HAL
open science

Continuous tasks and constraints transitions for the control of robots

Yang Tan

To cite this version:

Yang Tan. Continuous tasks and constraints transitions for the control of robots. Automatic. Université Pierre et Marie Curie - Paris VI, 2016. English. NNT : 2016PA066096 . tel-01310675v2

HAL Id: tel-01310675

https://theses.hal.science/tel-01310675v2

Submitted on 13 Oct 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE DE DOCTORAT

DE l'UNIVERSITÉ PIERRE ET MARIE CURIE

École doctorale Sciences Mécaniques, Acoustique, Électronique et Robotique de Paris

Spécialité

Mécanique - Robotique

Présentée par

Yang TAN

Transitions Continues des Tâches et des Contraintes pour le Contrôle de Robots

---

Soutenance prévue le 14 Mars 2016

Devant le jury composé de :

M. Jean-Yves FOURQUET	Professeur à l'Ecole Nationale d'Ingénieurs de Tarbes	Rapporteur
M. Michel TAIX	Maître de Conférences à l'Université de Toulouse	Rapporteur
M. Faiz BEN AMAR	Professeur à l'Université Pierre et Marie Curie	Examinateur
M. Philippe WENGER	Directeur de recherche à l'IRCCyN, Nantes	Examinateur
M. Alain MICAELLI	Directeur de recherche au CEA/LIST, Fontenay-aux-Roses	Membre invité
M. Philippe BIDAUD	Professeur à l'Université Pierre et Marie Curie	Directeur de thèse
M. Vincent PADOIS	Maître de Conférences à l'Université Pierre et Marie Curie	Encadrant

A dissertation submitted for the degree of

Doctor of Philosophy

of the

Pierre and Marie Curie University

in

Mechanics and Robotics

Presented by

Yang TAN

Continuous Tasks and Constraints Transitions for the Control of Robots

---

To be defended on March 14th***, 2016***

Committee in charge

M. Jean-Yves FOURQUET	Professor at National Engineering School of Tarbes	Referee
M. Michel TAIX	Associate professor at the University of Toulouse	Referee
M. Faiz BEN AMAR	Professor at the Pierre and Marie Curie University	Examiner
M. Philippe WENGER	Research director at IRCCyN, Nantes	Examiner
M. Alain MICAELLI	Research director at CEA/LIST, Fontenay-aux-Roses	Guest member
M. Philippe BIDAUD	Professor at the Pierre and Marie Curie University	Adviser
M. Vincent PADOIS	Associate professor at the Pierre and Marie Curie University	Adviser

Abstract

Large and sudden changes in the torques of the actuators of a robot are highly undesirable and should be avoided during robot control as they may result in unpredictable behaviours. Multi-objective control system for complex robots usually have to handle multiple prioritized tasks while satisfying constraints. Changes in tasks and/or constraints are inevitable for robots when adapting to the unstructured and dynamic environment, and they may lead to large sudden changes in torques. Within this work, the problem of task priority transitions and changing constraints is primarily considered to reduce large sudden changes in torques. This is achieved through two main contributions as follows.

Firstly, based on quadratic programming (QP), a new controller called Generalized Hierarchical Control (GHC) is developed to deal with task priority transitions among arbitrary prioritized task. This projector can be used to achieve continuous task priority transitions, as well as insert or remove tasks among a set of tasks to be performed in an elegant way. The control input (*e.g.* joint torques) is computed by solving one quadratic programming problem, where generalized projectors are adopted to maintain a task hierarchy while satisfying equality and inequality constraints.

Secondly, a predictive control primitive based on Model Predictive Control (MPC) is developed to handle presence of discontinuities in the constraints that the robot must satisfy, such as the breaking of contacts with the environment or the avoidance of an obstacle. The controller takes the advantages of predictive formulations to anticipate the evolutions of the constraints by means of control scenarios and/or sensor information, and thus generate new continuous constraints to replace the original discontinuous constraints in the QP reactive controller. As a result, the rate of change in joint torques is minimized compared with the original discontinuous constraints. This predictive con-

trol primitive does not directly modify the desired task objectives, but the constraints to ensure that the worst case of changes of torques is well-managed.

The effectiveness of the proposed control framework is validated by a set of experiments in simulation on the Kuka LWR robot and the iCub humanoid robot. The results show that the proposed approach significantly decrease the rate of change in joint torques when task priorities switch or discontinuous constraints occur.

Keywords: Redundant Robots, Dynamic Control, Model Predictive Control, Quadratic Programming, Torque-based Control, Continuous Constraints

Contents

1 Introduction **1**
1.1 Motivation . . . 2
1.2 Research questions . . . 4
1.3 Contribution of the Thesis . . . 4
1.4 Structure of the Thesis . . . 7
1.5 Related Publications . . . 8

2 Literature Review **9**
2.1 Problem statement . . . 9
2.1.1 Task definition . . . 11
2.1.2 Constraint definition . . . 12
2.1.3 Control framework . . . 13
2.2 Tasks . . . 17
2.2.1 Trajectory Planning . . . 17
2.2.2 Task Mode Changes . . . 18
2.2.3 Task Transitions . . . 20
2.3 Constraints . . . 23
2.4 Conclusion . . . 24

3 Generalized Hierarchical Control **27**
3.1 Introduction . . . 27
3.2 Generalized hierarchical control . . . 29
3.2.1 Basic Null-space Projector . . . 29
3.2.2 Generalized Projector . . . 31
3.2.3 Generalized Hierarchical Control Framework . . . 35
3.3 Results . . . 37
3.3.1 Insertion and Removal of Tasks . . . 38
3.3.2 Task Transitions Subject to Constraints . . . 40
3.3.3 Comparisons with HQP . . . 42
3.4 Discussion . . . 47
3.5 Conclusion . . . 47

4 Motion Constraint Generation **49**
4.1 Introduction . . . 49
4.2 Obstacle avoidance constraint . . . 52

4.3 Model Predictive Control 56
4.4 Motion constraint generation 58
4.4.1 MPC obstacle avoidance constraint generation 59
4.4.2 Analysis on the parameters of MPC for motion 62
4.5 Results 67
4.6 Conclusion 70

5 Force Constraint Generation **71**
5.1 Introduction 71
5.2 Force Constraint Generation 73
5.2.1 Contact Constraint 74
5.2.2 MPC Force Constraint Generation 75
5.2.3 Sensitivity Analysis of The Proposed MPC Scheme 78
5.3 Control Framework 85
5.4 Results 86
5.4.1 Discontinuous evolution of the force constraint. 86
5.4.2 Standing up from a chair 88
5.4.3 Lifting and putting down the foot 91
5.4.4 Walking 92
5.5 Conclusion 93

6 Conclusions and Perspectives **99**
6.1 Conclusion 99
6.2 Perspectives 102
6.2.1 Toward reduction of the computation cost of GHC 102
6.2.2 Toward intelligent task transitions 102
6.2.3 Toward a generic predictive framework 103

A Appendix **119**

List of Figures

1.1 Overview of the control framework developed for achieving continuous tasks and constraints transitions. Chapter 3 deals with task priority transitions. Chapter 4 focuses on the avoidance of moving obstacles. Chapter 5 concentrates on evolutions of contact states. d_c: obstacle avoidance constraints; F_c: contact constraints; d: distance to obstacles. 5

