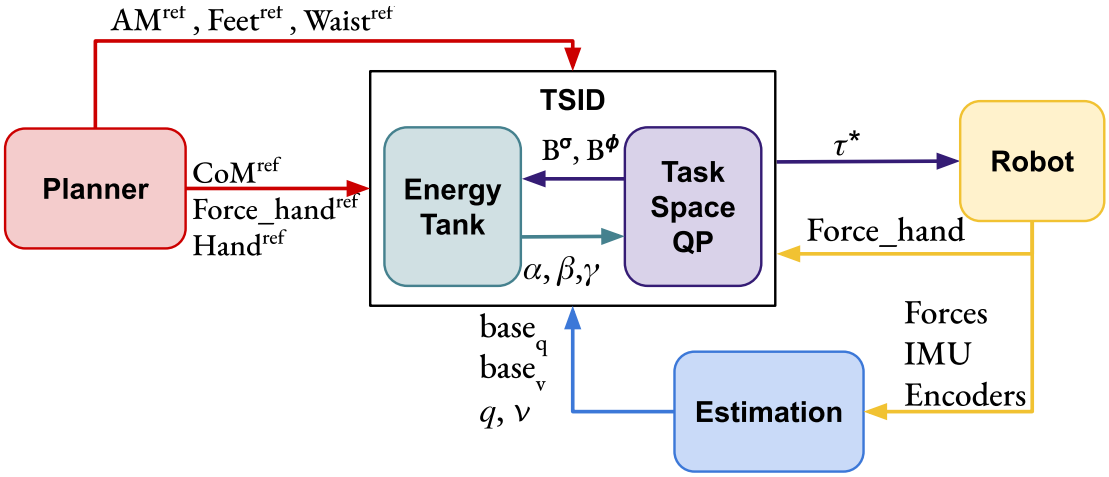

Figure 5.3: TSID torque control scheme with global energy tank.

control scheme is illustrated in Fig.5.3 and we obtain the new formulation:

$$
\begin{aligned}
&(1) \quad \text{Find} && \alpha, \beta, \gamma, \dot{\mathbf{H}} \\
&\quad \text{s.t.} && \text{Eq.5.18} - 5.23, 5.26 \\
\\
&(2) \quad \min_{y=[a,f]} && \sum_{i=0}^{\mathcal{N}} \lambda_i \left\| O_i y - \beta\gamma o_i \right\|^2 \\
&\quad \text{s.t.} && \begin{bmatrix} J_c & 0 & 0 \\ M & -J_c^T & -N^T \end{bmatrix} \begin{bmatrix} a \\ f \\ \tau \end{bmatrix} = \begin{bmatrix} -\dot{J}_c v \\ -h \end{bmatrix} \qquad (5.34) \\
& && \tau_{min} \leq \tau \leq \tau_{max} \\
& && q_j^{min} \leq q_j \leq q_j^{max} \\
& && \dot{q}_j^{min} \leq \dot{q}_j \leq \dot{q}_j^{max} \\
& && f_{min} \leq f \leq f_{max} \\
& && \dot{\mathbf{H}} \leq -v^T N^T \tau
\end{aligned}
$$

5.4 Simulations

In this section are detailed the simulations realized on Gazebo to demonstrate the validity of our implementation. The tests are performed on a humanoid robot TALOS. The chosen simulations are a force application task performed by the left hand and a walk of 20cm steps on flat ground. These simulations allow to cover multi-contact and locomotion skills in stationary environment. The set of tasks considered during the first simulation are presented in the Table.5.1, and in the Table.5.3 for the second one. The task gains are defined in Table 5.2. The authors have implemented this controller using the TSID [80] library in the open-source package [202]. The design parameters of the global energy tank are set as follows: $E_{tank}^{max} = 5$J, $E_{tank}^{min} = 0.1$J and $P_{low} = -0.5$W. They are set as low values in order to keep small the contribution of the tank energy compared to the overall energy of the system. In the simulations, we use the robot base-estimator of Flayols et al. [29]. The base information is estimated using robot joints configurations and velocities, the IMU and the force sensors of the robot. For the force task simulation, no planning methods were used as the robot remains in the same configuration. The force references were created by linear interpolation between the actual and desired force to be applied on the block. The reference trajectories of the 20cm step walk were computed with the multicontact-locomotion-planning framework [62] (see Section 2.4).

5.4.1 Pushing task with unplanned contact removal simulation

In this simulation the robot makes a contact between a tool and a surface and applies a 30N force along the z-axis. In Gazebo, the tool is simulated by a cylinder and the surface is a simple square block, shown in Fig.5.1. This set-up is a first step toward more complex operations, requiring contact and a force application, such as drilling. It exposes the problem of unexpected broken contact that can be due to slippage, disturbances or when the drill bit exits the hole. In this simulation, the reference force is raised in two steps, the first one reaches 10N to stabilize the contact, in Fig.5.4 some oscillations can be seen in the first slope, leading to some instability in the tank coefficients in Fig.5.5. Then it is set to 30N, and at 32.3s the block is removed. The energy tank is initialized at 3J.

As shown in Fig.5.4, the measured force along the z-axis falls to 0N when the block is removed. We first try a simulation without the energy tank and the passivity

Tasks	Priority	Weight
Feet contacts	I	100
CoM tracking	I	50
Cartesian Force-Contact	I	10
Waist orientation	I	1
Posture regularization in half-sitting	I	0.1
AM regularization to 0	I	2×10^{-2}

Table 5.1: Set of tasks used in the contact simulation.

Tasks Gains	K_P	K_D	K_I
CoM	[50, 50, 50]	[5, 5, 5]	-
Waist	[100, 100, 100]	$2\sqrt{K_P}$	-
Feet Contacts	[30, 30, 30, 30, 30, 30]	$2\sqrt{K_P}$	-
Feet Motions	[100, 100, 100,	[5, 5, 5,	-
(walk simulation)	100, 100, 100]	5, 5, 5]	-
Force-Contact (force simulation)	[10, 10, 10, 10, 10, 10]	-	$0.5K_P$
AM	[10, 10, 10]	$2\sqrt{K_P}$	-

Posture	Legs	Torso	Arms	Head
K_P	[80, 80, 80, 80, 80, 80]	[80, 80]	[80, 80, 80, 80, 80, 80, 80, 80]	[80, 80]
K_D	$2\sqrt{K_P}$	$2\sqrt{K_P}$	$2\sqrt{K_P}$	$2\sqrt{K_P}$

Table 5.2: Tasks gains of the control scheme.

constraint. As a result, the robot quickly falls and cannot react in time to try to balance itself or to remove the force task when detecting the loss of the contact. On the contrary, with the formulation presented previously, when the block is removed the robot does not fall immediately. Indeed, as presented in Fig.5.5, the α and γ coefficients are triggered. With the γ closes to 0, the force task is highly penalized and thus the hand slowly comes down. As a result, the system has the time to detect the loss of the contact and remove the task, leading to the pick in the consumption of the energy tank at 34.4s. By removing the task, the robot goes back to its initial position because of the postural task and thus raises its hand, going against the gravity and consuming energy. On can notice a continuous growth of energy during the steady state between 20 and 30 s. We think that this growth is due to the remaining error of the postural task which desired position is set to the half-sitting of the robot and not the contact configuration.

Because the energy constraint is enforced in the TSID constraints, the QP ensures that the passivity holds, i.e. $\dot{\mathbf{H}} \leq -v^T\tau$. The Fig.5.6 presents the evolution of the potential function derivative $\dot{H}$ during the removal of the block (at 32.2s) and the task. We also display the results without energy tank, when the robot falls. We can see that the mechanical power of the system is controlled in the passive implementation case whereas it presents peaks in the simple case, leading to the fall. One can notice that around second $34 - 35$ the tank energy slope is much steeper than -0.5 W (corresponding to the value of P_{low}). This is because the task is removed and the robot goes back to the postural task position (straighten up), acting against the gravity. Keep in mind that $\dot{E}_{tank} = s(1-\alpha)(\dot{s}^{sigma} + \dot{s}^{phi})$, thus while $\dot{s}^{sigma}$ is regulated by γ, $\dot{s}^{phi}$ is not as we cannot control it. Thus, the high values of $\dot{H}$ correspond to the power delivered by the gravity, Coriolis and contact forces terms.

For the moment to remove the task we use a hand-programmed emergency action, but it should be implemented in a better way in the QP. Or for instance, having a finite-state-machine which would tell the QP which tasks should be added/removed in function of the energy tank parameters.

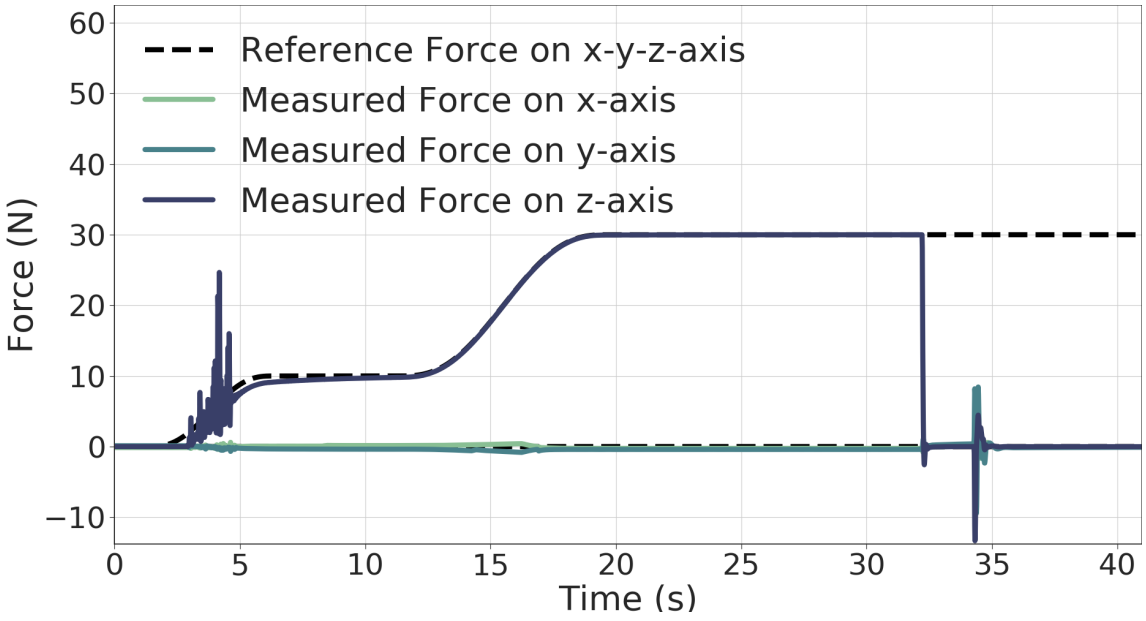

Figure 5.4: Reference and measured forces of the pushing task.

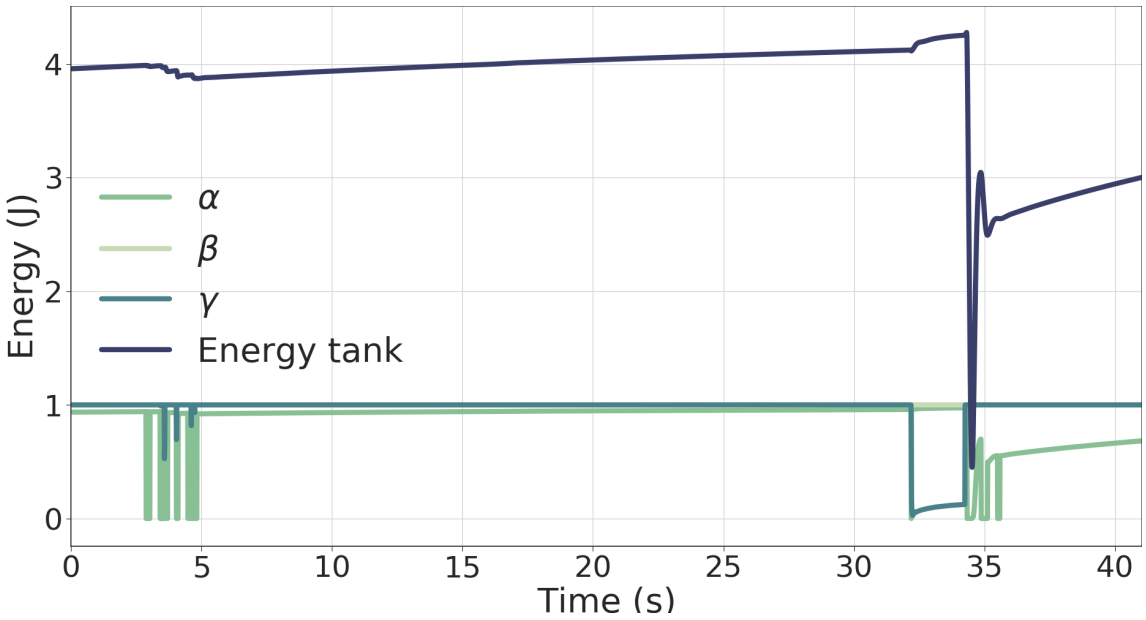

Figure 5.5: Energy tank and coefficients evolution during the force simulation.

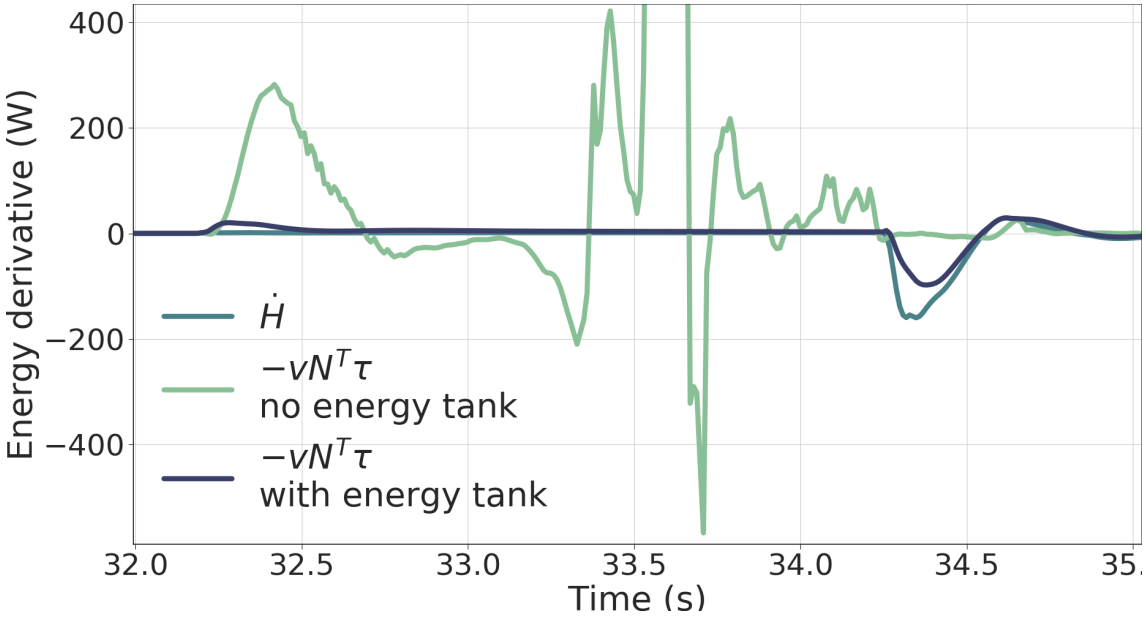

Figure 5.6: Energy derivative $\dot{\mathbf{H}}$ and its bound $-vN^T\tau$ during the removal of the bloc and the task. The result when no energy tank is added is also presented.

Tasks	Priority	Weight
Feet tracking	I	100
Feet contacts	I	100
CoM tracking	I	50
Waist orientation	I	1
Posture regularization in half-sitting	I	0.1
AM velocity-acceleration regularization	I	2×10^{-2}

Table 5.3: Set of tasks used in the walking simulation.

The video of the simulation is available at the following link: https://peertube.laas.fr/videos/watch/f32023d8-dda8-45b6-8c22-d1cd1a0285a5

5.4.2 Walk of 20cm steps

In this simulation the robot walks 1 meter forward on flat floor with 20cm steps, starting with the left foot (shown in Fig.5.1). The first and last steps are only 10cm long, and the first motion to swing the CoM of the robot is done in quasi-static. The double support duration is of 0.2s and the single support duration is of 1.2s. The set of tasks and gains used are the same as the previous simulation without the Cartesian Force-Contact task and with feet tasks motions during the single support stages. For clarity, the set of tasks are given in the Table.5.3 and the gains of the feet tasks motions are: $K_P = [100, 100, 100]; K_D = [5, 5, 5]$, see Table.5.2.

In the Fig.5.7 is presented the energy tank evolution during the simulation, with the regulating coefficients. One can see that at the beginning of the motion during the quasi-static swing the energy tank is emptied, leading the β parameter to decrease to 0 around the 10th seconds. The oscillations of the parameters lead

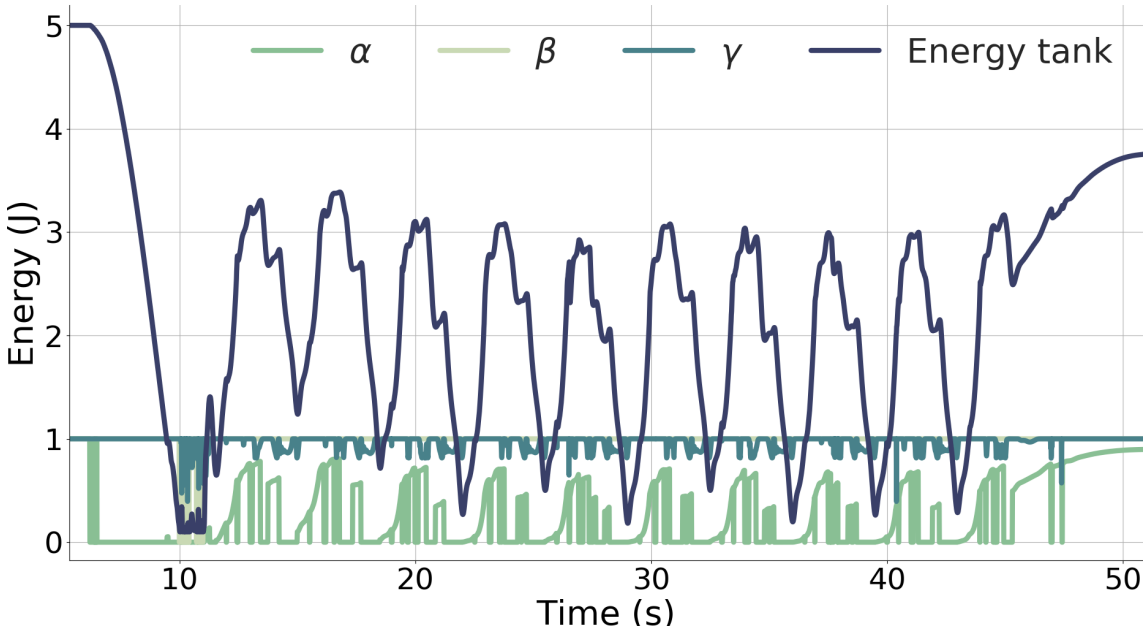

Figure 5.7: Energy tank and coefficients of the passive walk of 20cm steps.

the robot to oscillate as well because the β, γ parameters are used in the task errors formulation (see Eq.5.27). This behavior can be seen in the Fig.5.8 and 5.9 that respectively depict the CoM tracking along the y-axis (lateral plan) and the feet tracking along the z-axis (height of the steps, the actuators being positioned in the ankles, on the ground the sensors give a position of 0.107m). As a result, the potential function derivative $\dot{H}$ is really noisy around the 10th seconds (see Fig.5.10) and must have activated the constraint in the QP. To avoid this concerning behavior one can increase the budget of the energy tank (set at its maximum value, 5J, at the beginning of the motion), or try to replace the quasi-static part by a dynamic one to see if the tank is emptied. We think that the better solution should be to design the energy tank and P_{low} parameters in function of the reference trajectories Shahriari et al. [113].