2.1 Kinematic representation of humanoid systems. The base frame $\mathcal{R}_b$ of the mechanism is free-floating in a reference inertia frame $\mathcal{R}_0$. 9
2.2 The change of $b_i(t)$ in equality constraints can cause change in joint torques. The solution always lies on the constraint $\mathbf{A}\mathbb{X}^* = \boldsymbol{b}$. However, $\boldsymbol{b}(t_{k+1})$ changes from $\boldsymbol{b}(t_k)$, which causes a large change in the solution $\mathbb{X}^*$. 14
2.3 The changes of the dimension of $\boldsymbol{b}$ can cause change in joint torques. . . . 15
2.4 Due to the evolution of task reference, the inequality constraint switches from being inactive to being active, resulting in large instantaneous changes in joint torques. 16
2.5 Hierarchical quadratic programming. 21
2.6 Weighting strategy. 21

3.1 Overview of the methodology developed for achieving continuous tasks and constraints transitions. The features of Generalized Hierarchical Control are highlighted. 29
3.2 Evolution of αs (top) and task errors (bottom) during the insertion and the removal of the end-effector task. The end-effector task (task 1) is firstly inserted to be the task with the highest priority, then removed from the set of tasks. The parameter α_{11} is regulated for the insertion and removal of the end-effector task. The other parameters $\alpha_{12}, \alpha_{21}, \alpha_{13}$, and α_{31} are regulated for handling the priorities between the end-effector task and the other two tasks. α_{23} and α_{32} are constant to maintain the priority of task 2 over the task 3. The insertion and removal of task 1 occurs in a continuous manner and the prescribed priorities are well realized. 39
3.3 The desired and the resulting end-effector trajectory provided by GHC, when the end-effector task has the highest priority. The end-effector moves along the lemniscate-shaped trajectory with an orbital period of 2π seconds. 40
3.4 Task errors using GHC, with the end-effector tracking a lemniscate-shaped trajectory. Desired priority transitions as well as the insertion and removal of the elbow task are achieved. Strict task hierarchies are maintained. . . . 41

3.5 Evolution of joint velocities and joint torques. The upper and lower bounds of $\dot{\boldsymbol{q}}$ are 1.2 rad/s and -1.2 rad/s, respectively. The upper and lower bounds of $\boldsymbol{\tau}$ are 1.5 $N \cdot m$ and -1.5 $N \cdot m$, respectively. These bounds are voluntarily set low in order to easily illustrate the fact that they are satisfied. 41
3.6 Experiment of priority switching. 42
3.7 Task errors using HQP, with fixed task targets. Priority transitions as well as the insertion and removal of the elbow task are performed. The hierarchy rearrangement is instantaneous. 43
3.8 Evolution of αs (top) and task errors (bottom) using GHC, with fixed task targets. Priority transitions as well as the insertion and removal of the elbow task are performed. The hierarchy rearrangement duration is 0.005 seconds. 44
3.9 Evolution of αs (top) and task errors (bottom) using GHC, with fixed task targets. Priority transitions as well as the insertion and removal of the elbow task are performed. The hierarchy rearrangement duration is 2 seconds. 45
3.10 Integration of the absolute values of joint jerks using GHC and HQP, with fixed task targets. The value is increased each time the task hierarchy is changed. GHC generates less amount of joint jerks by performing slower hierarchy rearrangements; while HQP, which performs instantaneous hierarchy rearrangements generates larger joint jerks. 46

4.1 The control framework including a predictive primitive and a QP reactive controller is proposed to minimize large changes in joint torques due to abrupt movements of obstacles. 52
4.2 Description of the obstacle avoidance constraint. 53
4.3 The conception of continuously reacting to sudden movements of obstacles. 55
4.4 Concept of Model Predictive Control. At time step k, an optimal horizon of outputs is computed with respect to the output reference. 56
4.5 The generated continuous position, velocity and acceleration with respect to the discontinuous evolution the obstacle using the MPC with $T = 0.01s$, $NT = 1.5s$ and $\gamma = 10^{-5}$. 61
4.6 The resulting evolution of $\boldsymbol{x}_o^*$ and the percent overshoot using a set of different γs with the prediction horizon $NT = 1.5s$. 63
4.7 The resulting evolution of $\boldsymbol{x}_o^*$ and the percent overshoot using a set of different NTs with the weight $\gamma = 10^{-5}s$. 64
4.8 The generated continuous position, velocity and acceleration with respect to the discontinuous evolution the obstacle with the dynamic weighting matrix $\mathbf{D}$, given $T = 0.01s$, $NT = 1.5s$ and $\gamma = 10^{-5}$. The percent overshoot is effectively reduced. 67
4.9 Snapshot of the robot tracking a trajectory with one hand supported by a table while avoiding collision with a moving object. 68
4.10 The resulting trajectory, right hand force, joint torques and torque derivatives *without* MPC (left) and *with* MPC (right). 69

5.1 The control framework including a predictive control primitive and a QP reactive controller is proposed to minimize the rate of change in joint torques due to changes of contacts states. 73
5.2 Generating continuous evolution of the maximum allowable force using MPC at time k. 75
5.3 The resulting continuous maximum allowable force with respect to the original reference. 77
5.4 Continuous evolutions of the maximum and minimum allowable force are generated simultaneously using the MPC. 78
5.5 Evolution of $\overline{F}$, the maximum force derivative and the approximation error $E(F)$ using a set of different αs with the prediction horizon $NT = 1.5s$. . . 79
5.6 The instant when the MPC begins to reduce $\overline{F}$ with a set of different αs. . . 80
5.7 Evolution of $\overline{F}$, the maximum force derivative and the approximation error E using a set of different NTs with the weight $\alpha = 0.2$. 82
5.8 The evolution of the maximum force derivative and the approximation error E using a set of different αs and NTs with respect to the F_{max} with $5N$ instantaneous changes. 83
5.9 The evolution of the maximum force derivative and the approximation error E using a set of different αs and NTs with respect to the F_{max} with $50N$ instantaneous changes. 84
5.10 The evolution of the maximum force derivative and the approximation error with the instantaneous changes of F_{max}. 85
5.11 Scenario of Kuka performing a force task while satisfying discontinuous force constraint simultaneously. 86
5.12 Simulation results of the discontinuous force constraint *without* MPC smoothing. Large torque derivatives are clearly observed in (c). 87
5.13 Simulation results of the discontinuous force constraint *with* the MPC approach. The large torque derivatives are significantly reduced in (d). . . . 88
5.14 Snapshots of four steps as the robot stands up from a chair. 89
5.15 Evolution of the normal contact force between the robot and the chair with *the baseline approach* (a), *Salini's approach* (b) and *the proposed approach* (c). . . 90
5.16 Evolution of the knee joint torque and its torque derivative with *the baseline approach* (top) *Salini's approach* (middle) and *the proposed approach* (bottom). 94
5.17 Snapshots of lifting and putting down the right foot with 5 steps. 95
5.18 Evolution of normal contact force on feet with *the baseline approach* (a) and *the proposed approach* (b). 95
5.19 Evolution of hip joint torque and its torque derivative with *the baseline approach* (left) and *the proposed approach* (right). 96
5.20 Evolution of hip joint torque and its torque derivative with *the baseline approach* (left) and *the proposed approach* (right). 97

Chapter 1
Introduction

SINCE the birth of the first industrial robot, Unimate, Robotics has grown tremendously not only in computer and sensor technology, but also in control theory. These advances enable robots to be more dexterous and intelligent by taking advantage of their sophisticated mechanism and by means of visual, tactile and auditory perceptions. Nowadays, robots have entered all areas of people's lives with applications ranging from historical industrial manufacturing robots to medical devices including filed intervention machines. To realize these applications, control plays an irreplaceable role. In robotics, the purpose of control is to make the robot achieve desired behaviours accurately and in a stable way. Therefore, a controller is usually designed with feedback: the current behaviour of the robot, treated as the output, is compared with the desired one and their difference is employed by the controller to compute new control inputs, which bring the robot towards the desired behaviours.

The achievement of these desired behaviours is motivated by objectives to be reached. Correspondingly the controller is employed to drive the robotic system to reach its desired behaviours. In Robotics, objectives are commonly related to the environment and task functions for example are usually used to establish a link between the robot and its objectives, which change according to the evolution of the environment over time. Depending on different objectives, tasks are specified in different forms, *e.g.* tracking, grasping, manipulating, pushing, postural balance, etc.

Additionally, the robotic system and its applications are generally subject to constraints, issued form safety, specification or intrinsic limitations. The behaviours of the robot towards its objectives must comply with these constraints, and the control inputs

should therefore not induce violations. For example, when a robot tracks a trajectory towards its objective, several constraints can be accounted for: 1) a safety constraint can prevent the robot from colliding any obstacle in the environment, 2) the specification of its behaviour could impose a maximum allowable velocity on its end-effector, and 3) the range of joint angle physically restricts the robotic system.

For performance and safety reasons, the controller must be able to account for the task objectives and constraints concurrently. The control problem for a robotic system is to solve the actuator inputs either in term of joint positions, joint velocities or joint torques according to the controller. These actuator inputs then drive the robot towards its task objectives while satisfying its constraints. In Robotics, reactive controllers are used in most robot control architectures. They are in charge of converting operational references (task objectives) into actuator inputs at each time step. This feature of reactive controllers provides robots with real-time capabilities for precise and fast operations in a dynamic environment. Many algorithms are successfully built to achieve task objectives at different levels while satisfying multiple constraints. Their achievements mainly focus on the resolution and solvability of the control problem at each time step. In practice, the task objectives and constraints usually evolve over time due to complex behaviours and the dynamic environment. The actuator inputs thus have to be solved during the whole timeline until the robot reaches the desired behaviours. However, the continuity of the retained solution over time is rarely studied. In other words, large instantaneous changes in actuator inputs may come out with control instabilities when solving the control problem over time. Therefore, the goal of this thesis is to develop a control framework to prevent large instantaneous changes in actuator inputs over time.

1.1 Motivation

When solving the control problem, large changes in actuator inputs should be avoided, because they can produce undesired effects to the robotic system. First of all, large changes in actuator inputs may cause potential control instabilities and bad control performance. The resulting large changes in feedback of the controller can cause oscillations

of actuator inputs, which jeopardize the stability of the controller. Furthermore, these oscillations are directly shown on the robot behaviours. Vibrations are induced when the robot attempts to track its task objectives. Traditionally, robots are exploited for typical applications in structured industrial environments, such as pick and place, welding, painting and assembly. These applications require stable and precise motions of the robot. Nevertheless, vibrations induced by large changes in actuator inputs can degrade the accuracy of the motion and thus the workpiece may be destroyed, resulting in direct economic losses. Moreover, these vibrations can cause damage to the robot mechanical structures. Therefore, in order to avoid this kind of instability of the controller, it is very important and necessary to prevent large changes in actuator inputs when solving the control problem.

Many causes can result in large changes in actuator inputs when solving the control problem over time. Among these causes, tasks and constraints are two essential elements and have to be considered when formulating the controller. For a robot operating in a complex and dynamic environment including humans, the controller is required to have a great adaptability to react to sudden changes of task objectives and constraint states. However, these instantaneous changes can lead to large changes in actuator inputs if they are not properly dealt with.

Moreover, the robot redundancy[1] offers great flexibility, dexterity and versatility to adapt to the dynamic environment by preforming complex behaviours involving the simultaneous performance of multiple tasks. However, if the tasks are in conflict, the solution of the control problem may lead the robot to a state, where none of the task objectives are preformed. To handle this conflict, multiple tasks are regulated in the controller in two ways. The first one is the strict task hierarchy, which ensures that critical tasks are fulfilled with higher priorities and lower priority tasks are performed only in the null-space[2] of higher priorities tasks. The second one is non-strict task hierarchy, the solution of which is a compromise among task objectives with different weights. A lower priority task is not restricted in the null-space of higher priority tasks and thus it may still affect

[1]Redundancy occurs when a robot possesses more degrees of freedom than the minimum number required to execute a given task [Siciliano 90].

[2]The null-space of an m-by-n matrix $\mathbf{A}$ is defined by $null(\mathbf{A}) = \{\boldsymbol{x} \in \mathbb{R}^n : \mathbf{A}\boldsymbol{x} = \mathbf{0}\}$ [Golub 12, p. 51].

their performances. For the robot acting in a dynamically changing environment, task priorities are required to be changed among a set of tasks in order to cope with changing situations. These changes of task priorities are called *task transitions*, which can also produce large changes in actuator inputs when solving the control problem. Therefore, it is necessary and essential to propose a control framework, which enables the robot to continuously react to these instantaneous changes of task objectives and constraints as well as the task transitions without resulting in large changes in actuator inputs.

In summary, large changes in actuator inputs can jeopardize the stability of the control performance. Many causes related to tasks and constraints in the controller frequently result in large changes in actuator inputs. Therefore, to prevent large instantaneous changes in actuator inputs is necessary for robots to safely and stably adapt to the dynamic environment.

1.2 Research questions

The primary objective of this thesis is to develop a general control framework for robots to prevent large instantaneous changes in actuator inputs caused by suddenly changing tasks and constraints.

The following research questions are considered in this thesis to accomplish the primary research objective:

- How can the multiple task objectives be formulated in the controller such that the task transitions can be continuously achieved among a set of tasks?
- How can the instantaneous changes of constraint states be minimized in a generic way to prevent large changes in actuator inputs?