Except from the concerning beginning, the robot achieves the walking motion without trouble with a good tracking of the reference trajectories, as shown in the Fig.5.8 and 5.9. This motion can be seen as a repetitive sinusoidal one on the axis presented for the CoM and feet. This is reflecting on the energy tank behavior, as it is filled and then emptied in a sinusoidal manner. The increases of energy in the tank correspond of the motions taking the CoM in the middle of the support polygon and thus, the feet in double support. Reciprocally, the tank energy decreases when the robot CoM reaches the edge of the support polygon above the supporting foot. This behavior can be explained by two facts:

- ▷ First, in double support the feet motion tasks are removed, leaving only the contact motion tasks which present small displacement errors if any. The feet do not move, thus there is also no error or task velocity nor acceleration (no damping energy). This typically means that $\dot{S}_{contact} = 0$ and $\dot{s}^{\sigma}_{contact} = 0$ (see Eq.5.18), thus the contacts task motions do not contribute much to the storage function, as opposed to the feet motion tasks which consume energy due to the small tracking delay.

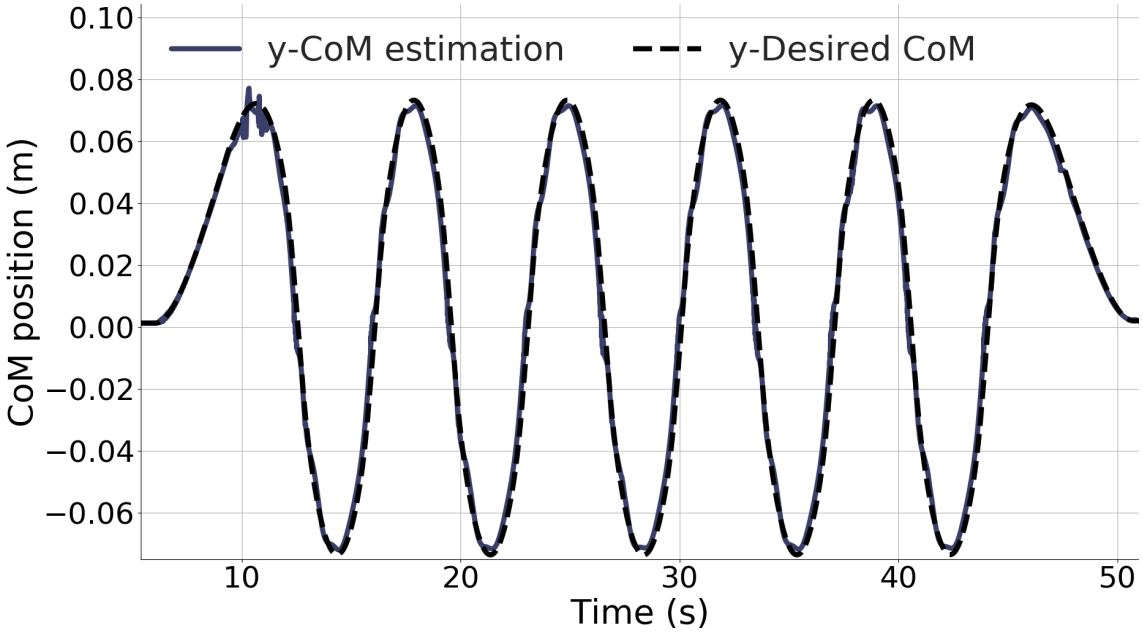

Figure 5.8: CoM tracking on the y-axis of the passive walk of 20cm steps.

▷ Secondly, the postural regularization task is set to the half-sitting position. Thus, when the robot swing on one side and another it is the furthest to its reference posture, the middle position is the closest to it, leading to this sinusoidal behavior of the energy tank. One may try to change the postural reference following the walk to see the evolution of the behavior.

The values of $\dot{H}$ oscillate a lot but remain under the passivity bound. Moreover, one can retrieve the sinusoidal pattern in it linked to the tasks behaviors. Looking at γ in the Fig.5.7, even if it oscillate its is always greater than 0.5 (except at the beginning). Thus the tasks are not too penalized and the robot can perform the walk. This means that the oscillations of $\dot{H}$ are either due to the non controllable part of the energy tank ($\dot{s}^{\phi}$) or from the potential energy of the tasks $\dot{S}$. Looking at the behavior of $\dot{S}$ for each tasks, the sinusoidal pattern seems to correspond to the feet potential energy while the oscillations come from the potential energy of the CoM when adding contact. This was expected, as the CoM reflects the action of the contact forces at the centroidal level (linked with the CoP). The behaviors of $\dot{H}$ with respect to $\dot{S}_{CoM}$ and $\dot{S}_{feet}$ is presented in Fig.5.11 to illustrate this comment.

The video of the simulation is available at the following link: https://peertube.laas.fr/videos/watch/7b5c3ae2-10d1-48da-a2ba-5d497e9c0249

This simple locomotion simulation presented does not show an application where the passivity approach is better than the classical one. But it demonstrates its capabilities on complex tasks as locomotion. The authors think that a walk where some parts of the ground are lower than planned can be a good application where the passivity scheme would behave better than the classical one. For instance, a walk where the robot goes down an unplanned step platform. The CoM and feet heights of the robot will then be wrong and the QP will try to correct them. Without the passivity, the classical approach would raise the CoM and accumulate errors on the feet, trying to make contact higher than the ground, potentially leading to

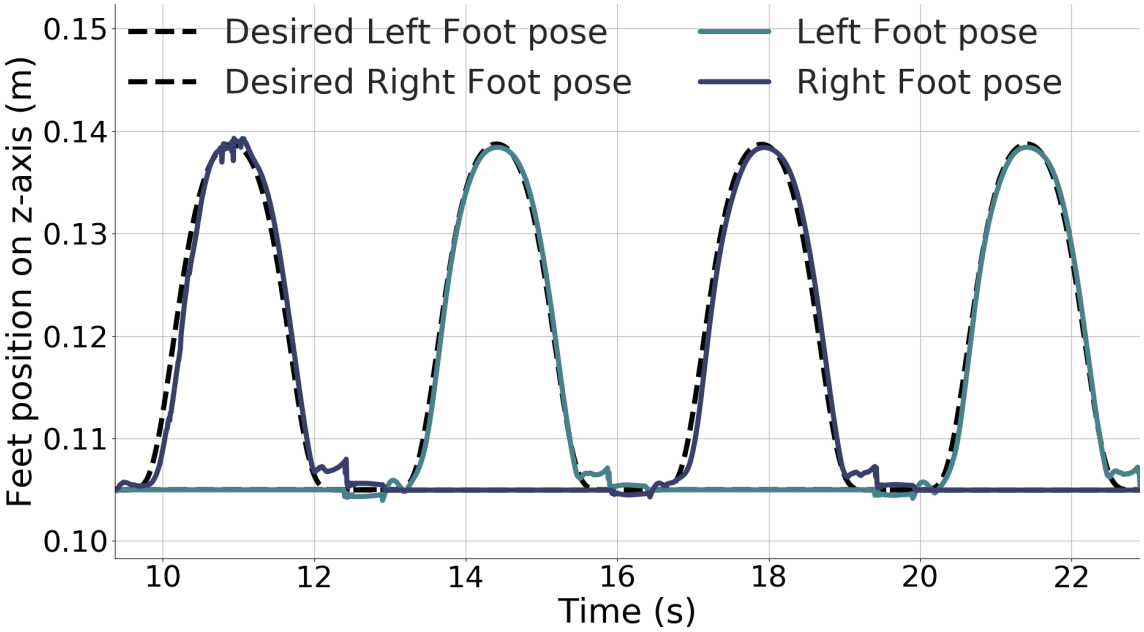

Figure 5.9: Feet tracking on the z-axis of the passive walk of 20cm steps.

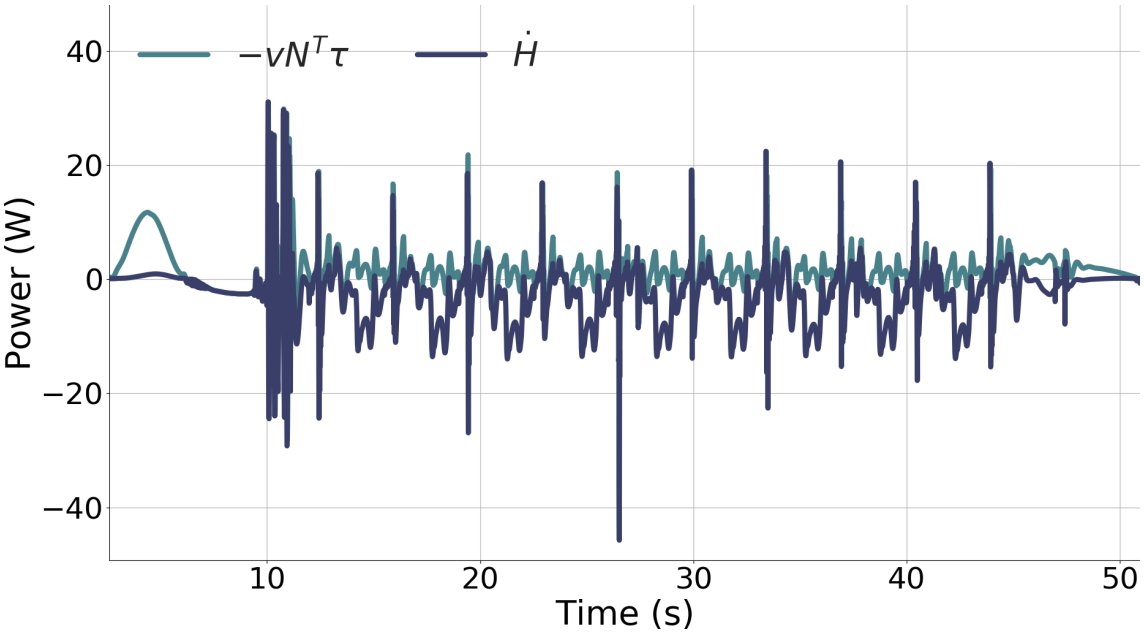

Figure 5.10: $\dot{H}$ constraint of the passive walk of 20cm steps.

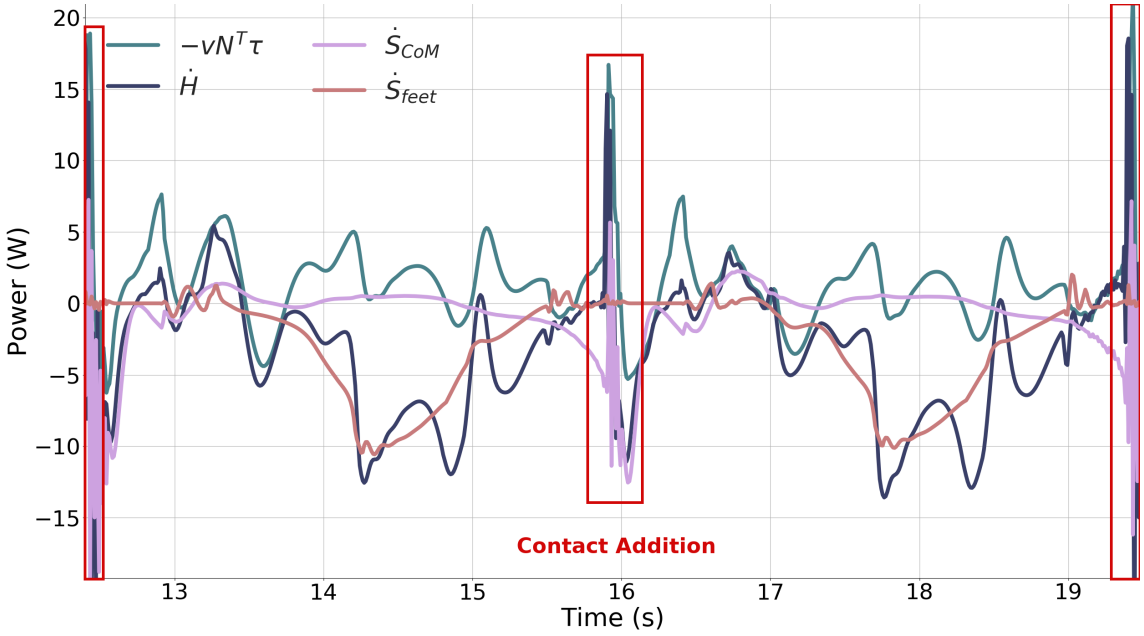

Figure 5.11: $\dot{H}$ evolution with respect to $\dot{S}_{CoM}$ and $\dot{S}_{feet}$, on the passive walk of 20cm steps.

divergence. With the passivity scheme and energy tank, the authors think that the tasks would be penalized but that the overall walking motion would be achieved.

5.5 Discussion

We first discuss the positivity of S in order to have H positive. In this paper, K_p has equal elements in function of the tasks to achieve this result. However, this is restrictive, one may want to find properties on ΛK_p to achieve the positivity with different gains [210] or to reformulate $S = \frac{1}{2}e^T K_P^{\frac{1}{2}} \Lambda K_P^{\frac{1}{2}} e$ to keep the symmetry. For the first solution, we analysed the behavior of S by changing the gains values and experimentally found that for the posture task, if the gains are equal along the kinematics chains (arms, legs, torso and head), the quadratic form of ΛK_p is positive. But for now, we do not have the proof of this result, we think that it comes from the structure of the mass matrix M which must be symmetric along the different kinematics chains of the robot. Thus, it may be possible to find a proof depending on the structures of M, J, Λ and K_p to have the positive definition of S. It would be interesting to investigate this lead as it keeps the presented mathematical formulations of E_{tank} and the proofs. For the second solution, one can reformulate S as $S = \frac{1}{2}e^T K_P^{\frac{1}{2}} \Lambda K_P^{\frac{1}{2}} e$ to make it symmetric. Then, one must replicate the changes along the set of equations and update E_{tank} accordingly. In our point of view, it should not change the overall proofs of the section.

During the first simulation, we observe oscillations on the α and γ coefficients due to the oscillations of the tool on the surface and after removing the force task. This behavior is mitigated thanks to smoothness functions Kronander and Billard [213] but can be improved. Indeed, in Shahriari et al. [113] the P_{low} value is not fixed but varies in function of the task progression, leading to a smoother γ because

it has been learnt and built in coherence with the task to achieve.

One problem of the presented formulation is to maintain the robot balance: if the coefficient β falls to 0, each task will be degraded to a zero tracking. Thus, nothing will keep the balance of the robot and it may fall. In the walking simulation, this leads to oscillations and motions of the legs that deteriorate the walk. The passive scheme has no mechanism to go back to a balancing state; in particular, when the energy consumption comes from external perturbations (such as walking on debris) it cannot handle the robot recovery. In general, to find a new solution respecting the robot equilibrium, it is necessary to re-plan new reference trajectories taking into account the disturbances. To palliate this problem, one can choose to not multiply the β coefficient on the postural task and setting the reference posture to the current one, in order to have the robot only compensates the gravity force when $\beta = 0$. However, this will prevent the postural task to be regulated by the energy tank and may lead to a non-passive behavior. Moreover, this solution is unreliable if the robot has a reference posture not guaranteeing the balance. Using a reference computed on-line by the planning to ensure the balance fix this last issue. We think that a better solution would be to design the energy tank maximum value like P_{low}, in function of the reference trajectories Shahriari et al. [113]. Then the case were β falls to 0 should be rare, only caused by dangerous external disturbances.

Another point of concern is the addition of the passivity constraint in the QP which may lead to a too constrained problem. For instance, a case may arise where the passivity needs a torque which exceeds the limits imposed on it by the QP. The two constraints will be in conflict and lead to oscillations in the solution. To solve this problem it will be necessary to take into account the constraints of the QP in the formulation of the energy tank. It may also be possible to reformulate the problem in order to compute the two stages of our formulation in Eq.5.34 simultaneously. Otherwise, it will be necessary to relax the passivity constraint. This is a limit of the passivity formulation at control level; the authors think that on-line planning can help for this kind of issue. For instance, using Model Predictive Control with a constraint on the passivity may be a future aim.

It is important to keep in mind that the passivity approach is first of all a way to ensure the stability of the system, in a global way. One can add monitoring mechanisms to detect the ill conditioned behaviors of the robot to protect the system. However, these mechanisms are dependant of previously identified dangerous cases. With the passivity stability, even cases where the monitoring processes do not detect the diverging tasks/motions/behaviors are taken into account. In particular, one can use the regulating coefficients of the energy tank in the monitoring mechanisms to improve the handling of the diverging cases.

Finally, the transition from the simulations to the real experiments is not straightforward. In the Gazebo simulator, the joint torque control is almost perfect because the dynamics of the motors are neglected. However, not taking these dynamics into account will lead to unrealistic and dangerous behaviors on the real robot. Moreover, the real robot is subject to imperfections such as errors in the chain actuation model, sensor noise, limited torque bandwidth or delays Englsberger et al. [74]. In the presented implementation, the signals retrieved from the robot are filtered and the force sensors on the feet are used to improve the robustness of the base-estimator. The authors are currently testing the simulation on the simulator

of the TALOS constructor, PAL robotics, which models the actuator dynamics of the robot.

5.6 Conclusion

In this chapter a novel control formulation for the passivation of an inverse dynamics framework is presented. It involves a global energy tank which monitors the exchange of energy between a set of tasks and regulates their gains. This method is close to the passive hierarchical impedance control relying on a strict hierarchy and null-space projection of Dietrich et al. [112]. It adds a protection valve for power flow variations and is transposed in a case of a non-strict hierarchy on an underactuated humanoid robot. The robustness of the controller is evaluated in simulations on Gazebo through a multi-contact scenario and 20cm walk involving six and five tasks. Thus, the proposed method extends the repertoire of multi-objectives non-hierarchical controllers for applications requiring physical interactions with the environment or a human. Its passive formulation ensures a stable, safe and robust behavior of the system.

For our future works, we plan to improve the behavior of the coefficients, in particular, the one on the power flow regulation can be computed as in Shahriari et al. [113], to vary in function of the task progression. The impact of these coefficients on the balance of the robot will be evaluated, indeed, for now, the robot may falls if all the system is penalized to respect the passivity. Finally, our new control formulation will be evaluated on the real robot.

In this chapter our passive torque controller implementing a contact-force task has been successfully tested in simulation on the humanoid robot TALOS. However, one can wonder how to apply this solution to an industrial context and industrial robots such as the MSDR. The transfer of the solution is not straightforward, as stated in the context of this thesis (see Section 1.4.3), the MSDR has no ROS interface, thus one will be needed to plug our solution to the low-level controller of the robot. The transfer of our state-of-the-art controller onto the MSDR industrial robot is discussed in the next chapter.

Chapter 6

Adapting the solution to industrial context

Contents

6.1 Introduction

This chapter describes the challenges that may arise when we will transfer the passive torque controller presented in the previous section on an industrial robot. The industrial robot MSDR considered has been presented in Section 1.4.3, as it is the solution designed by Airbus to achieve autonomous manufacturing operations (see Fig.6.1).

As stated before, the first issue to tackle is the difference of model between the robot TALOS and the MSDR: one is under-actuated while the other is not because it is fixed to the ground (for the initial setup considered in this thesis). Thus, the relevant part of our control solution can be seen as the application of a passive contact-force task on the robot arm. Using our formulation of the passivity, the control formulation will guarantee safe, stable and robust behaviors of the robot onto the environment. One can thus formulate a new set of tasks in our previous torque controller to take into account only the relevant parts of the complex problem presented on the humanoid robot.

Secondly, the MSDR uses the Fanuc proprietary software architecture which does not implement a ROS interface. The difficulty here lies in reading sensors and writing commands (joint position or torque vector) within a guaranteed deterministic time boundary in order to update the system information in real time. The frequency needed to achieve a correct behavior is at least 250Hz, our torque scheme achieves a control loop at 1kHz. Thus, one will need to implement an interface to connect the low-level control board of the MSDR to our high level solution.

Once these issues are solved, one can highlights the benefits and drawbacks of our state-of-the-art solution compared to the classical proposal for manipulator robots. The proposal on how our solution will work on the MSDR is done on a drilling process for a fuselage metal plate.

This Chapter is organized as follows: The first Section 6.2 details the change to be made in the formulations of the robot model, controller set of tasks and control law and how it will affect the results. Then, Section 6.3 presents an abstract method to create the interface between the robot low-level and our controller. Finally, the application proposal of our passive scheme compare to a classical drilling process is done in Section 6.4

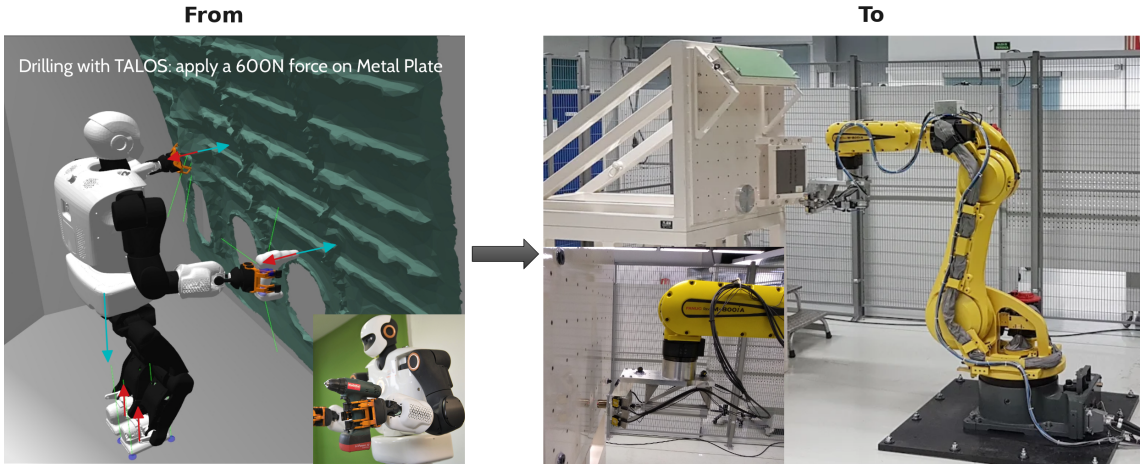

Figure 6.1: How to transfer the solution from TALOS to the MSDR ?
Example of the process of drilling a metal plane.

6.2 Whole Body Control Formulation

6.2.1 Robot Model

Because the MSDR is fixed to the ground, it has no under-actuated part. It can nonetheless be put on a forklift truck to be moved in the fuselage cell. Then, a state estimation is necessary to check the base position of the robot and correct possible errors by updating the frame placements. In some specific setup, additional auxiliary axis mounted within the end-effector might be considered as source of under actuated parts. They are not considered here as the concept behind the MSDR is closely related to simple end-effector without additional axis modifying its absolute position.

Consequently, in this chapter, the variables q, $\dot{q}$ and $\ddot{q}$ will be used to refer to the joint position, velocity and acceleration, respectively (as there is no under-actuated part).

The MSDR robot has stiffer joints (by design) than traditional industrial robots, as opposed to the flexible manipulators of Albu-Schäffer et al. [55]. Then, the equation of the robot dynamics (Eq.2.4) can be re-written as:

$$M(q)\ddot{q} + \underbrace{C(q,\dot{q})\dot{q} + g(q)}_{h} = \tau + \underbrace{\tau_{ext}}_{J_c^T F} \tag{6.1}$$

with $M \in \mathbb{R}^{n_j \times n_j}$ the symmetric and positive definite inertia matrix, $C \in \mathbb{R}^{n_j \times n_j}$ the Coriolis matrix and $g \in \mathbb{R}^{n_j}$ the gravity vector. $q \in \mathbb{R}^{n_j}$ is the the Denavit-Hartenberg Bejczy [28] representation of the robot joint angles. $\ddot{q}, \dot{q} \in \mathbb{R}^{n_j}$ are the accelerations and velocities of the joint configuration of the robot. $\tau \in \mathbb{R}^{n_j}$ are the joint torques of the actuators and $\tau_{ext} \in \mathbb{R}^{n_j}$ are the external torques.

The main difference from the humanoid robot formulation is that the accelerations a is replaced by the joint accelerations $\ddot{q}$ and thus the section matrix N disappears.

Tasks	Priority	Weight
End-Effector contact	I	100
Cartesian Force-Contact	I	50
Posture regularization	I	0.1

Table 6.1: Set of tasks that can be implemented for the MSDR using TSID.

6.2.2 Set of Tasks

As explained in the Section 1.5, the stack of tasks used for a humanoid robot and for manufacturing robots are similar (see Fig.1.9). Because the base of the robot is fixed, one can neglect to add a task keeping the base joint at the same position. Moreover, controlling the CoM of classical robot manipulators is usually not a major concern. Indeed, for robot manipulators, static stability is ensured since they are fixed to the ground and that the gravity effects on the joint torques are generally compensated for, either actively or using passive mechanical devices Albu-Schäffer et al. [55], Schindlbeck and Haddadin [125]. Unfortunately, most of the time these compensations are implemented by the industrial robots manufacturers without providing access to their algorithms or intrinsic parameters.

The order of priority of the tasks is set using their weights, indeed, each task is put in the cost function (see Table.6.1). One can set the contact task as a constraint but it can be avoided as we will implement additional constraints on the force and it can over-constrain the problem. Nonetheless, the contact task is set as the higher priority (with the higher gain) as it is necessary to remain in contact with the plate to drill and the position precision is important. Then, the force task is set as the second most important task to follow accurately the force profile in order to successfully drill the plate. Finally, a posture regularization task can be added to control the remaining DoFs of the robot. Table.6.1 presents the tasks that seem appropriate to implement to realize a force manufacturing operation.

The end-effector contact task is necessary to control the position of the cutting tool while drilling. Indeed, even if the robot is quite stiff, spring effects can still appear in the joints of the robot and the robot parts can wrap while applying huge forces. Moreover, spring back effects can be propagated due to the mechanical properties of the aircraft part material. This can lead to deflect in the position of the end-effector. In the actual industrial process, two strategies are implemented to tackle this issue unplanned displacement: the use of a clamping force and slippage threshold.

- ▷ **Clamping force:** To avoid any unexpected movement during drilling operation the end effector must be stuck against the aircraft part all along drilling cycle. To achieve this, the robot must apply a clamping thrust against the aircraft part. The clamping force is dependant on the stiffness of the surface. However, in the process of clamping, the robot can slip over the surface.

- ▷ **Slippage avoidance:** As stated Eq.2.11, to avoid the end-effector slippage along the parallel axis of the surface, it is necessary to have a the norm of the perpendicular force greater or equal than the norm of the normal one times the friction coefficient of the surface. However, this formulation leads to huge

application forces which can damage the surface and/or the end-effector. Thus an empirical slippage threshold is chosen in practice.

Therefore, the end-effector contact task will be modified to take these implementations into account as constraints in the QP formulation (replacing the Coulomb one). Note that this task still implements the motion task of Eq.4.8 in the cost function to control the position of the end-effector. Moreover, the reference position of this task can be computed using the visual information of the robot (cameras, lasers) to adjust locally the exact position of the hole.

6.2.3 Quadratic Programming Formulation

The formulation of the QP in TSID is modified a little for the MSDR. Using the passive controller of Section 5.3.3, we can adapt the solution to a fully actuated robot with the previous contact constraints. The obtained QP, for the $i \in \{0, .., 3\}$ previously defined tasks and their weights λ_i is expressed as follows:

$$
\begin{aligned}
\text{Find} \quad & \alpha, \beta, \gamma, \dot{\mathbf{H}} \\
\text{s.t.} \quad & \text{Eq.5.18} - 5.23, 5.26 \\
\min_{y=[\ddot{q}]} \quad & \sum_{i=0}^{3} \lambda_i \left\| O_i y - \beta \gamma o_i \right\|^2 \\
\text{s.t.} \quad & \begin{bmatrix} J_c & 0 & 0 \\ M & -J_c^T & -N^T \end{bmatrix} \quad \begin{bmatrix} \ddot{q} \\ F \\ \tau \end{bmatrix} = \begin{bmatrix} -\dot{J}_c\, v \\ -h \end{bmatrix} \\
& \tau_{min} \leq \tau \leq \tau_{max} \\
& F_{min} \leq F \leq F_{max} \\
& |f_{slip}| \leq |F_{\perp}^{EE}| \leq |f_{clamp}^{max}| \\
& q_{min} \leq q \leq q_{max} \\
& \dot{q}_{min} \leq \dot{q} \leq \dot{q}_{max} \\
& \dot{\mathbf{H}} \leq -v^T N^T \tau
\end{aligned}
\tag{6.2}
$$

with F^{EE} the force at the end-effector, f_{clamp}^{max} and f_{slip} the max clamping force and slippage threshold defined previously. We add constraints on the angular limits in position with q_{min}, q_{max} and in velocity $\dot{q}_{min}, \dot{q}_{max}$. Then the free variables of the problem are the joint accelerations $\ddot{q}$. The equations 5.18-5.23 and 5.26 can be solved for the MSDR robot as well has for any humanoid robot, one just has to replace the v variable by $\dot{q}$ and to remove the N matrix. Indeed, because there is no free-flyer the velocity vector v only contains q (the other variables are null) and there is thus no need of the selection matrix N. The calculations remain the same otherwise and lead to the same result for the energy tank.

6.3 Interface between the robot low-level and our controller

Once the problem is re-formulated for the MSDR, the QP can compute a torque command respecting the passivity of the system. However, it needs the robot actuators angular position and velocity and the forces measured at the end-effector level. This data must be retrieved in a timely manner to be able to execute the QP in real time. In the case of the MSDR, to have access to this data it is necessary to use the Fanuc software.

In the control scheme implemented on the TALOS robot, the data transfer is performed by the *ros-control* framework of the middle-ware ROS Adolfo Rodríguez Tsouroukdissian [19]. Recalling the scheme of the Fig.1.10, the roscontrol-sot package can communicate with the hardware interface of *ros-control* to retrieve the low-level information. Thus, to solve the issue of communication between the hardware of the MSDR and our control scheme, one can create the ROS robot hardware interface of the MSDR to communicate with the Fanuc software and retrieve the information. Tutorials exist online to implement your own hardware interface, it is however important to notice that the data of the MSDR can be internal sensors embedded within the robot (encoders, current, force sensors) or external sensors of the end-effector (lasers for normality, cameras, IMU, force sensors).

One issue that may arise is the importance of the delay between the reading of the sensors/actuators data by the Fanuc software and its transmission to the interface and to the controller. And vice-versa, if the delay between the command sent by the controller toward the Fanuc software and its application on the robot is too important it can lead to precision issues. One has to notice that in any case, the communication within the robot manufacturer's controller interfaces is mainly best effort and not real-time ($\sim$ 250Hz), as it is not the prioritized process. However, to reduce the delays, it can be appropriate to have the Fanuc software implements a quick regulation control loop which takes the high-level controller input at a low frequency depending of the delay (for instance 500Hz). Its role will be to regulate the high-level commands for the low-level until reaching the defined time to read the new high-level control input. This regulating controller can for instance be achieved by a PD+ (as the controller used by the PAL-Robotics company) or a Regulation Sensitivity and Tracking (RST) polynomial controller (more common in industry for digital control). The RST is a robust control able to track reference trajectories as well as to regulate the control during disturbances. The PID is a particular form of RST, with a particular choices of the polynomials R, S and T.

In Fig.6.2 is presented an example of complete solution incorporating all the elements from the planning to the control of the robot. In yellow are depicted the the visual processes to detect the holes (object recognition) and localize the metal plate and the robot (using for instance the Simultaneous Localization And Mapping (SLAM) approach). In red are presented the planning algorithms, which follows the stages detailed in Fig.2.7: first plan the motion, then find the contact poses and finally optimize the trajectory with a MPC. The motion can be decomposed in several sub-motions or states, which can be organized in a finite-state-machine. The state will present different constraints or tasks such as the presence or absence of contact for instance. The current state is thus given to the whole-body controller

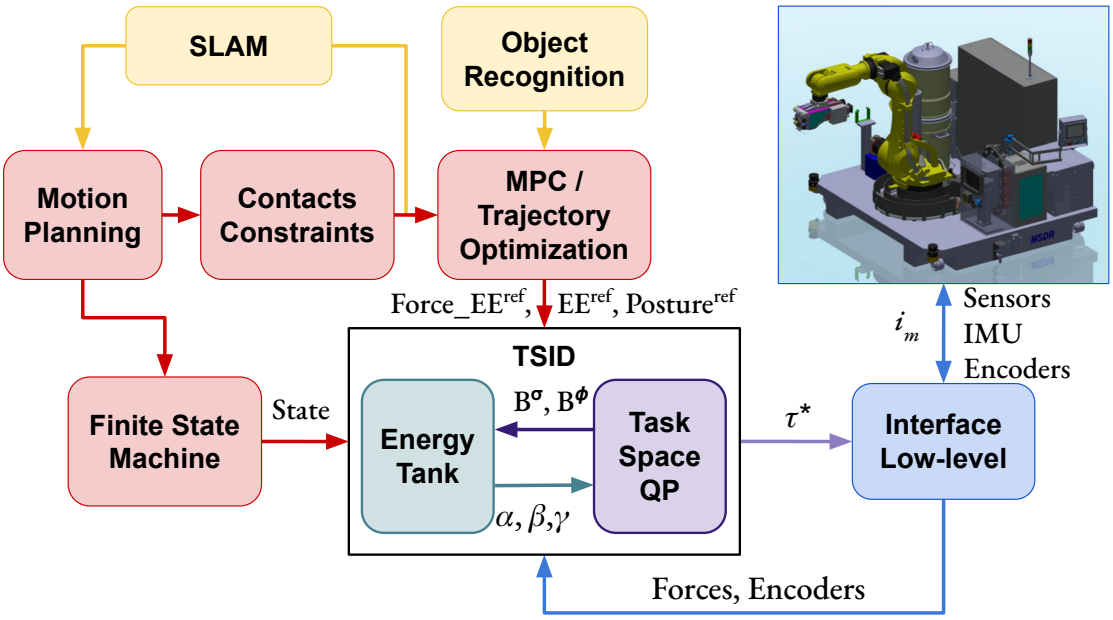

Figure 6.2: Whole control scheme for the MSDR using our solution.

to implements the appropriate set of tasks and constraints to achieve the desired motion. The controller is in the black rectangular box in the figure, with the characteristic necessary to ensure the passivity of the system (energy tank). Finally, the interface with the Fanuc software and the low-level controller is presented in blue between the robot and the controller.

6.4 Drilling Process Solution

Here we present a typical drilling process on a fuselage. Between two reference pre-holes ($R1, R2$, pre-hole means that the holes have not the final drill diameter) drilled by an operator, ten holes are performed by the machine, with a reference axis frame based on $R1$ and $R2$ (see Fig.6.3). The drilling process follows the steps detailed in the Table.6.2, illustrated in the Fig.6.3.

The position and orientation adjustments are done with the vision system and the normality correction with the lasers embedded in the end-effector. The current

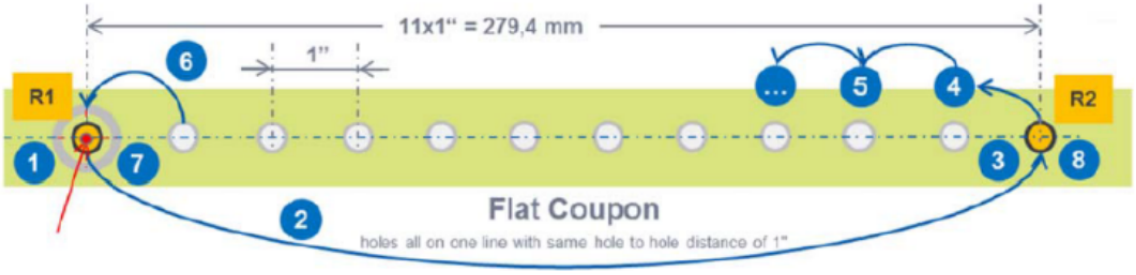

Figure 6.3: Drilling operation with 10 holes between two reference holes ($R1, R2$).