1.3 Contribution of the Thesis

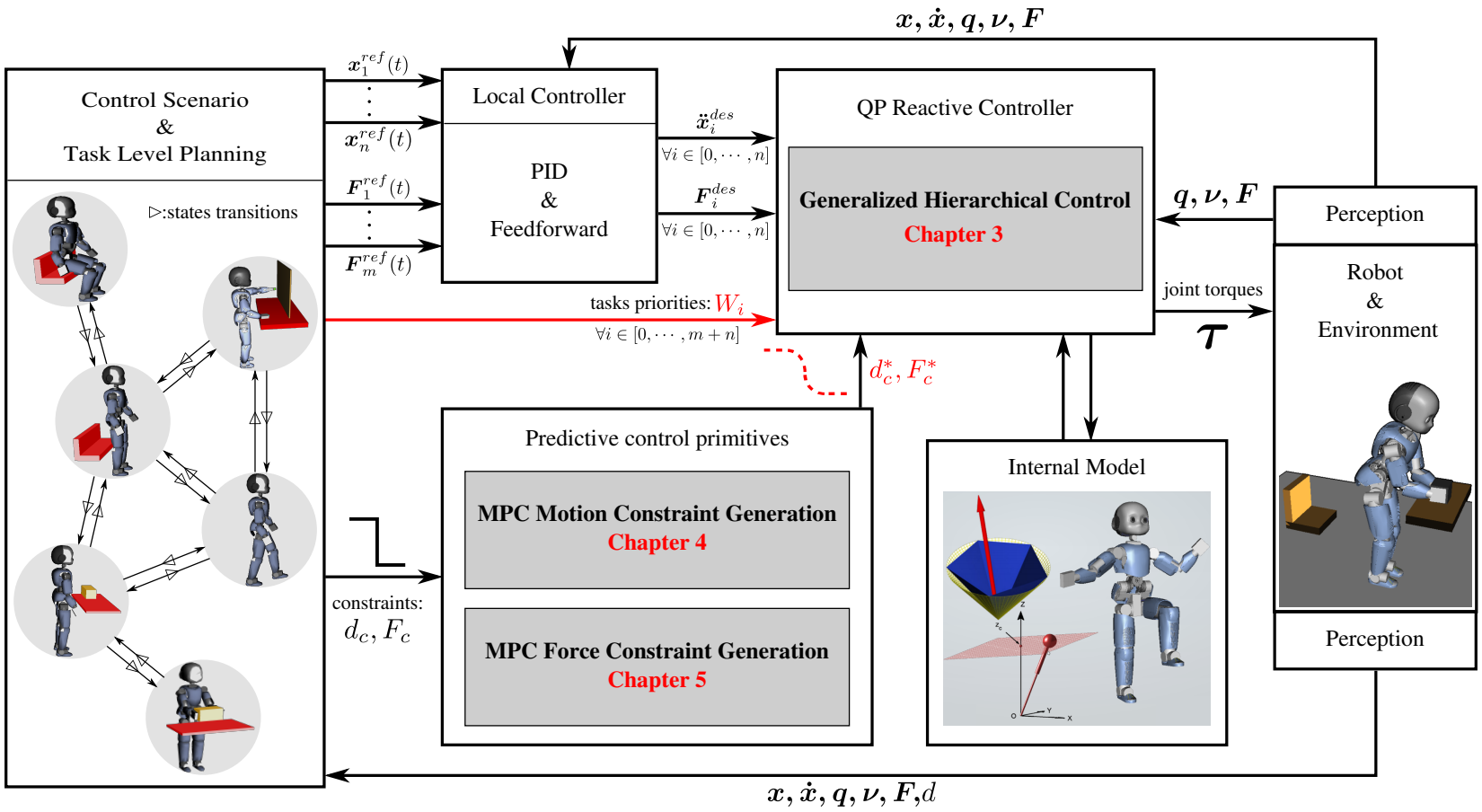


Figure 1.1: Overview of the control framework developed for achieving continuous tasks and constraints transitions. Chapter 3 deals with task priority transitions. Chapter 4 focuses on the avoidance of moving obstacles. Chapter 5 concentrates on evolutions of contact states. d_c: obstacle avoidance constraints; F_c: contact constraints; d: distance to obstacles.

In this thesis, the study on the modelling and controlling a redundant robotic system in dynamic environment is presented. A novel control framework to achieve continuous task and constraints transitions is proposed as illustrated in Figure 1.1. Based on the control scenario and high-level task planning, the changes of tasks priorities are determined. Using this information, the Generalized Hierachical Control can correspondingly achieve continuous task transitions. But it cannot handle the problem of changing constraints. In this case, predictive control primitives are employed to generate continuous obstacle avoidance constraints and continuous contacts constraints according to the information from control scenario, high-level task planning and sensors. Therefore, the composition of a reactive control approach with a predictive formulation allows the controller to ensure the performance of the robot while minimizing the rate of change in joint torques. The main contributions of this thesis are as follows:

Generalized Hierarchical Control. With the aim of preventing large changes in joint torques due to task transitions, a *generalized projector* is developed to regulate to what extent a lower priority task is projected into the null-space of a higher-priority task. In other words, this generalized projector allows a task to be completely, partially, or not at all projected into the null-space of some other tasks. Based on this projector, Generalized Hierarchical Control is proposed to account for an arbitrary number of strict and non-strict task hierarchies. More importantly, it can achieve continuous task transitions among arbitrary prioritized tasks. It also provides an elegant way of inserting and removing tasks among a set of prioritized tasks.

Predictive control primitives. A predictive control scheme to minimize the rate of change in joint torques over time is proposed in the presence of suddenly changing constraint, particularly including contact constraints and obstacle avoidance constraints. In some situations, the constraint is sensed in real time, but the robot may have a certain time to react to this constraint instead of responding to it immediately. This enables the robot to predict its future states with sensed information. In some situations, sudden changes in constraints can be known from the control scenario or the task planning in advance, for example, the constraint from the desired motion scenario for the system can be anticipated, such as when a robot is about to sit on a chair or to put a foot on the

ground. Model Predictive Control can thus preview changes in the constraint in advance using a receding horizon and produce a new constraint for the QP reactive controller, resulting in minimized changes in joint torques. One key feature of this approach is that the constraints of the reactive controller are modified rather than the task objectives. As such, the proposed method neither adds troubles of modifying or replanning the task objectives, nor complicates the regulation of multiple tasks by introducing extra tasks to satisfy constraints. But the proposed method ensures that changes in joint torques are minimized. Therefore, it is a very generic approach which can be applied independently from the way the control law is formulated.

The approaches developed in this thesis are validated by simulations that are carried out on the simulator Arboris-python [Salini 12b] with the virtual Kuka LWR robot and iCub robot.

1.4 Structure of the Thesis

This thesis contains six chapters to present the contributions towards the two identified research questions (see in Figure 1.1). The remainder of the thesis is organized as follows.

Chapter 2 recalls the dynamic model of the robot as well as its task and constraint definition, introduces the causes of large changes in actuator inputs and reviews the state-of-the-art in the robot control that are relevant to the contributions of the thesis.

Chapter 3 proposes the Generalized Hierarchical Control framework to achieve continuous task transitions among a set of prioritized tasks. Based on the null-space projector, a generalized projector is developed to deal with different priorities of multiple tasks simultaneously and perform task transitions continuously. The effectiveness of this approach is demonstrated on a simulated robotic manipulator (Kuka LWR robot) in a dynamic environment.

Chapter 4 develops a predictive control primitive, which integrates Model Predictive Control to a reactive controller, to minimize the rate of change in joint torques due to movements of obstacles. MPC previews the position of the moving obstacles based on sensed information and generates virtual continuous position, velocity and acceleration

profiles of the obstacle for obstacle avoidance constraints, which is used in the QP reactive controller. The effectiveness of the proposed approach is demonstrated by the simulation results on the iCub robot.

Chapter 5 presents the approach to minimize the rate of change in joint torques due to the changes of contact states. MPC previews the evolution of contact states in time and generates continuous allowable force for the QP reactive controller. The effects of the proposed approach are simulated for different scenarios on the iCub robot. The results show that the proposed approach significantly reduce the changes in joint torques at the instant when contacts are broken or established.

Chapter 6 concludes the thesis and presents future directions.

Appendix A provides the proof of the maintenance of strict hierarchies represented by standard lexicographic orders subject to constraints.

1.5 Related Publications

The novel developments introduced in this work have been peer-reviewed and validated with the publication of the contributions listed hereafter:

Journal articles

- M. Liu, Y. Tan and V. Padois, "Generalized Hierarchical Control", *Autonomous Robots*, vol.40, issue 1, pp. 17–31, 2016.

Conference publications

- Y. Tan, D. Lau, M. Liu, P. Bidaud and V. Padois, "Minimization of the rate of change in torques during motion and force control under discontinuous constraints", *ROBIO2015*, 2015.

- Y. Tan, D. Lau, M. Liu, P. Bidaud and V. Padois, "Minimization of the rate of change in torques during contact transitions for humanoids", *ECC2016*, 2015, submitted.