Step	Description
0	Initial position: MSDR moved to R1
1	Adjust the position and check the normality between the TCP and the surface
2	Move the TCP to R2
3	Repeat step 1 for R2 and move to 1st hole
4	Drill 1st hole
5	Drill remaining 9 holes with step 4
6	Move the TCP to R1
7	Repeat step 1 and drill R1 to final diameter
8	Move to R2 and repeat step 7 for R2

Table 6.2: Set of steps performed by a manipulator robot for drilling operation.

robot program is a point to point offline path planning executed without modifications at run time (outside of the offsets adjustments on $R1, R2$). This nominal process correspond to a *coordination between tasks and resources* (robot, humans and fuselage) which is *sequential*. When a problem arises, the recovery process is typically the following:

- ▷ The machine has failed to perform the task with the requested quality
- ▷ An operator needs to apply a repair process, before the machine restarts, or at the end of the machine process.

Which means that, as presented in the context section, the operator and the robot cannot currently work together, the machine has to be stopped for an human to act, and failure in the machine program stop the process. Moreover, in order to restart the robot production process, its program must be modified and re-validated offline. This leads to delays in the production chains.

The targeted goal is to achieve a *concurrent coordination*, where the robot can adapt from the planning if an error occurs to correct its task or skip the problematic hole completely to continue the process (even onto another part of the plate). Then, the operators can fix the issue while the robot continue the drilling process. It can be resumed as follows:

- ▷ All points are considered in the robot program and on operators tasks backlog (to be fixed only if the robot does not succeed to complete the task)
- ▷ When the robot moves to a point where tasks are performed, it skips the task to the next point or to the next subgroup
- ▷ If the robot is not able to realize its task with success, the task is re-planned or aborted and skipped, and the robot continue to the next target.

With the proposed solution of Fig.6.2, involving visual feedback, planning, finite-state machine and reactive control, it is possible to realize this targeted *concurrent coordination*.

Another point of interest is the *safety* of the robotic solution, indeed, with the concurrent solution (and even with the sequential one), the operators can perform manual tasks very close to the robot. Thus, the collision shall be avoided to ensure human safety. For now, the solution implemented is to progressively decrease the speed of the robot when the robot get closer to humans or the fuselage. What can be used in our proposed solution is the reactive planning of "collision free" path that can be computed on demand, while executing the initial path. Moreover, using our passive torque controller allows to benefit from the natural compliance of torque control and from the passivity of the system. This means that, subject to unplanned disturbances, the control will be penalized to ensure the safety of the whole system (environment, humans and robot), see Chapter 5. Thus, the tasks performed will be degraded or aborted in favor of the security. However, the qualification of this *safety* process for production will be difficult.

6.5 Conclusion

In this chapter is presented the two challenges that we think may arise when we will transfer our passive torque controller onto the MSDR. The reformulation of the optimization problem to match the robot fully-actuated model and new force constraints has been presented. Then, the issue of interfacing the hardware and low-level control of the robot to our solution has been discussed, a ROS hardware interface between the Fanuc software and our control scheme is possible. Finally, a proposal of complete control scheme using some of the Gepetto team works and the passive controller has been presented. Its application on a drilling process has been explored and the benefit of it stated. In particular, our solution improve the adaptability, reactivity and safety of the robotic solution.

The future works consist in implementing the proposed solutions on the MSDR robot and test them on the drilling process. This will however require a consequent amount of time to create a self-sufficient "tool-box" from the Gepetto team software. Moreover, the certification of the programs termination (analysis of time complexity) and robustness is an important security process for Airbus, which is not currently achievable. Despite this blocking point, Airbus is interested to transfer the developed solution of this thesis on the MSDR. A first step could be to directly implement a simple passive PD+ control using the Fanuc software. A second step may be to implement the complete solution as a "tool-box" in the scope of the joint-lab ROB4FAM.

Conclusion

Summary

This thesis aims to answer the issue of robotics whole-body control for manufacturing operations. During the development of my solutions, two major objectives had to be considered, the safety of the operation for the robot and for its environment, and the capabilities of the whole-body controller to achieve force application while keeping its balance.

A state-of-the-art is first presented in the Chapter 2. It details the robot model and centroidal dynamics formulation used in my thesis, the different type of control designed on humanoids robots for locomotion and manipulation scenarios and the stability criteria considered in my works.

Secondly, a process for identifying and modeling the elbow chain actuator of the robot TALOS is described in the Chapter 3. This model and the identified parameters are used to design a low level current controller to protect the robotic system using a MPC. The process of identification can be applied on any rigid chain actuator (for any robot) to retrieve its inertial parameters, it is only subject to correctly model the chain actuation. Even if the MPC is computationally slow, it allows TALOS to carry high payload (up to 34 kg on one arm) while realizing motions and protects the system.

The Chapter 4 details the three whole-body controllers implemented in this thesis, two in position and one in torque. Their respective tasks functions and formulations are presented to achieve locomotion and manipulation scenarios. This chapter compare them in simulations on complex locomotion problems such as stair climbing and walking on non flat terrain. These simulations have yet to be tested on the real robot using the modeling of the hip flexibility of TALOS. The intermediate steps performed on the real robot are detailed as well as the blocking points preventing us to successfully achieve the experiments. Because the torque controller has proven to be energy efficient, capable of quite high speed locomotion and adaptable to uncommon scenarios (such as navigating in collaboration with a human, while using similar gait); I chose to focus on this solution for the next developments of the thesis.

Yet, using this controller on the real TALOS robot leads to instabilities in the solution on the simplest scenario. Thus, to realize the objective of safe operations, stability analysis and in particular the passivity analysis methods are investigated in the Chapter 5. I implemented a novel formulation for my torque control scheme to obtain a passive system (a dissipative system), ensuring a stable, safe and robust behavior of the system. My whole-body control scheme embeds an energy tank

monitoring the exchanged power in the system. The tank regulates the controller tasks gains in function of the input-output system power. This new scheme is tested in two simulations. First on a multi-contact scenario employing a parallel position/force task as a first step for manufacturing operations. Secondly on a simple locomotion scenario, a walk of 20cm steps on flat ground.

Finally, the Chapter 6 presents the expected difficulties to consider when transferring the passive solution from the humanoid robot TALOS to the manipulator robot MSDR. The two main aspects to consider are the reformulation of the optimization problem to match specific force constraints and the creation of an interface between the Fanuc software and my controller. This chapter also highlights the benefits that my solution and, in a more global approach, the planning and visual methods of the Gepetto team, can bring. The team hopes to improve the adaptability, reactivity and safety of the industrial robotics solution.

Future Work

Identification and Modeling of the Whole Chain Actuation

The extension of the work presented in the Chapter 3 would be to identify all the actuators of the robot and to implement the solution over high-performance dedicated and embedded electronics board attached to each actuator of the robot. However, due to the sanitary crisis this identification process has been delayed.

Moreover, with the model and parameters of each of the actuator it will be possible to implement a torque controller mapping the desired joint torques of the QP controller to motor currents and send them directly to the robot. This way, there will be no need to use the PAL-Robotics company low-level controller to try my simulations on the real robot. For now, this proprietary controller is a black box and I encounter some problems to interface my complete solution with it even if it uses a *ros-control* architecture, as presented in Section 4.6. This is why, currently, even the simulation of the contact-force has not be tested on the real robot, while the flexibility of the robot is not a blocking point. It is a known difficulty of torque control implementations to successfully transfer the simulation results on the real robot. I should have started to work on the interface with the PAL-Robotics low-level controller sooner to solve this problem. Or, it might have been quicker to try to create a simple PD+, validated in the PAL-Robotics simulator, to transform my desired joint torques to motor currents and completely bypass the black-box controller.

Interfacing my Controller with the Flexibility Compensation Model

To finalize the work presented in the Chapter 4 and test the walking simulations on the real robot, it would be necessary to implement the flexibility compensation of Villa et al. [206] in my controllers. In this paper, the hip bending is estimated from the commanded torque and the elastic deflection (θ) is added to the measured hip configuration q_{hip}. The same is done for the derivative of the deflection. The new values of the hip configuration and joint velocities are then replaced in the coordinate and velocity vectors q and v and are used by the base-estimator and the controller. Because the implementations done in Villa et al. [206] have been made using TSID, it should be easy to join my controllers with them and then test the simulations on the

real robot. Be aware that for torque control, the previous point still holds and the problem with the PAL-Robotics low-level controller must be solved. Furthermore, in Villa et al. [206], the modified coordinate and velocity vectors q and v taking into account the deflection are also used in a locomotion planner to compute coherent reference trajectories. An interesting future step would be to use this planner in tandem with my controllers to adequately take into account the hip flexibility of the robot TALOS.

Finally, with the modified torque controller using the flexibility compensation, one would want to continue the collaboration with Isabelle Maroger on the ANR-COBOT project. The aim would be to test the whole developed framework for human-robot co-navigation in real time on the robot TALOS. In particular, using the contact-force task added in my torque controller and the passivity implementation, it would be amazing to have TALOS successfully carry and move a table in collaboration with a human. Thus, the stability of the system would be guaranteed and it would be possible to control the amount of force the robot has to apply on the table (currently only the control of the robot hands positions is possible).

Improving the Energy Tank Shaping and Regulating Coefficients

To improve the results obtained in the Chapter 5 on the passivation of the control scheme, one would revise the definitions of the budget of the energy tank and of the regulating coefficients. Indeed, for now, the robot may falls if all the system is penalized to respect the passivity, because the coefficients β, γ are acting on every tasks. However, as discussed before, the passivity proof will not hold if these coefficients only act on some of the tasks. One can try to implement an energy tank for the tasks responsible of the robot balancing, such as the CoM, the feet contacts and the posture ones and another tank for the other tasks. Then, if the activation of the system does not come from the balancing tasks, they will not be penalized and the robot would keep its balance.

Moreover, it would be interesting to design the budget and the maximal value allocated to the energy tank in function of the reference trajectories. The upper bound of the tank might then increase or decrease in function of the planning. For instance, while moving on flat floor the robot does not need too much energy, but if it has to climb an obstacle the necessary amount may rise. Thus, one would increase the maximum value of the tank a little before-hand to store energy to anticipate the climb. Similarly, the coefficient on the power flow regulation γ can be computed as in Shahriari et al. [113] to vary in function of the task progression, close to what I described before for the maximal value of the tank, using the reference trajectories. Finally, one would want to use the regulating coefficients of the energy tank in higher level monitoring mechanisms to improve the handling of the diverging cases.

Reformulating the QP Formulation or Developing the Passivity Criterion on a MPC

Another way to improve the results obtained in the Chapter 5 would be to reformulate the alternate minimization problem of Eq.5.34 to solve the energy parameters concurrently to the complete optimization problem (subject to constraints). Then,

the passivity as a constraint in the QP will not be necessary as the energy parameters will be optimized on the same problem formulation than the QP (including the constraints of the system). For now, the addition of the passivity constraint in the formulation can lead to a too constrained problem and it is possible that it enters in conflict with another constraint and lead to oscillations in the solution.

A second possibility could be to implement the passivity criterion in a MPC, for instance the one developed by Dantec et al. [147] which has achieved great results recently. The passivity can be implemented by using the value function of the Optimal Control as the storage function and constraining it. Or it is possible to reformulate the problem and to add a sate constraint over the trajectory to respect the passivity inequality Raff et al. [140].

Testing the Complete Approach in a Factory

Finally, one of the future goal would be to bring the TALOS and TIAGo robots to an Airbus factory and test in real conditions the overall approach including localization, visual-servoing and control. This aim was first expected by the end of my thesis, but due to the sanitary crisis, we accumulate delays and could only successfully test the localization and navigation stack of the robot TIAGo. Once some of the previous points are fixed, one would try to demonstrate the developed solutions in the industrial environment. This is a future goal for the ROB4FAM joint-lab, which could be achieved before or in parallel with the implementation of the results on the MSDR.

Appendices

Appendix 1: Complete Scheme of the Walking Pattern Generator and the Dynamic Filter

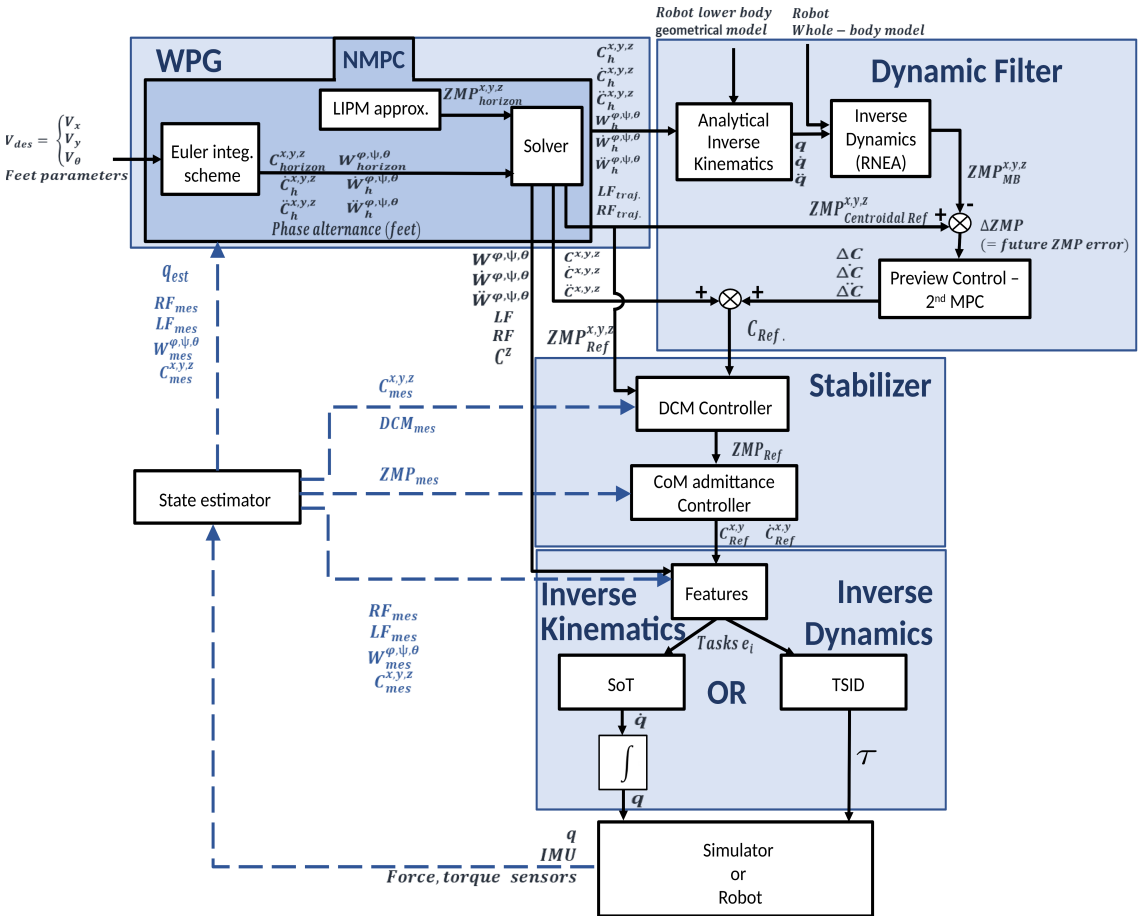

Figure 6.4: Detailed Scheme illustrating the Walking Pattern Generator and the Dynamic Filter Scherrer [1].

Appendix 2: Differential Dynamics Programming Algorithm

Algorithm 1 DDP Solver

1: **function** DDPSolver(x_0, U_0)
2: $U \leftarrow$ random $U_0 = \{u_0, ..., u_{T-1}\}$
3: $X \leftarrow \{x_0, U\}$
4: $\Delta V \leftarrow value > \epsilon$
5: **while** $\Delta V > \epsilon$ **do**
6: $V(T) \leftarrow l^T(x_T)$
7: $V_x(T) \leftarrow l_x^T(x_T)$
8: $V_{xx}(T) \leftarrow l_{xx}^T(x_T)$
Backward Phase
9: **for** $i = T-1;\ i \leq 0;\ i = i-1$ **do**
10: $Q_x \leftarrow l_x + f_x^T V_x^{i+1}$
11: $Q_u \leftarrow l_u + f_x^T V_u^{i+1}$
12: $Q_{xx} \leftarrow l_{xx} + f_x^T V_{xx}^{i+1} f_x + V_x^{i+1} f_{xx}$
13: $Q_{uu} \leftarrow l_{uu} + f_u^T V_{xx}^{i+1} f_u + V_x^{i+1} f_{uu}$
14: $Q_{ux} \leftarrow l_{ux} + f_u^T V_{xx}^{i+1} f_x + V_x^{i+1} f_{xx}$
15:
16: $\Delta V^i \leftarrow -\frac{1}{2} Q_u Q_{uu}^{-1} Q_u$
17: $V_x^i \leftarrow Q_x - Q_u Q_{uu}^{-1} Q_{ux}$
18: $V_{xx}^i \leftarrow Q_{xx} - Q_{ux} Q_{uu}^{-1} Q_{ux}$
19:
20: $k^i \leftarrow -Q_{uu}^{-1} Q_u$
21: $K^i \leftarrow -Q_{uu}^{-1} Q_{ux}$
22: **end for**
Forward Phase
23: $\hat{x}(0) \leftarrow x_0$
24: **for** $i = 0;\ i \geq T-1;\ i = i+1$ **do**
25: $\hat{u}(i) \leftarrow u(i) + \alpha k^i + K^i(\hat{x}(i) - x(i))$
26: $\hat{x}(i+1) \leftarrow f^i(\hat{x}(i), \hat{u}(i))$
27: **end for**
28: $\Delta V \leftarrow \|\Delta V^i\|$
29: **end while**
30: $U \leftarrow \{\hat{u}(0), ..., \hat{u}(T-1)\}$
31: $X \leftarrow \{x_0, \hat{x}(1), ..., \hat{x}(T), U\}$
32: **return** $\{X, U\}$
33: **end function**

The upper-scripts of $l, f, k, , K, V$ correspond to the iteration numbers ($i = 0$ to T), while for x, u this number is in subscript.

The backward pass is initialized with the value function set to the terminal cost and its derivatives (l.$6-8$). Then Q is filled according to the Jacobian and Hessian of its pseudo-Hamiltonian form (l.$10-14$), and V updated (l.$16-18$). The terms k and K correspond respectively to the feed-forward and feedback terms of the optimal control local policy updating u (l.$25-26$) defined by Eq.2.78.

Appendix 3: Frameworks of the different controllers

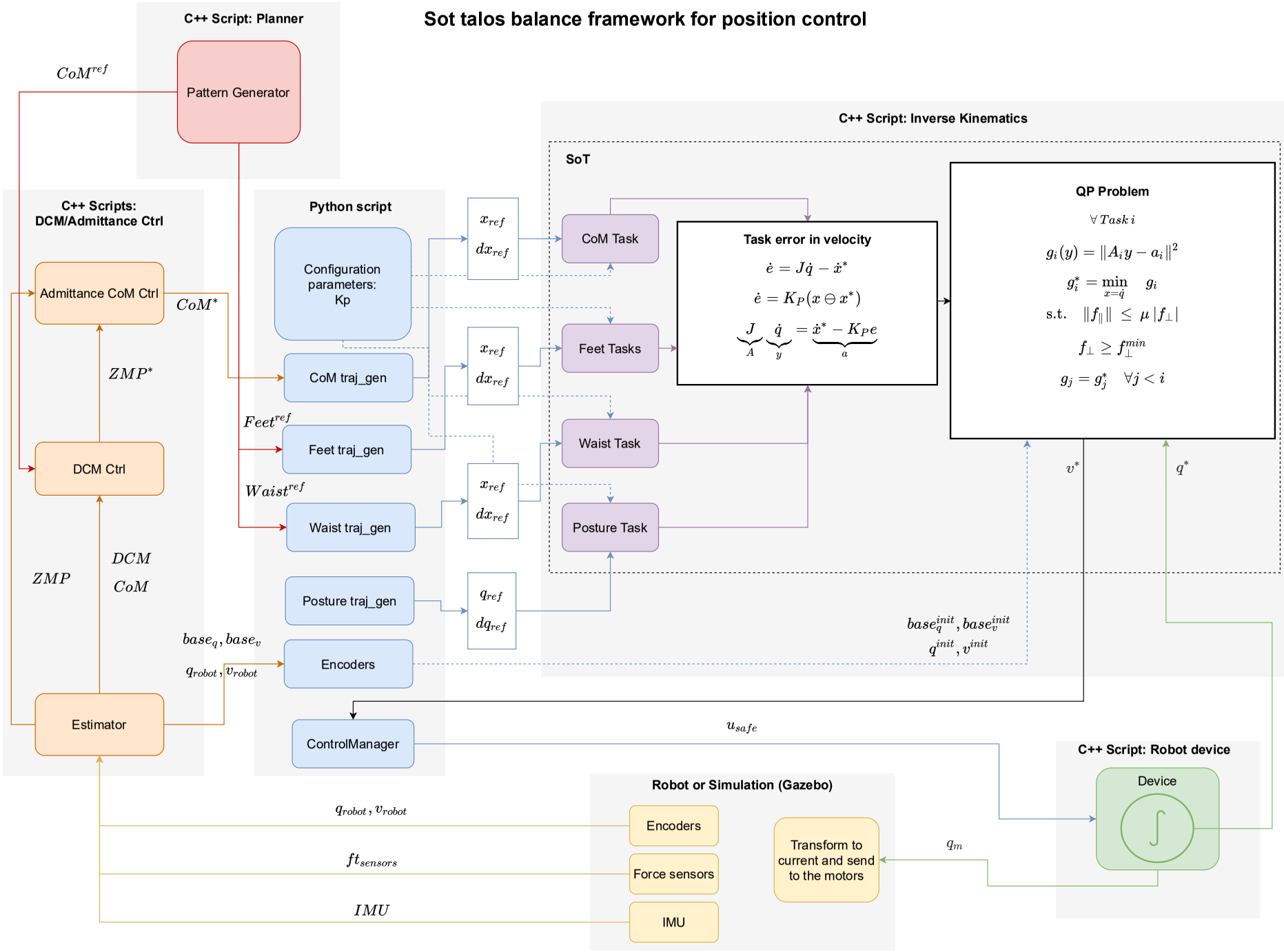

Figure 6.5: Framework of sot-talos-balance in position mode.

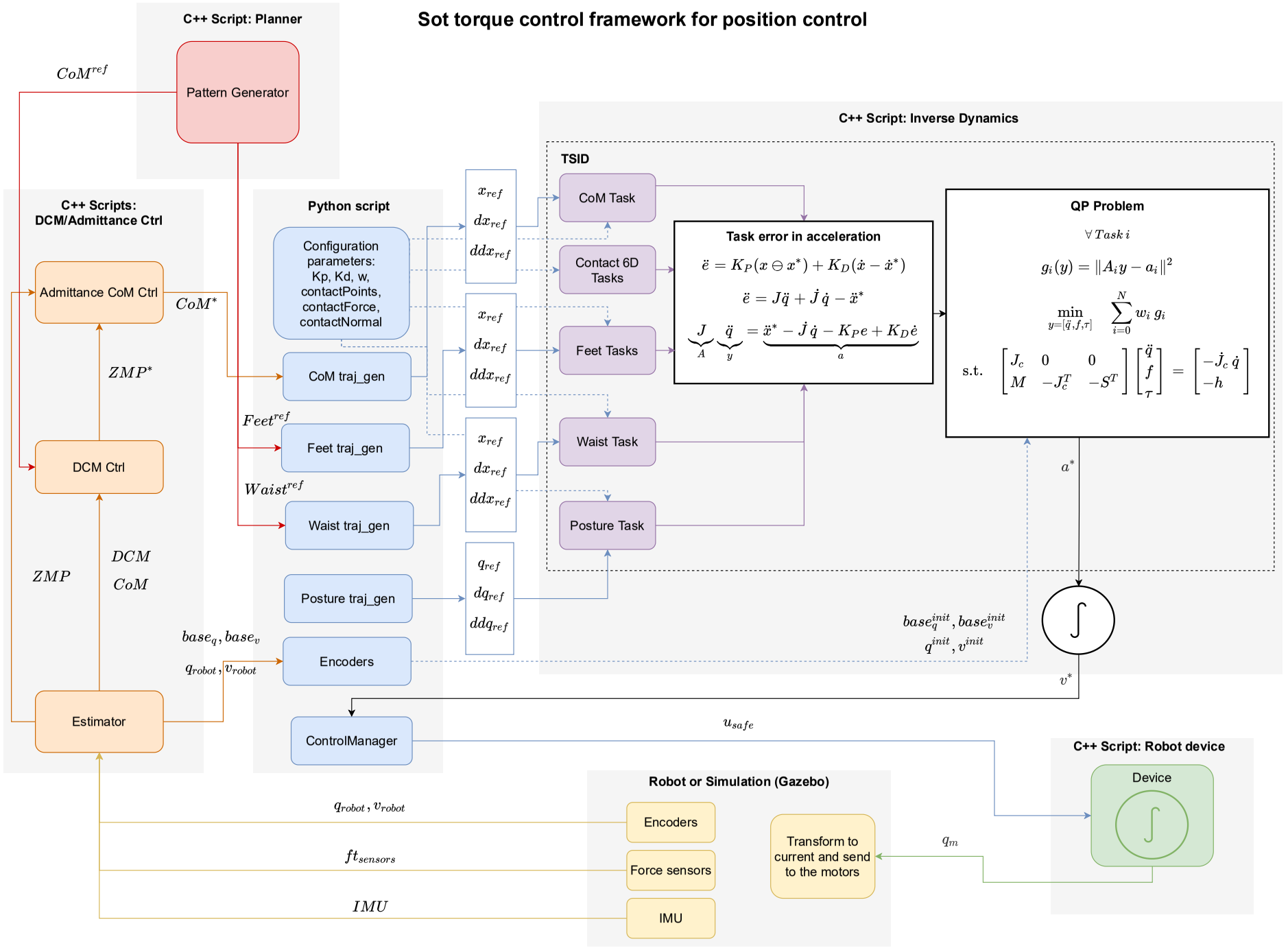

Figure 6.6: Framework of sot-torque-control in position mode.

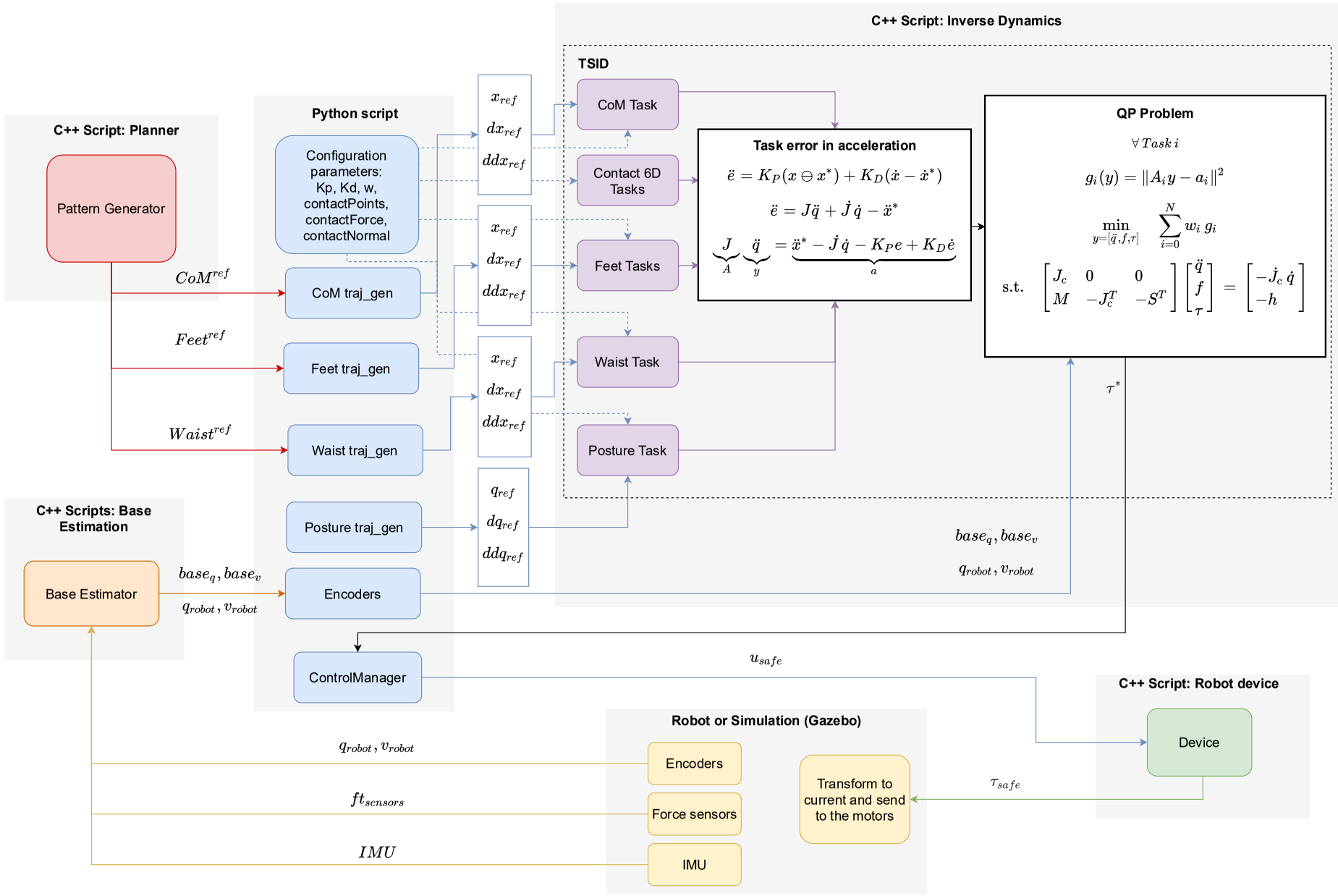

Figure 6.7: Framework of sot-torque-control in torque mode.

Appendix 4: TALOS Hip Flexibility Identification and Control

As presented in Sec.1.4.1.a, the robot TALOS presents a flexibility at its hip. In this section is presented the solutions implemented in this thesis to try to compensate it while using a position controller.

Flexibility Identification

In this section is described the first experiment realized to identify the flexibility caused by the supporting structure of the TALOS hip and not only by the actuator compliance (gearbox, torque sensor ...). This flexibility is located not only at the hip roll-pitch, but also on the yaw, which means that part of it can not be observed with the joint encoders. To estimate the flexibility, the feet distances were collected with a motion capture system. The robot was rigidly controlled in position, and by applying forces on the feet, the whole legs were deflected during the experiment. The experiment is described in Fig.6.8, where the flexibility on the hip is denoted K (1 or 2 in function of the leg side), d (in mm) is the measured feet distances between what is measured by the motion capture and the feet positions calculated with direct kinematics. α is the angle of the deflection, τ is the torque measured by the sensors. We applied manually a force F to bring the feet closer and record the torques sensors measurements, the feet distances collected by the motion capture and the feet positions calculated with direct kinematics.

The flexibility is calculated under the following set of hypothesis:

- ▷ The torque at the hip attachment is approximated to the roll-pitch torque measurement at the sensor, since the level arm between the two is low compared to the leg. Thus $\tau_{K_1} \approx \tau_1$ and $\tau_{K_2} \approx \tau_2$.
- ▷ The flexibility on both legs are approximated to be the same: $K_1 \approx K_2$, $\alpha_1 \approx \alpha_2$.

We noticed during this experiment that the obtained distance between the measured and computed feet positions d is proportional to the measured torques $(\tau_1 + \tau_2)/2$. We thus performed a linear regression to find the slope parameter linking the measured torques to the feet errors. This gave the estimated hip flexibility around the X-axis to 1.027 mm/Nm which corresponds to 973.201 Nm/m. Approximating the leg of TALOS to 1m long (level-arm), it corresponds to a torsional stiffness of 973.201 Nm/rad.

Flexibility Control

Without Model

Using the previous estimated flexibility, it is possible to compute the hip deformation not measured by the encoder. This deformation Δq^{hip} can then be used to modify the position control at the hip pitch and roll joints to take the flexibility into account.

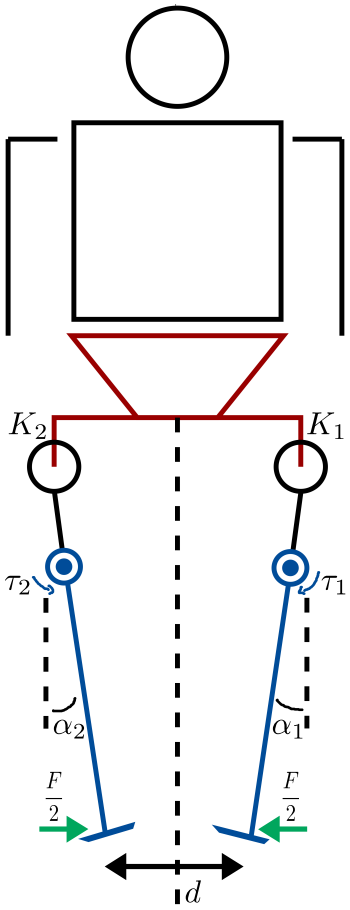

Figure 6.8: Scheme to illustrate the identification of the flexibility

The new position command is then defined by:

$$\begin{aligned} \Delta q^{hip} &= \frac{\tau^{hip}}{K} \\ q^{hip}_{cmd} &= q^{hip}_{des} + \Delta q^{hip} \end{aligned} \tag{6.3}$$

However, the noise in torque sensors requires filtering, which introduces important delays. Unfortunately, previous experiences with such a technique carried out with the robot gave poor performances. Only a repeatable one step walk experiment on TALOS in position control was achieved using fixed flexibility compensation (see Section 4.6). In torque control however, this issue is mitigated because the flexibility is considered by the control system as external disturbances. Nonetheless, to achieve robust experiments, it will be necessary to take into account this flexibility which cannot be compensated without a proper identification and modeling.

With Model

A second method has been tested to control the hip flexibility in Villa et al. [206], relying on the use of a specific model for the flexibility. It uses the standard model

defined by Nakaoka et al. [214], introducing passive joints in the waist-leg connection, where the link cross-section is reduced, as the link deflections of the robot concentrate there. The torque on each passive joint is related to its deflection θ as a spring-damper:

$$\tau_f = -k_f\theta - d_f\dot{\theta}, \tag{6.4}$$

with link stiffness k_f and damping d_f coefficients.

As the stiffness is coming for the vertical linkage, only the flexibility along pitch and roll deflections is modeled, which produce the main impact on foot placement. As a result, the model for the robot TALOS is augmented to reach 42 DoFs composed by 32 actuated joints, 4 elastic passive joints and the robot free-flyer. The obtained results in Villa et al. [206] are quite satisfying, see the video at the following link: https://peertube.laas.fr/videos/watch/9a3c5258-e5b7-49a5-a153-02e804a06f65.

With this modeling a new value of the stiffness flexibility has been identified. It is important to notice that between the identification experiments the legs have been unmounted and a backlash in the knees have been corrected. The second experiment consists in lifting a leg and putting the CoM at the center of the support foot, to match the CoP position computed from the F/T sensors, it allows to identify the static stiffness. For the left foot, without compensation, the CoP does not equal the CoM, it is naturally going toward the outside of the robot. With the compensation and a stiffness set at 3000 Nm/rad, the CoP and CoM nearly match. With a lower stiffness, the CoP goes toward the inside of the robot.

This observation is confirmed by checking the torque measured by the ankle F/T sensors. There should be no torque around the X-axis once the reference is reached, but there is $\sim$ 20Nm toward the outside of the robot without compensation. Using the flexibility compensation model with a stiffness of 3000 Nm/rad the torque is lowered to $\sim$ 5Nm and decreasing the stiffness makes the torque goes to the opposite direction. Thus, the second estimation of the stiffness flexibility found a value of $\sim$ 3000 Nm/rad on the left leg. For the right leg a value of $\sim$ 4750 Nm/rad has been identified.

It is a work in progress to interface the controllers developed in this thesis and the compensation model implemented in the paper Villa et al. [206].

Appendix 5: On the semi-positive definition of the potential energy S

This Appendix details the results presented in the open-access document Ramuzat [210].

Problem statement: We have Λ a real symmetric semi-positive definite matrix and K_p a real positive diagonal matrix (thus also symmetric). We defined in Chapter 5 the potential energy of a task as $S = \frac{1}{2}e^T\Lambda K_P e$. The product ΛK_p is a square matrix but non-symmetric. We want to prove that ΛK_p is semi-positive definite with respect to the Definition 5.

If the elements of the diagonal K_p are equal, i.e. $K_p = aI$ with $a \in \mathbb{R}^+\backslash\{0\}$, then the product $\Lambda K_p = a\Lambda$ is a real symmetric semi-positive definite matrix. Thus, for a diagonal K_p with equal elements we have directly $x^T\Lambda K_p x \geq 0 \, \forall x \in \mathbb{R}^n\backslash\{0\}$.