Chapter 2
Literature Review

The work presented in this thesis aims at developing a control framework that prevents large instantaneous changes in joint torques. In this chapter, a study on the state of the art in the robot control that relates to the large instantaneous changes in joint torques is presented. Section 2.1 investigates the modelling of robotic system and classical control methods to highlight the causes of large sudden changes in joint torques. Section 2.2 reviews different applications that solve the problem caused by the changes of the robot tasks. Section 2.3 presents recent techniques to deal with equality and inequality constraints and avoid large changes in joint torques due to these constraints. Finally, Section 2.4 summarizes the literature review and identifies key issues that have not yet been resolved.

2.1 Problem statement

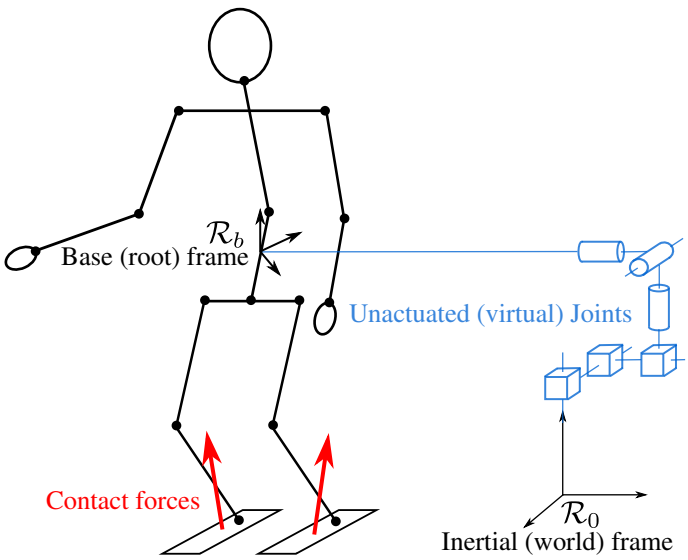


Figure 2.1: Kinematic representation of humanoid systems. The base frame $\mathcal{R}_b$ of the mechanism is free-floating in a reference inertia frame $\mathcal{R}_0$.

Large instantaneous changes in joint torques should be avoided because they can cause potentially undesired effects: 1) vibrations; 2) bad control performances. The problem of robot control can be formulated as a computation of the joint torques which drive the actuators so as to guarantee the execution of the robot task. Modelling the robotic system is generically the first step. For a free-floating robotic structure, the position and orientation of a *base frame* $\mathcal{R}_b$ attached to the robot body are measured with respect to the fixed inertial frame $\mathcal{R}_0$, called the *world frame*. Figure 2.1 illustrates this presentation by showing a 6-DoF virtual joint, which links the base frame to the inertial frame. The equation of motion for such systems can be derived from the Euler-Lagrange formalism and expressed as [Murray 94]:

$$\underbrace{\begin{bmatrix} \mathbf{M}_b & \mathbf{M}_{bj} \\ \mathbf{M}_{bj}^T & \mathbf{M}_j \end{bmatrix}}_{\mathbf{M}(\boldsymbol{q})} \underbrace{\begin{bmatrix} \dot{\boldsymbol{\nu}}_b \\ \ddot{\boldsymbol{q}}_j \end{bmatrix}}_{\dot{\boldsymbol{\nu}}} + \underbrace{\begin{bmatrix} \boldsymbol{n}_b \\ \boldsymbol{n}_j \end{bmatrix}}_{\boldsymbol{n}(\boldsymbol{q},\boldsymbol{\nu})} = \mathbf{S}\boldsymbol{\tau} + \mathbf{J}_c(\boldsymbol{q})^T \boldsymbol{F}_c \,, \tag{2.1}$$

with

- $\boldsymbol{q} \in SE(3) \times \mathbb{R}^n$: the generalized coordinates that parametrizes the configuration of the free-floating system with n joints.
- $\boldsymbol{q}_j, \dot{\boldsymbol{q}}_j \in \mathbb{R}^n$: the joint configurations and the joint velocities, respectively.
- $\boldsymbol{\nu} \in \mathbb{R}^{n+6}$: the system velocities, concatenating the floating-base twist $\boldsymbol{\nu}_b \in SE(3)$ and the joint velocities $\dot{\boldsymbol{q}}_j$.
- $\mathbf{M}(\boldsymbol{q}) \in \mathbb{R}^{(n+6)\times(n+6)}$: the generalized inertia matrix. Indices $\bullet_b, \bullet_j$ and $\bullet_{bj}$ denote their definition with respect to the base, the joints and both, respectively.
- $\boldsymbol{n}(\boldsymbol{q}, \boldsymbol{\nu}) \in \mathbb{R}^{n+6}$: the vector of Coriolis, centrifugal and gravity terms.
- $\mathbf{S} = [\mathbf{0}_{n\times 6} \;\; \mathbf{I}_{n\times n}]^T$: the actuated joint selection matrix. In this work, each joint is assumed to be driven by only one actuator.
- $\boldsymbol{\tau} \in \mathbb{R}^n$: the vector of actuation joint torques.
- $\mathbf{J}_c(\boldsymbol{q}) = [\mathbf{J}_{c,\,1}(\boldsymbol{q})^T \;\; \dots \;\; \mathbf{J}_{c,\,n_c}(\boldsymbol{q})^T]$: the Jacobian matrix for all n_c contact points.

- $\boldsymbol{F}_c = \left[\boldsymbol{F}_{c,\,1}^T \;\ldots\; \boldsymbol{F}_{c,\,n_c}^T\right]^T$: the vector of external contact wrenches.
- $\mathbb{X} = \left[\dot{\boldsymbol{\nu}}^T, \boldsymbol{\tau}^T, \boldsymbol{F}_c^T\right]^T$: the dynamic variable of the system.

2.1.1 Task definition

Robotic behaviors are generically defined as objectives to be reached. A robot objective could be for example tracking a trajectory in operational space or in joint space, or applying a certain force on an object. Commonly these objectives are decomposed into a set of intermediate targets to reach over time, ultimately leading to the goal objectives. To achieve this, a generic task function $T(\mathbb{X}_t, t)$ can be defined as an error between a desired acceleration/force and the actual acceleration/force of the robot at any time t. The task function on one hand measures performances of robots, on the other hand it is used to compute the control actions to drive the robot towards the objectives [Samson 91]. For example, several task functions are defined in this thesis:

- Operational space acceleration $\quad T(\mathbb{X}_t, t) = \mathbf{J}(\boldsymbol{q}_t)\dot{\boldsymbol{\nu}}_t + \dot{\mathbf{J}}(\boldsymbol{q}_t, \boldsymbol{\nu}_t)\boldsymbol{\nu}_t - \ddot{\boldsymbol{x}}_t^d$
- Joint space acceleration $\quad T(\mathbb{X}_t, t) = \dot{\boldsymbol{\nu}}_t - \dot{\boldsymbol{\nu}}_t^d$ (2.2)
- Operational space force $\quad T(\mathbb{X}_t, t) = \boldsymbol{F}_{c,t} - \boldsymbol{F}_{c,t}^d$

where $\ddot{\boldsymbol{x}}$ is the acceleration of a frame attached to the robot in Cartesian space. The subscript $_t$ means the time-variant variable and its value at time t. The superscript ^{d} refers to the desired acceleration/force, which can be defined by a proportional derivative control. For instance, the desired acceleration is :

$$\ddot{\boldsymbol{x}}_t^d = \ddot{\boldsymbol{x}}_t^{goal} + K_p \boldsymbol{\epsilon}_t + K_d \dot{\boldsymbol{\epsilon}}_t \tag{2.3}$$

where $\boldsymbol{\epsilon}_t$ is the pose error at time t, which can be trivially computed when dealing with position, or needs to resort to a non-minimal representation of the orientation such as quaternions [Altmann 05]. K_p and K_d are the proportional and derivative gains, respectively. The superscript ^{goal} indicates the position, velocity and acceleration wanted for the body or joint.