For different elements on the diagonal, we have the following results:

- The eigenvalues of ΛK_p are real and non-negatives
- If $\frac{\Lambda K_p + (\Lambda K_p)^T}{2}$ has non-negative eigenvalues, then the quadratic form of ΛK_p is semi-positive definite

Proof. Let us recall some useful matrix definitions and properties:

Definition 4 (Real symmetric semi-positive definite matrix)**.** *A is a real symmetric matrix (its eigenvalues are thus real): A is semi-positive definite $\iff$ all its eigenvalues are non-negatives.*

Definition 5 (Quadratic form semi-positive definition)**.** *A matrix A is semi-positive definite $\iff$ $x^TAx \geq 0$, $\forall x \in \mathbb{R}^n\backslash\{0\}$*

Property 2. *If A is a real semi-positive definite matrix, then B^TAB is semi-positive definite for any matrix B.*

Property 3 (Matrix congruent to a symmetric matrix)**.** *Any matrix congruent to a symmetric matrix is again symmetric: If A is a symmetric matrix then so is B^TAB for any matrix B.*

Proof that the eigenvalues of ΛK_p are real and non-negatives

Because K_p is diagonal and positive we can write $K_p = K_p^{\frac{1}{2}}K_p^{\frac{1}{2}}$, $K_p^{\frac{1}{2}}$ is also real positive and symmetric thus invertible. Let us reformulate the matrix ΛK_p:

$$\begin{aligned} \Lambda K_p &= \Lambda K_p^{\frac{1}{2}} K_p^{\frac{1}{2}} \\ &= K_p^{-\frac{1}{2}}(K_p^{\frac{1}{2}}\Lambda K_p^{\frac{1}{2}})K_p^{\frac{1}{2}} \end{aligned} \tag{6.5}$$

It corresponds to a change of basis of $K_p^{\frac{1}{2}}$. Because the eigenvalues (denoted λ) are invariant to change of basis we have:

$$\lambda(\Lambda K_p) = \lambda(K_p^{\frac{1}{2}} \Lambda K_p^{\frac{1}{2}}) \tag{6.6}$$

Using the Properties. 2 and 3, because Λ is a real symmetric semi-positive definite matrix, $(K_p^{\frac{1}{2}})^T \Lambda K_p^{\frac{1}{2}} = K_p^{\frac{1}{2}} \Lambda K_p^{\frac{1}{2}}$ is a real symmetric semi-positive definite matrix. Thus its eigenvalues are real and non-negatives (Definition.4) and so are the ones of Λ because of Eq.6.6. Then we have proven our first result: the eigenvalues of ΛK_p are real and non-negatives.

Semi-positive definition of ΛK_p

In this part we look at a way to prove the semi-positive definition of ΛK_p by studying its symmetric part in the Toeplitz decomposition:

Definition 6 (Toeplitz decomposition)**.** *Every square matrix A can be decomposed uniquely as the sum of two matrices U and V, where U is symmetric and V is skew-symmetric.*

$$A = U + V = \frac{1}{2}(A + A^T) + \frac{1}{2}(A - A^T) \tag{6.7}$$

In our case $A = \Lambda K_p$ and $U = \frac{1}{2}(\Lambda K_p + (\Lambda K_p)^T)$ is symmetric.

We recall the fact that the quadratic form of a skew-symmetric matrix equals to zero. Indeed, by definition $V^T = -V$ and thus $x^T V x = (x^T V^T x)^T = -x^T V x$ which holds only if it equals to zero. Thus, the quadratic form of $x^T \Lambda K_p x$ is the same one of $x^T U x$, i.e:

$$x^T \Lambda K_p x = x^T (\frac{\Lambda K_p + (\Lambda K_p)^T}{2}) x \tag{6.8}$$

One can thus prove the semi-positive definition of the symmetric matrix U to prove the semi-positive definition of ΛK_p. Indeed, if $x^T U x \geq 0$, $\forall x \in \mathbb{R}^n \backslash \{0\}$, using Eq.6.8, we obtain $x^T \Lambda K_p x \geq 0$, $\forall x \in \mathbb{R}^n \backslash \{0\}$: proving of the semi-positive definition of ΛK_p.

One way to prove the semi-positive definition of U is to look at its eigenvalues. Because U is symmetric, if its eigenvalues are non-negatives then U is semi-positive definite (see Definition 4). This gives our second result: if U has non-negative eigenvalues, then ΛK_p is semi-positive definite.

On the eigenvalues of U

One may notice that we have further information on the eigenvalues of U with respect to the ones of ΛK_p. Using the following theorem of Fan on matrices [215] (Chapter 10, Theorem 10.28):

Theorem 1 (Eigenvalues majorization of Ky Fan)**.** *Let A be an $n \times n$ matrix with eigenvalues $\lambda_1(A), ..., \lambda_n(A)$ and $\mathcal{R}_e\lambda_i(A)$ their real parts. Then:*

$$\sum_{i=1}^{n} \mathcal{R}_e\lambda_i(A) \leq \sum_{i=1}^{n} \lambda_i\Big(\frac{A + A^T}{2}\Big) \tag{6.9}$$

Then, because the eigenvalues of ΛK_p are real and non-negatives (as proven in the first part), we have using the Theorem 1:

$$0 \leq \sum_{i=1}^{n} \lambda_i(\Lambda K_p) \leq \sum_{i=1}^{n} \lambda_i(U) \tag{6.10}$$

Thus, we know that the sum of the eigenvalues of U is non-negative, however to prove the semi-positive definition of U it is needed to prove that all its eigenvalues are non-negatives.

□

Bibliography

[1] Louise Scherrer. Reactive walking for the humanoid robot pyrène. In *HAL open archive*, 2021. 1, 6, 20, 27, 35, 163

[2] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics: Modelling, Planning and Control*. Springer Publishing Company, Incorporated, 1st edition, 2008. ISBN 1846286417. 2, 22, 54, 55, 56, 57, 58, 59, 61

[3] Matthew Spenko, Stephen Buerger, and Karl Iagnemma. *The DARPA Robotics Challenge Finals: Humanoid Robots To The Rescue*. 01 2018. ISBN 978-3-319-74665-4. doi: 10.1007/978-3-319-74666-1. 2

[4] O. Stasse, T. Flayols, R. Budhiraja, K. Giraud-Esclasse, J. Carpentier, J. Mirabel, et al. Talos: A new humanoid research platform targeted for industrial applications. In *IEEE-RAS ICHR*, 2017. 4, 5

[5] K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi, and T. Isozumi. Humanoid robot hrp-2. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, volume 2, pages 1083–1090 Vol.2, 2004. doi: 10.1109/ROBOT.2004.1307969. 5, 6

[6] International Electrotechnical Commission. Industrial communication networks–fieldbus specifications–part 3–12: data-link layer service definition–part 4–12: datalink layer protocol specification–type 12 elements. *IEC*, 2007. 6

[7] Stanford Artificial Intelligence Laboratory et al. Robotic operating system. URL https://www.ros.org. 6, 14, 112, 113

[8] David Coleman, Ioan Sucan, Sachin Chitta, and Nikolaus Correll. Reducing the barrier to entry of complex robotic software: a moveit! case study. *arXiv preprint arXiv:1404.3785*, 2014. 6

[9] Sachin Chitta, Ioan Sucan, and Steve Cousins. Moveit![ros topics]. *IEEE Robotics & Automation Magazine*, 19(1):18–19, 2012. 6

[10] Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, volume 3, pages 2149–2154. IEEE, 2004. 6

[11] F. Negrello, M. Garabini, M.G. Catalano, P. Kryczka, W. Choi, D.G. Caldwell, A. Bicchi, and N.G. Tsagarakis. Walk-man humanoid lower body design optimization for enhanced physical performance. In *International Conference on Robotics and Automation (ICRA)*, 2016. 6

[12] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, USA, 2004. 11, 25, 38, 39, 41, 61

[13] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiraux, O. Stasse, and N. Mansard. The pinocchio c++ library – a fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives. In *Int. Symp. on System Integrations*, 2019. 14, 15, 16, 35, 62

[14] GEPETTO Team LAAS-CNRS. dynamic-graph. https://github.com/stack-of-tasks/dynamic-graph, . 14, 38

[15] R. Featherstone. *Rigid Body Dynamics Algorithms*. Springer, 2008. 14, 35, 62

[16] N. Mansard, O. Stasse, P. Evrard, and A. Kheddar. A versatile generalized inverted kinematics implementation for collaborative working humanoid robots: The stack of tasks. In *International Conference on Advanced Robotics (ICAR)*, 2009. 14, 15, 38, 90, 95, 96

[17] MathWorks. Simulation and model-based design, 2020. URL https://www.mathworks.com/products/simulink.html. 14

[18] GEPETTO Team LAAS-CNRS. roscontrol_sot. https://github.com/stack-of-tasks/roscontrol_sot, . 15

[19] Adolfo Rodríguez Tsouroukdissian. Ros control, an overview. URL https://roscon.ros.org. 15, 154

[20] GEPETTO Team LAAS-CNRS. sot-core. https://github.com/stack-of-tasks/sot-core, . 15, 38, 90

[21] A. Del Prete, N. Mansard, O. Ponce, O. Stasse, and F. Nori. Implementing torque control with high-ratio gear boxes and without joint-torque sensors. *International Journal of Humanoid Robotics*, 2016. 16, 91

[22] N. Ramuzat, F. Forget, V. Bonnet, M. Gautier, S. Boria, and O. Stasse. Actuator model, identification and differential dynamic programming for a talos humanoid robot. In *European Control Conference (ECC)*, 2020. 16, 18, 67, 86, 89

[23] Noëlie Ramuzat, Gabriele Buondonno, Sébastien Boria, and Olivier Stasse. Comparison of position and torque whole body control schemes on the humanoid robot talos. In *20th International Conference on Advanced Robotics (ICAR)*, 2021. 17, 18, 88, 114

[24] Isabelle Maroger, Noëlie Ramuzat, Olivier Stasse, and Bruno Watier. Human trajectory prediction model and its coupling with a walking pattern generator of a humanoid robot. *IEEE Robotics and Automation Letters*, 6(4):6361–6369, 2021. 17, 20, 65, 88, 111, 112, 113, 114, 118, 119

[25] Noëlie Ramuzat, Olivier Stasse, and Sébastien Boria. Benchmarking whole body controllers on talos. *Frontiers Robotics and AI*, 2022. 17

[26] Noëlie Ramuzat, Sébastien Boria, and Olivier Stasse. Passive inverse dynamics control using a global energy tank for torque-controlled humanoid robots in multi-contact. *IEEE Robotics and Automation Letters*, 2022. 17, 127

[27] M. Naveau, M. Kudruss, O. Stasse, C. Kirches, K. Mombaur, and P. Soueres. A reactive walking pattern generator based on nonlinear model predictive control. *IEEE Robotics and Automation Letters (RAL)*, 2, 2017. 20, 28, 34, 35

[28] Antal K. Bejczy. Robot arm dynamics and control. 1974. 21, 54, 151

[29] T. Flayols, A. Del Prete, P. Wensing, A. Mifsud, M. Benallegue, and O. Stasse. Experimental evaluation of simple estimators for humanoid robots. In *Int. Conf. on Humanoid Robotics (ICHR)*, 2017. 22, 98, 138

[30] Bernd Henze, Máximo A. Roa, and Christian Ott. Passivity-based whole-body balancing for torque-controlled humanoid robots in multi-contact scenarios. *The International Journal of Robotics Research*, 2016. 22, 23, 126, 128

[31] Pierre-Brice Wieber, Russ Tedrake, and Scott Kuindersma. Modeling and control of legged robots. In *Springer Handbook of Robotics, 2nd Ed.*, 2016. 23

[32] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. In *Int. Conf. on Robotics and Automation (ICRA)*, 2003. 23, 26, 28, 34, 35, 61

[33] O. E. Ramos, N. Mansard, O. Stasse, C. Benazeth, S. Hak, and L. Saab. Dancing humanoid robots: Systematic use of osid to compute dynamically consistent movements following a motion capture pattern. *IEEE Robotics Automation Magazine*, 2015. 23

[34] D. Orin, A. Goswami, and S.-H. Lee. Centroidal dynamics of a humanoid robot. *Autonomous Robot*, 35:161–176, 2013. 24, 25, 30

[35] Kevin M. Lynch and Frank Chongwoo Park. Modern robotics: Mechanics, planning, and control. 2017. 25, 32, 39, 40

[36] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 1987. doi: 10.1109/JRA.1987.1087068. 25, 38

[37] C. Samson, M. Leborgne, and B. Espiau. Robot control. the task-function approach. In *Oxford Engineering Science Series, vol. 22*. Oxford University Press, 1991. 25, 38, 94

[38] A. Escande, N. Mansard, and P.B. Wieber. Hierarchical quadratic programming: Fast online humanoid-robot motion generation. *Int. Jour. of Robotics Research*, 2014. 25, 38, 94

[39] Dusan Surla, Miomir Vukobratovic, and Branislav Borovac. *Biped Locomotion: Dynamics, Stability, Control and Application*. 1990. 26

[40] S. Kajita, H. Hirukawa, K. Harada, and K. Yokoi. *ZMP and Dynamics*. Springer Berlin Heidelberg, 2014. 26

[41] S. Kajita, O. Matsumoto, and M. Saigo. Real-time 3d walking pattern generation for a biped robot with telescopic legs. In *IEEE ICRA*, 2001. 26, 33

[42] A. Herdt, N. Perrin, and P. Wieber. Walking without thinking about it. In *IEEE/RSJ IROS*, 2010. 28, 34

[43] J. Pratt, J. Carff, S. Drakunov, and A. Goswami. Capture point: A step toward humanoid push recovery. In *Int. Conf. on Humanoid Robotics (ICHR)*, 2006. 29, 89

[44] T. Takenaka, T. Matsumoto, and T. Yoshiike. Real time motion generation and control for biped robot-4th report: Integrated balance control. In *Int. Conf. on Intelligent Robots and Systems (IROS)*, 2009. 29

[45] J. Englsberger, C. Ott, and A. Albu-Schäffer. Three-dimensional bipedal walking control based on divergent component of motion. *IEEE Transactions on Robotics*, 31(2), 2015. 29, 89

[46] Toru Takenaka, Takashi Matsumoto, and Takahide Yoshiike. Real time motion generation and control for biped robot-1st report: Walking gait pattern generation-. In *Int. Conf. on Intelligent Robots and Systems (IROS)*, 2009. 29

[47] S. Kajita, M. Morisawa, K. Miura, S. Nakaoka, K. Harada, K. Kaneko, F. Kanehiro, , and K. Yokoi. Biped walking stabilization based on linear inverted pendulum tracking. In *Int. Conf. on Intelligent Robots and Systems (IROS)*, 2010. 29, 89

[48] T. Sugihara. Standing stabilizability and stepping maneuver in planar bipedalism based on the best com-zmp regulator. In *Int. Conf. on Robotics and Automation (ICRA)*, 2009. 29

[49] G. Mesesan, J. Englsberger, G. Garofalo, C. Ott, and A. Albu-Schäffer. Dynamic walking on compliant and uneven terrain using dcm and passivity-based whole-body control. In *Int. Conf. on Humanoid Robotics (ICHR)*, 2019. 29, 37, 89, 102

[50] S. Caron, A. Kheddar, and O. Tempier. Stair climbing stabilization of the hrp-4 humanoid robot using whole-body admittance control. In *Int. Conf. on Robotics and Automation (ICRA)*, 2019. 29, 30, 37, 47, 89, 108

[51] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa. Resolved momentum control: Humanoid motion planning based on the linear and angular momentum. In *Int. Conf. on Intelligent Robots and Systems (IROS)*, 2003. 30, 116

[52] S. Lee and A. Goswami. A momentum-based balance controller for humanoid robots on non-level and non-stationary ground. *Autonomous Robots*, 2012. 30, 89

[53] P. M. Wensing and D. E. Orin. Generation of dynamic humanoid behaviors through task-space control with conic optimization. In *Int. Conf. on Robotics and Automation (ICRA)*, 2013. 30

[54] T. Koolen, S. Bertrand, G. Thomas, T. de Boer, T. Wu, J. Smith, J. Englsberger, and J. Pratt. Design of a momentum-based control framework and application to the humanoid robot atlas. *Int. Journal of Humanoid Robotics*, 13, 2016. 30, 42, 47, 89, 95, 110

[55] A. Albu-Schäffer, C. Ott, and G. Hirzinger. A unified passivity-based control framework for position, torque and impedance control of flexible joint robots. *The International Journal of Robotics Research*, 23:23–39, 2007. 31, 57, 151, 152

[56] E. Dombre W. Khalil. *Modeling, identification and control of robots*. Eds. Hermès Penton, London, United Kingdom, 2002. 31, 32, 69

[57] V. Bonnet, K. Pfeiffer, P. Fraisse, A. Crosnier, and G. Venture. Self-generation of optimal exciting motions for identification of a humanoid robot. *International Journal of Humanoid Robotics*, 15(06), 2019. 32, 68, 72

[58] M. Gautier and A. Jubien. Force calibration of KUKA lwr-like robots including embedded joint torque sensors and robot structure. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2014. 32, 68, 71, 72

[59] O. Stasse, B. Verrelst, P.-B. Wieber, B. Vanderborght, P. Evrard, A. Kheddar, and K. Yokoi. Modular architecture for humanoid walking pattern prototyping and experiments. *Advanced Robotics, Special Issue on Middleware for Robotics –Software and Hardware Module in Robotics System*, 22(6):589–611, 2008. 33, 34

[60] GEPETTO Team LAAS-CNRS. jrl-walkgen. https://github.com/stack-of-tasks/jrl-walkgen, . 33, 34

[61] J. Carpentier, A. D. Prete, S. Tonneau, T. Flayols, F. Forget, A. Mifsud, K. Giraud-Esclasse, D. Atchuthan, P. Fernbach, R. Budhiraja, M. Geisert andJ. Solà, O. Stasse, and N. Mansard. Multi-contact locomotion of legged robotsin complex environments – the loco3d project. In *RSS Work-shop on Challenges in Dynamic Legged Locomotion*, 2017. 33, 36

[62] GEPETTO Team LAAS-CNRS. multicontact-locomotion-planning. https://github.com/loco-3d/multicontact-locomotion-planning, . 33, 36, 138

[63] P. Wieber. On the stability of walking systems. In *Proceedings of the International Workshop on Humanoid and Human Friendly Robotics*, 2002. 33

[64] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. In *IEEE ICRA*, 2003. 33

[65] P. Wieber. Trajectory free linear model predictive control for stable walking in the presence of strong perturbations. In *IEEE-RAS Humanoids*, 2006. 34

[66] S. Faraji, S. Pouya, C. G. Atkeson, and A. J. Ijspeert. Versatile and robust 3d walking with a simulated humanoid robot (atlas): A model predictive control approach. In *IEEE ICRA*, 2014. 34

[67] R. J. Griffin and A. Leonessa. Model predictive control for dynamic footstep adjustment using the divergent component of motion. In *IEEE ICRA*, 2016. 34

[68] N. Scianca, D. De Simone, L. Lanari, and G. Oriolo. Mpc for humanoid gait generation: Stability and feasibility. *IEEE T-RO*, 2020. 34

[69] M. Naveau, M. Kudruss, O. Stasse, C. Kirches, K. Mombaur, and P. Souères. A reactive walking pattern generator based on nonlinear model predictive control. *IEEE RA-L*, 2017. 34, 113

[70] S. Caron and A. Kheddar. Dynamic walking over rough terrains by nonlinear predictive control of the floating-base inverted pendulum. In *IEEE/RSJ IROS*, 2017. 34

[71] Kevin Giraud-Esclasse. Towards a reactive motion generation on exteroceptive feedback for generalized locomotion of humanoid robots. In *Thesis in HAL open archive*, 2019. 34

[72] S. Tonneau, D. Song, P. Fernbach, N. Mansard, M. Taïx, and A. Del Prete. SL1M: Sparse L1-norm Minimization for contact planning on uneven terrain. In *Int. Conf. on Robotics and Automation (ICRA)*, 2020. 36

[73] B. Ponton, A. Herzog, S. Schaal, and L. Righetti. On time optimisation of centroidal momentum dynamics. *Int. Conf. on Robotics and Automation (ICRA)*, 2018. 36

[74] Johannes Englsberger, George Mesesan, Alexander Werner, and Christian Ott. Torque-based dynamic walking - a long way from simulation to experiment. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018. 37, 47, 62, 109, 146

[75] C. Chevallereau and W. Khalil. A new method for the solution of the inverse kinematics of redundant robots. In *Int. Conf. on Robotics and Automation (ICRA)*, 1988. 38

[76] N. Mansard, A. Del Prete, M. Geisert, S. Tonneau, and O. Stasse. Using a memory of motion to efficiently warm-start a nonlinear predictive controller. In *Int. Conf. on Robotics and Automation (ICRA)*, 2018. 03. 38, 62

[77] Mingxing Liu, Yang Tan, and Vincent Padois. Generalized hierarchical control. *Autonomous Robots*, 40(1):17–31, 2015. URL https://hal.archives-ouvertes.fr/hal-01068404. 38

[78] N. Mansard and F. Chaumette. Task sequencing for sensor-based control. In *IEEE Transactions on Robotics*, 2007. 38, 91

[79] GEPETTO Team LAAS-CNRS. sot-torque-control. https://github.com/stack-of-tasks/sot-torque-control, . 38, 91

[80] GEPETTO Team LAAS-CNRS. Tsid. https://github.com/stack-of-tasks/tsid, . 38, 91, 97, 129, 138

[81] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, NY, 2006. 38, 39, 44

[82] A. Hofmann, M. Popovic, and H. Herr. Exploiting angular momentum to enhance bipedal center-of-mass control. In *Int. Conf. on Robotics and Automation (ICRA)*, 2009. 39

[83] O. Kanoun, F. Lamiraux, and P.-B. Wieber. Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task. *IEEE Transactions on Robotics*, 27, 2011. 42

[84] K. Bouyarmane, J. Vaillant, K. Chappellet, and A. Kheddar. Multi-robot and task-space force control with quadraticprogramming. *IEEE Transactions on Robotics*, 35(1):64–77, 2019. 42, 47, 95

[85] Daniele Pucci, Gabriele Nava, and Francesco Nori. Automatic gain tuning of a momentum based balancing controller for humanoid robots. *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 158–164, 2016. 42

[86] M. Teshnehlab and K. Watanabe. Self tuning of computed torque gains by using neural networks with flexible structures. *IEE Proceedings - Control Theory and Applications*, 1994. 42

[87] Luigi Penco, Enrico Mingo Hoffman, Valerio Modugno, Waldez Gomes, Jean-Baptiste Mouret, and Serena Ivaldi. Learning robust task priorities and gains for control of redundant robots. *IEEE Robotics and Automation Letters*, 2020. doi: 10.1109/LRA.2020.2972847. 42

[88] E. M. Hoffman, A. Laurenzi, L. Muratore, N. G. Tsagarakis, and D. G. Caldwell. Multi-priority cartesian impedance control based on quadratic programming optimization. In *Int. Conf. on Robotics and Automation (ICRA)*, 2018. 43, 94

[89] B. Henze, A. Dietrich, and C. Ott. An approach to combine balancing with hierarchical whole-body control for legged humanoid robots. *IEEE Robotics and Automation Letters (RAL)*, 1:700–707, 2016. 43, 47, 94

[90] L. Saab, O. E. Ramos, F. Keith, N. Mansard, P. Souères, and J. Fourquet. Dynamic whole-body motion generation under rigid contacts and other unilateral constraints. *IEEE Transactions on Robotics*, 29(2):346–362, 2013. 43

[91] Grzegorz Ficht and Sven Behnke. Bipedal humanoid hardware design: A technology review. *CoRR*, abs/2103.04675, 2021. URL https://arxiv.org/abs/2103.04675. 45

[92] PAL-Robotics. Talos - torque controlled balancing, 2018. URL https://www.youtube.com/watch?v=lX6Q0zgFut0. 45

[93] Hyobin Jeong, KangKyu Lee, Wooshik Kim, Inho Lee, and Jun-Ho Oh. Design and control of the rapid legged platform gazelle. *Mechatronics*, 66, 2020. 45, 47

[94] N. G. Tsagarakis, D. G. Caldwell, F. Negrello, W. Choi, L. Baccelliere, V. Loc, J. Noorden, L. Muratore, A. Margan, A. Cardellino, L. Natale, E. M. Hoffman, H. Dallali, N. Kashiri, J. Malzahn, J. Lee, P. Kryczka, D. Kanoulas, M. Garabini, M. G. Catalano, M. Ferrati, V. Varricchio, L. Pallottino, C. Pavan, A. Bicchi, A. Settimi, A. Rocchi, and A. Ajoudani. WALK-MAN: A high-performance humanoid platform for realistic environments. *J. Field Robotics*, 34(7):1225–1259, 2017. 45

[95] Rajesh Subburaman, Jinoh Lee, Darwin G. Caldwell, and Nikos G. Tsagarakis. Online falling-over control of humanoids exploiting energy shaping and distribution methods. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 448–454, 2018. doi: 10.1109/ICRA.2018.8462880. 45

[96] Gerrit A. Folkertsma and Stefano Stramigioli. Energy in robotics. *Foundations and Trends® in Robotics*, pages 140–210, 2017. 45, 49, 51, 52, 128

[97] J. Englsberger, A. Werner, C. Ott, B. Henze, M. A. Roa, G. Garofalo, R. Burger, A. Beyer, O. Eiberger, K. Schmid, and A. Albu-Schäffer. Overview of the torque-controlled humanoid robot toro. In *IEEE/RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2014. 45, 68, 89

[98] Johannes Englsberger, Alexander Dietrich, George Mesesan, Gianluca Garofalo, Christian Ott, and Alin Albu-Schäeffer. Mptc – modular passive tracking controller for stack of tasks based control frameworks. In *Robotics Science and Systems*, 2020. 47, 49, 53, 128, 136

[99] Abderrahmane Kheddar, Stéphane Caron, Pierre Gergondet, Andrew Comport, Arnaud Tanguy, Christian Ott, Bernd Henze, George Mesesan, Johannes Englsberger, Máximo A Roa, Pierre-Brice Wieber, François Chaumette, Fabien Spindler, Giuseppe Oriolo, Leonardo Lanari, Adrien Escande, Kevin Chappellet, Fumio Kanehiro, and Patrice Rabate. Humanoid robots in aircraft manufacturing. *IEEE Robotics and Automation Magazine*, 2019. doi: 10.1109/MRA.2019.2943395. URL https://hal-lirmm.ccsd.cnrs.fr/lirmm-02303117. 47

[100] T. Shirai, Y. Nagamatsu, H. Suzuki, S. Nozawa, K. Okada, and M. Inaba. Design and evaluation of torque based bipedal walking control system that prevent fall over by impulsive disturbance. In *Int. Conf. on Intelligent Robots and Systems (IROS)*, 2018. 47

[101] Andrea Bacciotti and Lionel Rosier. *Liapunov Functions and Stability in Control Theory*. 2005. 48

[102] Gabriele Nava, Francesco Romano, Francesco Nori, and Daniele Pucci. Stability analysis and design of momentum-based controllers for humanoid robots. In *Int. Conf. on Intelligent Robots and Systems (IROS)*, 2016. 49

[103] R. Cisneros, M. Benallegue, A. Benallegue, M. Morisawa, H. Audren, P. Gergondet, A. Escande, A. Kheddar, and F. Kanehiro. Robust humanoid control using a qp solver with integral gains. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2018. 49, 68, 95

[104] Arjan Schaft. *L2-Gain and Passivity in Nonlinear Control*, volume 218. Springer International Publishing, 01 2000. ISBN 1852330732. doi: 10.1007/978-1-4471-0507-7. 49, 50, 129

[105] Lucas Joseph, Vincent Padois, and Guillaume Morel. Towards x-ray medical imag-ing with robots in the open: safety without compromising performances. *IEEEInternational Conference on Robotics and Automation (ICRA)*, 2018. 49, 136

[106] M.K. Camlibel, A.A. Julius, R. Pasumarthy, and J.M.A. Scherpen, editors. *Mathematical Control Theory I: Nonlinear and Hybrid Control Systems*, chapter 3. Lecture Notes in Control and Information Sciences. Springer, 2015. 49, 128

[107] Jan C. Willems. *Dissipative dynamical systems part I: General theory.* 1972. 49, 129

[108] Gianluca Garofalo and Christian Ott. Passive energy-based control via energy tanks and release valve for limit cycle and compliance control. *IFAC-PapersOnLine*, 51(22):73 – 78, 2018. 12th IFAC Symposium on Robot Control SYROCO 2018. 51, 52, 53, 128, 130

[109] C. Ott, A. Albu-Schaffer, A. Kugi, and G. Hirzinger. On the passivity-based impedance control of flexible joint robots. *IEEE Transactions on Robotics*, 2008. 51, 52, 130

[110] Paolo Robuffo Giordano, Antonio Franchi, Cristian Secchi, and Heinrich H Bülthoff. A passivity-based decentralized strategy for generalized connectivity maintenance. *The International Journal of Robotics Research*, 32(3):299–323, 2013. doi: 10.1177/0278364912469671. 51, 130

[111] Alexander Dietrich, Christian Ott, and Stefano Stramigioli. Passivation of projection-based null space compliance control via energy tanks. *IEEE Robotics and Automation Letters*, 1(1):184–191, 2016. doi: 10.1109/LRA.2015.2512937. 52, 130

[112] Alexander Dietrich, Xuwei Wu, Kristin Bussmann, Christian Ott, Alin Albu-Schäffer, and Stefano Stramigioli. Passive hierarchical impedance control via energy tanks. *IEEE Robotics and Automation Letters*, 2(2):522–529, 2017. doi: 10.1109/LRA.2016.2645504. 52, 53, 128, 130, 134, 147

[113] Erfan Shahriari, Lars Johannsmeier, Elisabeth Jensen, and Sami Haddadin. Power flow regulation, adaptation, and learning for intrinsically robust virtual energy tanks. *IEEE Robotics and Automation Letters*, 5(1):211–218, 2020. doi: 10.1109/LRA.2019.2953662. 52, 53, 93, 134, 142, 145, 146, 147, 161

[114] Tadele Shiferaw Tadele, Theo De Vries, and Stefano Stramigioli. Combining energy and power based safety metrics in controller design for domestic robots. In *Int. Conf. on Robotics and Automation (ICRA)*, 2014. 52

[115] A. Albu-Schäffer, C. Ott, and G. Hirzinger. A unified passivity-based control framework for position, torque and impedance control of flexible joint robots. *The international journal of robotics research*, 2007. 52, 54, 56, 57, 58

[116] Andrea Giusti, Jörn Malzahn, Nikolaos G. Tsagarakis, and Matthias Althoff. On the combined inverse-dynamics/passivity-based control of elastic-joint robots. *IEEE Transactions on Robotics*, 2018. 53

[117] J. A Acosta and M. Lopez-Martinez. Constructive feedback linearization of underactuated mechanical systems with 2-dof. In *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, 2006. 53

[118] Gordon Cheng, Sang-ho Hyon, Jun Morimoto, Ales Ude, Glenn Colvin, Wayco Scroggin, and Stephen C. Jacobsen. Cb: A humanoid research platform for exploring neuroscience. In *2006 6th IEEE-RAS ICHR*, 2006. 53

[119] Bernd Henze, Máximo A. Roa, and Christian Ott. Passivity-based whole-body balancing for torque-controlled humanoid robots in multi-contact scenarios. *Int. Jour. of Robotics Research*, 2016. 53, 128

[120] Arvid QL Keemink, Herman van der Kooij, and Arno HA Stienen. Admittance control for physical human–robot interaction. *The International Journal of Robotics Research*, 37(11):1421–1444, 2018. doi: 10.1177/0278364918768950. 54, 55

[121] Wen Yu and Adolfo Perrusquía. Simplified stable admittance control using end-effector orientations. *Int. J. Soc. Robotics*, 12:1061–1073, 2020. 54

[122] W.-S. Lu and Q.-H. Meng. Impedance control with adaptation for robotic manipulations. *IEEE Transactions on Robotics and Automation*, 7(3):408–415, 1991. doi: 10.1109/70.88152. 54, 56

[123] R.J. Anderson and M.W. Spong. Hybrid impedance control of robotic manipulators. *IEEE Journal on Robotics and Automation*, 4(5):549–556, 1988. doi: 10.1109/56.20440. 54, 55, 59, 60

[124] A. Albu-Schaffer and G. Hirzinger. Cartesian impedance control techniques for torque controlled light-weight robots. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, volume 1, pages 657–663 vol.1, 2002. doi: 10.1109/ROBOT.2002.1013433. 54, 56, 57, 58

[125] Christopher Schindlbeck and Sami Haddadin. Unified passivity-based cartesian force/impedance control for rigid and flexible joint robots via task-energy tanks. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 440–447, 2015. doi: 10.1109/ICRA.2015.7139036. 54, 57, 134, 152

[126] M. H. Raibert and J. J. Craig. Hybrid Position/Force Control of Manipulators. *Journal of Dynamic Systems, Measurement, and Control*, 103(2):126–133, 06 1981. doi: 10.1115/1.3139652. URL https://doi.org/10.1115/1.3139652. 55, 58, 59

[127] Christian Ott, Ranjan Mukherjee, and Yoshihiko Nakamura. A hybrid system framework for unified impedance and admittance control. *Journal of Intelligent and Robotic Systems*, 78, 06 2014. doi: 10.1007/s10846-014-0082-1. 55, 57, 60

[128] Hyomin Kim, Jaesung Kwon, Yonghwan Oh, Bum Jae You, and Woosung Yang. Weighted hybrid admittance-impedance control with human intention based stiffness estimation for human-robot interaction. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–6, 2018. doi: 10.1109/IROS.2018.8594435. 55, 60

[129] S. Chiaverini and L. Sciavicco. The parallel approach to force/position control of robotic manipulators. *IEEE Transactions on Robotics and Automation*, 9 (4):361–373, 1993. doi: 10.1109/70.246048. 55, 60, 61

[130] Bruno Siciliano. Parallel force/position control of robot manipulators. In Georges Giralt and Gerhard Hirzinger, editors, *Robotics Research*, pages 78–89, London, 1996. Springer London. 55, 60, 61

[131] J. Kenneth Salisbury. Active stiffness control of a manipulator in cartesian coordinates. In *1980 19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, pages 95–100, 1980. doi: 10.1109/CDC.1980.272026. 55, 57

[132] M.d.P.A. Fonseca, B.V. Adorno, and P. Fraisse. Task-space admittance controller with adaptive inertia matrix conditioning. *Journal of Intelligent Robot System*, 41, February 2021. doi: https://doi.org/10.1007/s10846-020-01275-0. 56

[133] Arash Ajoudani, Nikos G. Tsagarakis, Jinoh Lee, Marco Gabiccini, and Antonio Bicchi. Natural redundancy resolution in dual-arm manipulation using configuration dependent stiffness (cds) control. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1480–1486, 2014. doi: 10.1109/ICRA.2014.6907047. 57, 58

[134] A. Albu-Schaffer, M. Fischer, G. Schreiber, F. Schoeppe, and G. Hirzinger. Soft robotics: what cartesian stiffness can obtain with passively compliant, uncoupled joints? In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, volume 4, pages 3295–3301 vol.4, 2004. doi: 10.1109/IROS.2004.1389925. 57, 58

[135] Arash Ajoudani, Nikos G. Tsagarakis, and Antonio Bicchi. On the role of robot configuration in cartesian stiffness control. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1010–1016, 2015. doi: 10.1109/ICRA.2015.7139300. 58

[136] J. Rawlings, D. Mayne, and M. Diehl. *Model Predictive Control: Theory, Computation and Design*. Nob Hill Publishing, 2017. 61

[137] Mikulas Huba. *Selected Topics on Constrained and Nonlinear Control*, pages 87–185. 01 2011. ISBN 978-80-968627-4-0. 61

[138] J. Rawlings, E. Meadows, and K. Muske. *Nonlinear model predictive control: A tutorial and survey*. IFAC Advanced Control of Chemical Processes, 1994. 61

[139] Rolf Findeisen and Frank Allgöwer. An introduction to nonlinear model predictive control. In *Conference: 21st Benelux Meeting on Systems and Control*, 2002. 61

[140] T. Raff, C. Ebenbauer, and P. Allgöwer. *Nonlinear Model Predictive Control: A Passivity-Based Approach*, pages 151–162. Springer, 2007. 61, 162

[141] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *Int. Conf. on Intelligent Robots and Systems (IROS)*, 2012. 61, 64

[142] H. Dai, A. Valenzuela, and R. Tedrake. Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. In *Int. Conf. on Humanoid Robotics (ICHR)*, 2014. 61

[143] Donghyun Kim, Jared Di Carlo, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control. *CoRR*, abs/1909.06586, 2019. URL http://arxiv.org/abs/1909.06586. 61

[144] Michael Neunert, Cédric de Crousaz, Fadri Furrer, Mina Kamel, Farbod Farshidian, Roland Siegwart, and Jonas Buchli. Fast nonlinear model predictive control for unified trajectory optimization and tracking. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, page 1398–1404. IEEE Press, 2016. doi: 10.1109/ICRA.2016.7487274. URL https://doi.org/10.1109/ICRA.2016.7487274. 61

[145] Ruben Grandia, Farbod Farshidian, Rene Ranftl, and Marco Hutter. Feedback mpc for torque-controlled legged robots. In *EEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4730–4737, 11 2019. doi: 10.1109/IROS40897.2019.8968251. 61

[146] David Mayne. A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems. *International Journal of Control*, 3(1):85–95, 1966. 61

[147] Ewen Louis Dantec, Rohan Budhiraja, Adria Roig, Teguh Lembono, Guilhem Saurel, Olivier Stasse, Pierre Fernbach, Steve Tonneau, Sethu Vijayakumar, Sylvain Calinon, Michel Taïx, and Nicolas Mansard. Whole Body Model Predictive Control with a Memory of Motion: Experiments on a Torque-Controlled Talos. In *International Conference on Robotics and Automation (ICRA 2021)*, 2021. URL https://hal.archives-ouvertes.fr/hal-02995796. 61, 62, 123, 162