2.1.2 Constraint definition

While performing the task, the robot also has to satisfy constraints simultaneously. Therefore, when solving the control problem, constraints have to be taken into account in order to ensure that the control actions will not induce violations at any time. Generally, robot constraints are either related to some intrinsic limitations of robots, such as joint limits and actuation capacities, or to the surrounding environment, such as obstacles to avoid and contacts to maintain. Joint limits, which are determined by the mechanisms of the robot, are bounds on the joint position. They can be written $\boldsymbol{q}_j \in [\boldsymbol{q}_{j,min},\ \boldsymbol{q}_{j,max}]$. The property of actuators determines actuation capacities, which consist of joint velocity limits ($\dot{\boldsymbol{q}}_j \in [\dot{\boldsymbol{q}}_{j,min},\ \dot{\boldsymbol{q}}_{j,max}]$), and joint torque limits ($\boldsymbol{\tau} \in [\boldsymbol{\tau}_{min},\ \boldsymbol{\tau}_{max}]$). However, joint limits and joint velocity limits have no exact relation with the dynamic variable $\mathbb{X}$. Based on a discrete linear approximation with a time step δt, joint limits and actuation capacities can be expressed as follows:

$$\begin{aligned} &\boldsymbol{q}_{j,min} \leq \boldsymbol{q}_{j,t_k} + \dot{\boldsymbol{q}}_{j,t_k}\delta t + \ddot{\boldsymbol{q}}_{j,t_k}\tfrac{\delta t^2}{2} \leq \boldsymbol{q}_{j,max} \\ \leftrightarrow &\begin{bmatrix} 0 & \mathbf{I} & 0 \\ 0 & -\mathbf{I} & 0 \end{bmatrix} \mathbb{X}_{t_k} \leq \tfrac{2}{\delta t^2} \begin{bmatrix} \boldsymbol{q}_{j,max} - (\boldsymbol{q}_{j,t_k} + \dot{\boldsymbol{q}}_{j,t_k}\delta t) \\ -(\boldsymbol{q}_{j,min} - (\boldsymbol{q}_{j,t_k} + \dot{\boldsymbol{q}}_{j,t_k}\delta t)) \end{bmatrix}, \end{aligned} \tag{2.4}$$

$$\begin{aligned} &\dot{\boldsymbol{q}}_{j,min} \leq \dot{\boldsymbol{q}}_{j,t_k} + \ddot{\boldsymbol{q}}_{j,t_k}\delta t \leq \dot{\boldsymbol{q}}_{j,max} \\ \leftrightarrow &\begin{bmatrix} 0 & \mathbf{I} & 0 \\ 0 & -\mathbf{I} & 0 \end{bmatrix} \mathbb{X}_{t_k} \leq \tfrac{1}{\delta t} \begin{bmatrix} \dot{\boldsymbol{q}}_{j,max} - \dot{\boldsymbol{q}}_{j,t_k} \\ -(\dot{\boldsymbol{q}}_{j,min} - \dot{\boldsymbol{q}}_{j,t_k}) \end{bmatrix}, \end{aligned} \tag{2.5}$$

$$\begin{aligned} &\boldsymbol{\tau}_{min} \leq \boldsymbol{\tau}_{t_k} \leq \boldsymbol{\tau}_{min} \\ \Leftrightarrow &\begin{bmatrix} 0 & \mathbf{I} & 0 \\ 0 & -\mathbf{I} & 0 \end{bmatrix} \mathbb{X}_{t_k} \leq \begin{bmatrix} \boldsymbol{\tau}_{max} \\ -\boldsymbol{\tau}_{min} \end{bmatrix}, \end{aligned} \tag{2.6}$$

where $\boldsymbol{q}_{j,t_k}, \dot{\boldsymbol{q}}_{j,t_k}, \ddot{\boldsymbol{q}}_{j,t_k}$ and $\boldsymbol{\tau}_{t_k}$ are the joint position, joint velocities, joint accelerations and joint torques at time t_k. Dynamic variable $\mathbb{X}_{t_k}$ is constrained at time t_k . Additionally, when the robot makes contact with its environment, the contact existence conditions for each contact point can be expressed according to the Coulomb friction model:

$$
\begin{aligned}
C_{j,t_k}\boldsymbol{F}_{c,j,t_k} &\leq 0\,,\ \forall j \in [1,\ldots,n_c] \\
\mathbf{J}_{c,j}(\boldsymbol{q}_{t_k})\dot{\boldsymbol{\nu}}_{t_k} + \dot{\mathbf{J}}_{c,j}(\boldsymbol{q}_{t_k},\boldsymbol{\nu}_{t_k})\boldsymbol{\nu}_{t_k} &= 0\,,\ \forall j \in [1,\ldots,n_c]
\end{aligned}
\tag{2.7}
$$

where j means the jth contact point, C_{j,t_k} the corresponding linearized friction cone at time t_k [Siciliano 08], and F_{c,j,t_k} the contact force. For the sake of clarity, the above definitions of constraints can be expressed in the form of equality and in the form of inequality:

$$\mathbf{A}(t)\mathbb{X}_t = \boldsymbol{b}(t)\,, \tag{2.8}$$

$$\mathbf{G}(t)\mathbb{X}_t \leq \boldsymbol{h}(t)\,. \tag{2.9}$$

2.1.3 Control framework

Common task realization methods are based on automatic techniques and reactive approaches with feedback control [Franklin 94]. The joint torques are computed to draw the robot towards the targeted task objectives in these approaches. It is noted that task objectives and constraints are generally time-dependent. Therefore, a sequence of joint torques over time needs to be solved, until the robot reaches the goal objectives. Given one task, robot dynamic equation and constraints, the computation of the joint torques at time t can be formulated as follows:

$$\boldsymbol{\tau}_t^* = \underset{\mathbb{X}_t}{\arg\min} \quad T(\mathbb{X}_t, t) \tag{2.10a}$$

$$\text{subject to} \quad \mathbf{M}(\boldsymbol{q}_t)\dot{\boldsymbol{\nu}}_t + \boldsymbol{n}(\boldsymbol{q}_t,\boldsymbol{\nu}_t) = \mathbf{S}\boldsymbol{\tau}_t + \mathbf{J}_c(\boldsymbol{q}_t)^T\boldsymbol{F}_{c,t} \tag{2.10b}$$

$$\mathbf{A}(t)\mathbb{X}_t = \boldsymbol{b}(t) \tag{2.10c}$$

$$\mathbf{G}(t)\mathbb{X}_t \leq \boldsymbol{h}(t) \tag{2.10d}$$

In equation (2.10), the task function $T(\mathbb{X}_t, t)$ is the objective to be minimized and computed joint torques must conform to dynamic model (2.10b) as well as satisfy constraints (2.10c)-(2.10d) at time t. Many mathematical techniques can be used to solve this equation to attain $\boldsymbol{\tau}_t^*$. In practical applications, the equation (2.10) is commonly solved over the entire execution timeline and a sequence of joint torques over time ($\boldsymbol{\tau}_t^*, t \in [t_0,\ldots,t_k,\ldots t_{end}]$)

is figured out to drive the the robot to the goal objectives. In other words, joint torques evolve with time. In reality, the computed time-variant joint torques $\boldsymbol{\tau}_t^*$ may be discontinuous[1], or even undergo large instantaneous changes, which is undesirable. However, when solving a sequence of the control problem (2.10) over time, discontinuities or large instantaneous changes in joint torques may occur under following situations:

- The desired task objective in $T(\mathbb{X}, t)$ is discontinuous, which is related to large changes of task goals or switches between the motion task and the force task. For example, the target position in Cartesian space $\boldsymbol{x}_t^{goal}$ undergoes large changes from t_k to t_{k+1} to adapt to the dynamic environment.

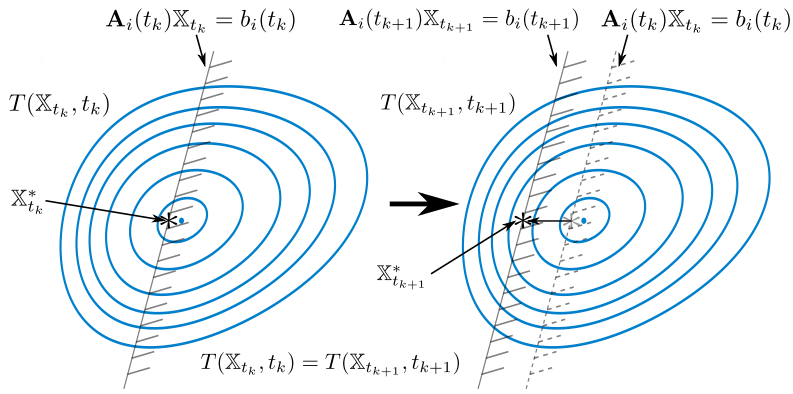


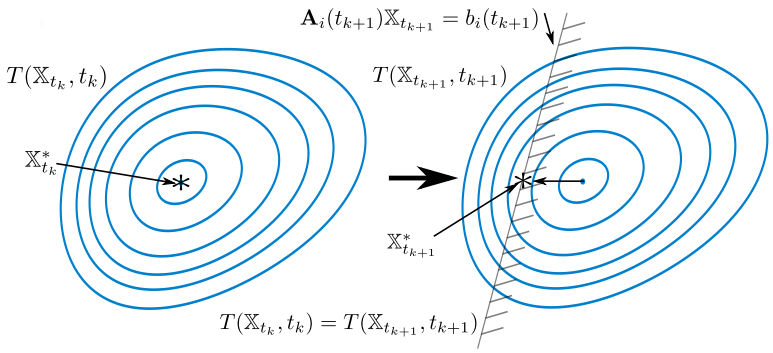
Figure 2.2: The change of $b_i(t)$ in equality constraints can cause change in joint torques. The solution always lies on the constraint $\mathbf{A}\mathbb{X}^* = \boldsymbol{b}$. However, $\boldsymbol{b}(t_{k+1})$ changes from $\boldsymbol{b}(t_k)$, which causes a large change in the solution $\mathbb{X}^*$.

- Equality constraints (2.10c) are active at all feasible solutions[2]. Assuming that the task objective is constant and $\mathbf{A}(t)$ is continuous, two possible cases may result in large instantaneous changes in joint torques. 1) Any $b_i(t)$ in $\boldsymbol{b}(t) = [b_1(t) \dots b_m(t)] \in \mathbb{R}^m$ has large changes over time. This means that $b_i(t)$ is discontinuous. For example, Figure 2.2 illustrates the fact that from t_k to t_{k+1} the large change of $b_i(t)$ can lead to large change in joint torques. 2) The dimension of $\boldsymbol{b}$ increases or decreases

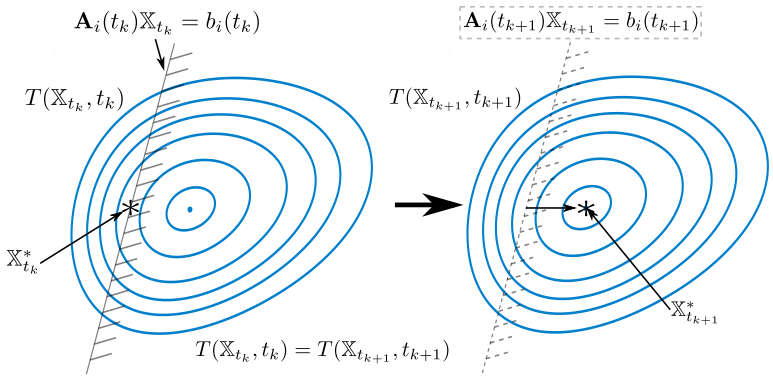
[1]In mathematics, a *continuous* function is a function for which small changes in the input result in small changes in the output. Otherwise, a function is said to be a *discontinuous* function. (From http://mathworld.wolfram.com/ContinuousFunction.html)

[2]If $\mathbb{X}_t$ is feasible and $\boldsymbol{G}_i(t)\mathbb{X}_t = h_i(t)$ ($\boldsymbol{G}_i(t)$ is the row vector of the ith row of $\mathbf{G}$), we say the ith inequality constraint $\boldsymbol{G}_i(t)\mathbb{X}_t \leq h_i(t)$ is active at $\mathbb{X}_t$. If $\boldsymbol{G}_i(t)\mathbb{X}_t < h_i(t)$, we say the constraint $\boldsymbol{G}_i(t)\mathbb{X}_t \leq h_i(t)$ is inactive [Boyd 04, p. 128].

with additions and removals of equality constraints over time, which is shown in Figure 2.3.



(a) A new equality constraint is added at t_{k+1}, which requires that the solution $\mathbb{X}^*$ must lie on this constraint. This can cause a large change in the solution.



(b) An equality constraint is removed at t_{k+1}. The solution $\mathbb{X}^*$ may undergo a large change.

Figure 2.3: The changes of the dimension of $\boldsymbol{b}$ can cause change in joint torques.

- Assuming $\mathbf{G}(t)$ in (2.10d) is continuous, there are also two possible situations where large instantaneous changes in joint torques are related to inequality constraints (2.10d). 1) When the inequality constraint is active, it is equivalent to the equality constraints. It can cause large instantaneous changes in joint torques in the same way equality constraints do. 2) The solution changes from being on the interior of the feasible set of (2.10d) to the boundary, or vice versa (see in Figure 2.4). In other words, any of the constraints $\boldsymbol{G}_i(t)\mathbb{X}_t \leq h_i(t)$ switches between being inactive and being active.

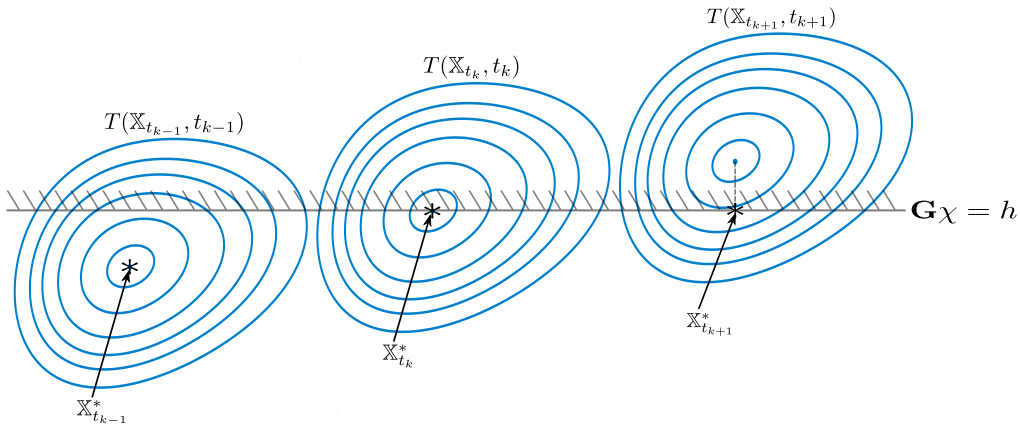


Figure 2.4: Due to the evolution of task reference, the inequality constraint switches from being inactive to being active, resulting in large instantaneous changes in joint torques.