[148] A. Majumdar, G. Hall, and A. A. Ahmadi. Recent scalability improvements for semidefinite programming with applications in machine learning, control, and robotics. 2020. 62

[149] Y. Tassa, T. Erez, and W. D. Smart. Receding horizon differential dynamic programming. In *Advances in Neural Information Processing Systems*, 2008. 62

[150] D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2017. 62

[151] Y. Tassa, N. Mansard, and E. Todorov. Control-limited differential dynamic programming. In *IEEE ICRA*, 2014. 62, 63, 112

[152] Z. Xie, C. K. Liu, and K. Hauser. Differential dynamic programming with nonlinear constraints. In *Int. Conf. on Robotics and Automation (ICRA)*, pages 695–702, 2017. 62

[153] A. Herzog, N. Rotella, S. Mason, F. Grimminger, S. Schaal, and L. Righetti. Momentum control with hierarchical inverse dynamics on a torque-controlled humanoid. *Autonomous Robots*, 2014. 62, 89, 92, 94, 95, 114

[154] M. Diehl, H. G. Bock, , and J. P. Schlödre. A real-time iteration scheme for nonlinear optimization in optimal feedback control. In *SIAM Journal on control and optimization*, 2005. 62

[155] Richard Bellman. On the theory of dynamic programming. *Proceedings of the National Academy of Sciences*, 38(8):716–719, 1952. doi: 10.1073/pnas.38.8.716. 62

[156] F. Forget, K. Giraud-Esclasse, R. Gelin, N. Mansard, and O. Stasse. Implementation, identification and control of an efficient electric actuator for humanoid robots. In *International Conference on Informatics in Control, Automation and Robotics, ICINCO*, 2018. 63, 73, 79

[157] Yuval Tassa, Nicolas Mansard, and Emo Todorov. Control-limited differential dynamic programming. In *icra*, 2014. 63

[158] W. Li and E. Todorov. Iterative linearisation methods for approximately optimal control and estimation of non-linear stochastic system. *Int. Jour. of Control*, 2007. 63, 74

[159] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction, second edition*. The MIT Press, 2018. 64

[160] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–444, 2015. 64

[161] Roland Siegwart Jemin Hwangbo, Inkyu Sa and Marco Hutter. Control of a quadrotor with reinforcement learning. *IEEE Robotics and Automation Letters 2096–2103*, 2017. 64

[162] D. Esteban, L. Rozo, and D. G. Caldwell. Learning deep robot controllers by exploiting successful and failed executions. In *Int. Conf. on Humanoid Robotics (ICHR)*, 2018. 64

[163] L. Bottou and O. Bousquet. *Optimization for Machine Learning*, chapter The Tradeoffs of Large Scale Learning, page 351–368. Cambridge: MIT Press, 2012. 64

[164] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 2019. 64, 69

[165] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. In *Robotics: Science and Systems*, 2018. 64, 69

[166] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2016. 64

[167] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. 64

[168] Stephen James, Michael Bloesch, and Andrew J. Davison. Task-embedded control networks for few-shot imitation learning. *ArXiv*, 2018. 64

[169] Kosuge K., Yoshida H., and Fukuda T. Dynamic control for robot-human collaboration. In *Proceedings of 1993 2nd IEEE International Workshop on Robot and Human Communication.*, 1993. 64

[170] Isabelle Maroger, Olivier Stasse, and Bruno Watier. Inverse Optimal Control to Model Human Trajectories During Locomotion. *Computer Methods in Biomechanics and Biomedical Engineering*, 2021. doi: 10.1080/10255842.2021.1962311. URL https://hal.laas.fr/hal-03142655. 65

[171] N. Jarrassé, V. Sanguineti, and E. Burdet. Slaves no longer: review on role assignment for human–robot joint motor action. *Adaptive Behavior, SAGE Publications*, 2013. 65

[172] Antoine Bussy, Pierre Gergondet, Abderrahmane Kheddar, Francois Keith, and Andre Crosnier. Proactive behavior of a humanoid robot in a haptic transportation task with a human partner. In *IEEE RO-MAN*, 2012. 65

[173] K. Otani, K. Bouyarmane, and S. Ivaldi. Generating assistive humanoid motions for co-manipulation tasks with a multi-robot quadratic program controller. In *IEEE ICRA*, 2018. 65

[174] P. Teja S. and R. Alami. Hateb-2: Reactive planning and decision making in human-robot co-navigation. In *IEEE RO-MAN*, 2020. 65

[175] J. Lanini, A. Duburcq, H. Razavi, C.G. Le Goff, and A. Ijspeert. Interactive locomotion: Investigation and modeling of physically-paired humans while walking. *PLoS ONE*, 2017. 65

[176] J. Lanini, H. Razavi, J. Urain, and A. Ijspeert. Human intention detection as a multiclass classification problem: Application in physical human–robot interaction while walking. *IEEE RA-L*, 2018. 65

[177] Daniel Aarno and Danica Kragic. Motion intention recognition in robot assisted applications. *Robotics and Autonomous Systems*, 2008. 65

[178] R. Kelley, M. Nicolescu, A. Tavakkoli, M. Nicolescu, C. King, and G. Bebis. Understanding human intentions via hidden markov models in autonomous mobile robots. In *ACM/IEEE HRI*, 2008. 65

[179] Q. Li, Z. Zhang, Y. You, Y. Mu, and C. Feng. Data driven models for human motion prediction in human-robot collaboration. *IEEE Access*, 2020. 65

[180] M. Awais and D. Henrich. Human-robot collaboration by intention recognition using probabilistic state machines. In *RAAD*, 2010. 65

[181] David A. Winter. Biomechanics and motor control of human gait. In *Waterloo, Ont. : University of Waterloo Press*, 1991. 65

[182] Mirko Raković, Srdjan Savić, José Santos-Victor, Milutin Nikolić, and Branislav Borovac. Human-inspired online path planning and biped walking realization in unknown environment. *Frontiers in Neurorobotics*, 2019. 66

[183] Alessandro Papadopoulos, Luca Bascetta, and Gianni Ferretti. Generation of human walking paths. In *IEEE IROS*, 2013. 66

[184] Gustavo Arechavaleta, Jean-Paul Laumond, Halim Hicheur, and Alain Berthoz. On the nonholonomic nature of human locomotion. *Autonomous Robots*, 2008. 66

[185] K. Mombaur and J.-P. Laumond. From human to humanoid locomotion - an inverse optimal control approach. *Autonomous Robots*, 2010. 66

[186] Isabelle Maroger, Olivier Stasse, and Bruno Watier. Walking human trajectory models and their application to humanoid robot locomotion. In *IEEE/RSJ IROS*, 2020. 66, 112

[187] Isabelle Maroger, Olivier Stasse, and Bruno Watier. Inverse optimal control to model human trajectories during gait. In *HAL open archive*, 2021. 66, 112, 116

[188] S. Park, J. Sim, and J. Park. System design of humanoid robot dyros-jet. In *IEEE/SICE International Symposium on System Integration (SII)*, 2019. 68

[189] O. Stasse, T. Flayols, R. Budhiraja, K. Giraud-Esclasse, J. Carpentier, and al.. Talos: A new humanoid research platform targeted for industrial applications. In *IEEE/RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2017. 68

[190] O. Khatib, L. Sentis, J. Park, and J. Warren. Whole-body dynamic behavior and control of human-like robots. *International Journal of Humanoid Robotics*, 2004. 68

[191] O. Khatib, P. Thaulad, T. Yoshikawa, and J. Park. Torque-position transformer for task control of position controlled robots. In *IEEE/RAS Int. Conf. on Robotics and Automation (ICRA)*, 2008. 68

[192] A. Del Prete, N. Mansard, O. Ponce O., Stasse, and F. Nori. Implementing torque control with high-ratio gear boxes and without joint-torque sensors. *International Journal of Humanoid Robotics*, 2015. 68

[193] F. Nori, S. Traversaro, J. Eljaik, F. Romano, A. Del Prete, and D. Pucci. icub whole-body control through force regulation on rigid non-coplanar contacts. *Frontiers in Robotics and AI*, 2015. 68

[194] M. Gautier and S. Briot. Global identification of joint drive gains and dynamic parameters of robots. *ASME Journal of Dynamic Systems*, 2014. 68, 71

[195] M. Gautier. Dynamic identification of robots with power model. *ICRA*, 1997. 72

[196] G. Kumar Hari Shankar Lal Das, B. Tondu, F. Forget, J. Manhes, O. Stasse, and P. Soueres. Controlling a multi-joint arm actuated by pneumatic muscles with quasi-ddp optimal control. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2016. 73, 74

[197] C. Presse and M. Gautier. New criteria of exciting trajectories for robot identification. In *IEEE/RAS Int. Conf. on Robotics and Automation (ICRA)*, 1993. 78

[198] GEPETTO Team LAAS-CNRS. eiquadprog. https://github.com/stack-of-tasks/eiquadprog, . 94

[199] Donald Goldfarb and Ashok U. Idnani. A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming*, 27: 1–33, 1983. 94

[200] G. Romualdi, S. Dafarra, Y. Hu, P. Ramadoss, F. Andrade Chavez, S. Traversaro, and D. Pucci. A benchmarking of dcm based architectures for position, velocity and torque controlled humanoid robots. *Int. Journal of Humanoid Robotics*, 2019. 95, 105

[201] GEPETTO Team LAAS-CNRS. sot-talos-balance. https://github.com/loco-3d/sot-talos-balance, . 96

[202] GEPETTO Team LAAS-CNRS. talos-torque-control. https://github.com/stack-of-tasks/talos-torque-control, . 97, 138

[203] D. Torricelli, J. Gonzalez-Vargas, J. Veneman, K. Mombaur, N. Tsagarakis, A. del Ama, A. Gil-Agudo, J. Moreno, and J. Pons. Benchmarking bipedal locomotion: A unified scheme for humanoids, wearable robots, and humans. *IEEE Robotics Automation Magazine*, 2015. 98, 99, 100

[204] C. Mummolo and J. H. Kim. Passive and dynamic gait measures for biped mechanism: formulation and simulation analysis. In *Robotica*, 2012. 99, 106, 108

[205] S. Collins, A. Ruina, R. Tedrake, and M. Wisse. Efficient bipedal robots based on passive-dynamic walkers. *Science*, 2005. 106

[206] Nahuel A. Villa, Pierre Fernbach, Nicolas Mansard, and Olivier Stasse. Addressing flexibility in biped locomotion with robust control and closed-loop model-predictive control. In *IEEE/RAS Int. Conf. on Robotics and Automation (ICRA)*, submitted in 2022. 109, 110, 160, 161, 170, 171

[207] C. Mastalli, R. Budhiraja, W. Merkt, G. Saurel, B. Hammoud, M. Naveau, et al. Crocoddyl: An efficient and versatile framework for multi-contact optimal control. In *IEEE ICRA*, 2020. 112

[208] Isabelle Maroger, Olivier Stasse, and Bruno Watier. Metrics to assess a human trajectory prediction model during gait. In *HAL open archive*, 2021. 118

[209] Dirk Merkel. Docker: lightweight linux containers for consistent development and deployment. *Linux journal*, 2014(239):2, 2014. 123

[210] Noëlie Ramuzat. On the semi-positive definition of the product of a positive diagonal matrix and a symmetric semi-positive definite matrix. working paper or preprint, January 2022. URL https://hal.archives-ouvertes.fr/hal-03434195. 131, 145, 172

[211] Adi Ben-Israel and Thomas N.E. Greville. *Generalized Inverses: Theory and Applications*. Springer, 2003. ISBN 0-387-00293-6. 132

[212] T. N. E. Greville. Note on the generalized inverse of a matrix product. *SIAM Review*, 8(4):518–521, 1966. doi: 10.1137/1008107. 132

[213] Klas Kronander and A. Billard. Passive interaction control with dynamical systems. *IEEE Robotics and Automation Letters (RAL)*, 2016. 134, 145

[214] Shin'ichiro Nakaoka, Shizuko Hattori, Fumio Kanehiro, Shuuji Kajita, and Hirohisa Hirukawa. Constraint-based dynamics simulator for humanoid robots with shock absorbing mechanisms. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3641–3647. IEEE, 2007. 171

[215] Fuzhen Zhang. *Matrix theory: basic results and techniques*. Springer, 2011. 173

Glossary

AIST National Institute of Advanced Industrial Science and Technology. 5, 45, 47

AM Angular Momentum. 28, 30, 47, 92, 114, 116

CoM Center of Mass. 12, 16, 23–26, 28–30, 33–35, 37, 47, 48, 89–92, 95–98, 100, 102, 109, 111–114, 116, 117, 119, 120, 125, 141–143, 152, 161, 171

CoP Center of Pressure. 23, 25, 26, 33, 34, 122, 123, 143, 171

DCM Divergent Component of Motion. 29, 47, 89, 95–98, 100, 102, 104, 109, 120

DDP Differential Dynamic Programming. 62, 63, 68, 69, 73, 74, 112

DoF Degree of Freedom. 2, 3, 5, 7, 9, 10, 21, 45, 53, 54, 58, 61, 64, 132, 152, 171

IMU Inertial Measurement Unit. 5, 7, 9, 22, 45, 68, 98, 138, 154

LAAS Laboratory for Analysis and Architecture of Systems. 3, 4, 10

LIPM Linear Inverted Pendulum Model. 26, 28–30, 33, 35, 47, 89

MPC Model Predictive Control. 4, 16, 17, 34, 61–64, 68, 123, 154, 159, 162

MSDR Middle Size Drilling Robot. 3, 5, 9, 10, 12, 17, 147, 150, 151, 153, 154, 156, 157, 160, 162

NMPC Non-linear Model Predictive Control. 113, 114

OCP Optimal Control Problem. 61, 62, 65, 66, 112, 119

PD+ Proportional Derivative with feed-forward. 53, 55, 109, 128, 154, 157, 160

PID Proportional Integral Derivative. 32, 37, 55, 102, 154

QP Quadratic Programming. 30, 38, 40–44, 47, 49, 53, 73, 89, 91, 92, 94–98, 109, 124, 125, 127, 128, 136, 139, 142, 143, 146, 153, 154, 160–162

ROB4FAM Robots For the Future of Aircraft Manufacturing. 1, 3, 4, 7, 11, 20, 157, 162

ROS Robot Operating System. 6, 14, 15, 147, 150, 154, 157

RST Regulation Sensitivity and Tracking. 154

SoT Stack-of-Tasks. 12, 14, 15, 123

TSID Task Space Inverse Dynamics. xi, 15, 16, 38, 91–94, 96, 128, 129, 152, 153, 160

URDF Unified Robot Description Format. 8, 15

WPG Walking Pattern Generator. 17, 28, 33–36, 98, 111, 113, 114, 116, 118–120, 123

ZMP Zero Moment Point. 26, 28–30, 34, 35, 98, 100, 102, 105, 108, 109

Résumé

Les robots humanoïdes ne sont pas encore capables de travailler en sécurité dans un environnement fait pour l'Homme et d'effectuer les mêmes tâches. L'objectif de cette thèse est d'étudier des algorithmes de contrôle corps-complet pour robots humanoïdes afin d'effectuer des opérations de productions de structures avions telles que le perçage. Cette thèse s'inscrit dans le cadre du laboratoire commun entre le LAAS-CNRS et la société Airbus Operations SAS. Les travaux et études présentés appartiennent aux domaines scientifiques de l'optimisation pour le contrôle de systèmes dynamiques et de l'analyse de leur stabilité.

Les contributions majeures de cette thèse sont la conception et l'implémentation de nouveaux algorithmes de contrôle temps-réel pour des robots humanoïdes, intégrés sur le robot TALOS. Les applications présentées traitent de la locomotion du robot et de la réalisation d'opérations nécessitant l'application de force. Premièrement, les paramètres de la chaîne d'actionnement du robot sont identifiés afin de commander directement le robot en courant tout en garantissant la protection du système. La solution proposée confirme les capacités du robot TALOS contrôlé en couple lors de la manipulation d'une charge élevée. Ainsi, des algorithmes de contrôle temps-réel en couple pour le corps-complet du robot sont ensuite étudiés. Trois contrôleurs sont implémentés et comparés, deux en position et un en couple. Leur analyse est effectuée sur des scénarios de locomotion complexes en utilisant plusieurs critères comme le suivi de trajectoires et la dépense énergétique. Le contrôleur corps-complet en couple est validé en simulation et ses avantages par rapport aux schémas en position confirment le choix de l'implémenter pour des opérations de productions. Cependant, les premiers tests réalisés sur le robot réel ont conduit à une dangereuse divergence de la solution. C'est pourquoi, une analyse de stabilité a été effectuée pour assurer la sécurité et la robustesse de la solution pour des applications industrielles. La solution développée est basée sur la théorie de la passivité, augmentant le contrôleur en couple avec un réservoir d'énergie contrôlant les transmissions d'énergie au sein du système. Ce nouveau contrôleur passif est validé en simulation sur des scénarios de locomotion et de multi-contacts avec l'environnement. Ce dernier met en place une tâche d'application de force comme première étape vers des opérations de productions.

Les résultats de cette thèse ont été intégrés dans la suite logicielle *Stack-of-Tasks* du LAAS-CNRS.

Mots clés : Contrôle corps-complet, Robots humanoïdes, Contrôle en force/ couple

---

Abstract

Humanoids robots are not yet able to safely work in similar environment than humans and to perform the same tasks. The objective of this thesis is to study whole-body control algorithms for humanoids robots in order to perform aircraft manufacturing operations such as drilling. This thesis is part of the joint-lab between the LAAS-CNRS laboratory and the Airbus Operations SAS company. The presented research and innovative studies rely on recent advances on control theory, optimization and stability analysis.

The main contributions of this thesis are the design and implementation of new real-time controllers for humanoid robot, integrated on the robot TALOS. The illustrated applications are locomotion and force task operations. First, the chain actuation parameters of the robot are identified in order to directly control the robot in current while having low level protection mechanisms. The proposed solution confirms the capabilities of the robot TALOS to achieve torque control in complex scenario with high payload. Thus, real-time whole-body torque controllers are then investigated. Three controllers are implemented and benchmarked, two in position and one in torque. Their comparison is performed on complex locomotion scenarios using several metrics including trajectory tracking and energetic criteria. The whole-body torque controller is validated in simulations and its advantages compared to the position schemes confirms the decision to implement it for manufacturing operations. However, the first tests realized on the real robot led to a dangerous divergence of the solution. Thus, a stability analysis was performed to ensure the safety and robustness of the solution for industrial applications. The developed solution is based on passivity theory, augmenting the whole-body torque control scheme with a global energy tank monitoring the power flow of the system. This new passive scheme is validated in simulations on locomotion and multi-contact scenarios. The latter involves force applications as a first step toward manufacturing operations.

The results of this thesis have been integrated in the *Stack-of-Tasks* framework of the LAAS-CNRS.

Keywords: Humanoid Whole-Body Control, Humanoids Robots, Force/Torque Control