To benefit from the redundancy of the system, multiple tasks could be assigned and resolved simultaneously. However, task conflicts may occur when none of task objectives can be satisfied simultaneously. In order to handle task conflicts, a priority level is attached to each task so that the robot has the ability to realize the most important task. In this case, the objective (2.10a) of the control problem (2.10) can be formulated as a function of n_t tasks with their priority $\lambda_i(t)$:

$$T(\mathbb{X}_t, t) = f(\lambda_1(t),\ T_1(\mathbb{X}_t, t),\ \lambda_2(t),\ T_2(\mathbb{X}_t, t),\ \ldots,\ \lambda_{n_t}(t),\ T_{n_t}(\mathbb{X}_t, t))\,. \tag{2.11}$$

When solving the control problem of multiple tasks (2.11) over time, any discontinuous task function $T_i(\mathbb{X}_t, t)$ may result in large instantaneous changes in joint torques. Apart from this, large instantaneous changes in joint torques may occur due to task rearrangements among a set of tasks:

- Any task $T_i(\mathbb{X}_t, t)$ may be added into or removed from a set of tasks at any time t.
- The priority $\lambda_i(t)$ of a task $T_i(\mathbb{X}_t, t)$ may be changed abruptly in order to adapt to changing situations.

These sudden task transitions can lead to large instantaneous changes in joint torques.

As stated above, large sudden changes in task goals, constraint evolution or sudden task transitions can result in large instantaneous changes in joint torques independently

and respectively. Different techniques for handling large sudden changes in task goals, constraint evolution and multiple prioritized tasks are reviewed in the following section.

2.2 Tasks

Tasks are the core of the robot controller and they directly affect the continuity of the joint torques. In practice, tasks may go through discontinuities as stated above, such as task goal changes, task mode changes and task transitions, in order to adapt to a dynamic environment. For example, the task target may change due to moving targets, or avoiding collisions with obstacle on the path. Some tasks require the controller to switch between free motion control and constrained force control to achieve the desired target. In some cases, task transitions are needed to perform a complex mission. All of these discontinuous task changes may lead to large instantaneous changes in joint torques. Different techniques have been proposed to deal with task goal changes, task mode changes and task transitions, respectively.

2.2.1 Trajectory Planning

Trajectory planning is a fundamental issue for robotic applications in general. The goal of trajectory planning is to generate the reference inputs to the robot control system that ensures the implementation of desired motions. It can be assumed that a trajectory planner takes the path description, the path constraints and the constraints imposed by the robot dynamics as inputs, whereas the output is the trajectory of the joints, or of the end-effector, expressed in terms of a sequence of values of position, velocity and acceleration over time [Gasparetto 12]. In order to ensure the tracking accuracy and the control stability, it is essential to generate smooth trajectories, *i.e.*, trajectories with a high order of continuity like C^2 continuous[3]. The vibrations induced by non-smooth trajectories can degrade the tracking performance of the trajectory.

[3]A function with k continuous derivatives is called a C^k function. For example, $f(x)$ is a two times differentiable function. If $f''(x)$ is continuous, $f(x)$ is said to be C^2 continuous. From http://mathworld.wolfram.com/C-kFunction.html

To realize smooth trajectories, a common way is to parametrize the path by using functions that are at least C^2 continuous, *i.e.*, continuous evolutions of accelerations. Cubic splines are widely used for path parametrization with time as the cost function since they can assure the continuity of velocity and acceleration [Lin 83]. In [Lambrechts 04], a fourth order trajectory planner is proposed to generate smooth trajectories. A quintic polynomial for representing the entire trajectory is used to provide a smooth trajectory [Andersson 89]. Although polynomial can easily produce smooth trajectory with continuous acceleration, higher order polynomials can result in large oscillations of the trajectory [Schatzman 02].

Another way of smoothing trajectories is to minimize the jerk (the derivative of the acceleration). A large jerk adversely affects the smoothness of the trajectory [Kyriakopoulos 88]. The approach described in [Bobrow 85, Macfarlane 03, Herrera-Aguilar 06] introduces a time-optimal, jerk-limited trajectory planner to generate a continuous acceleration trajectory, subject to the maximal velocity, acceleration and jerk in both Cartesian and joint space. However, in practice the bounds on the jerk and the limits on the torque derivatives are difficult to define and choose. Some methods successfully generate smooth trajectories by minimizing the jerk in an optimization problem. In [Simon 93], the integral of squared jerk is minimized on the trajectory execution time along the set of via-points and thus a smooth trajectory is generated. The method in [Piazzi 00] globally minimizes the maximum absolute value of the jerk along a trajectory, the execution time of which is known a priori. The produced trajectory shows low maximum absolute jerk value.

In short, different approaches can produce a smooth trajectory, which means that the task references in $T(\mathbb{X}_t, t)$ can be continuous. However, this is only a necessary condition for preventing large instantaneous changes in joint torques, because trajectory planning does not solve the problem due to changes of task mode and task transitions.

2.2.2 Task Mode Changes

In practice, robots are required to change from one mode to the other readily to achieve a single task, for example, an assembly task. In this case, the controller has to switch

from free motion control to constrained force control. This task mode change has the significant problem of the impact force [Youcef-Toumi 89]. These impact forces may be very large and can yield large changes in joint torques, which result in degraded performances or instabilities. In order to solve this problem, the control of the noncontact-to-contact transition has been studied for many years. Basically, the existing methods can be divided into three main categories.

The first approach is to exploit the kinematic redundancy of the robot to reduce the impact force. An impact model is created to estimate the magnitude of the impact force [Walker 90]. In this model, the impact force is determined by the pre-contact velocity, stiffness and compliance of the object, and the configuration of the robot. Within the impact model, the approach in [Pagilla 01, Padois 07] decreases the velocity normal to the constrained surface to zero in order to reduce the impact force. In [Brufau 05], the redundancy of the robot is used to diminish the impact force by reducing the inertia of the robot, which is seen at the end-effector, in the normal direction to the contact surface.

The second one is impedance control [Hogan 85]. The external environment is treated as a mechanical impedance. The impedance control has the advantage of providing a uniform control structure for the non-contact, contact transition and contact phase of the task. The experimental results in [Kazerooni 90] shows that a stable transition form free motion to constrained motion is successfully achieved.

The third mainstream approach is to exploit a control scheme that consists of a position controller in the precontact phase and a force controller during the contact phase. Different techniques are used to achieve continuous switches between these two controllers. The approach proposed in [Volpe 93] is based on an impact controller with negative feedback gains. This method can maintain stability during contact transition. The model-based adaptive control technique is used to reduce the impact force by creating a contact transition task during the contact transition [Akella 94]. The work in [Tarn 96] represents a control method using positive acceleration feedback to control transient force response to reduce the peak of the impulsive force. The smooth switch strategy in [Wu 96] between position controller and force controller is based on the minimum measured contact force.

Three mainstream approaches are reviewed and discussed above. They use different techniques to solve the problem of the noncontact-to-contact transition, the goal of reducing large changes in control inputs is reached during task mode transitions.

2.2.3 Task Transitions

In order to handle multiple prioritized tasks, a large amount of hierarchical control frameworks are presented in the robotic literature to solve the control problem, generating joint torques to realize task executions. Analytical methods based on null-space projections can ensure that the higher priority task is executed employing all capabilities of the robotic system and the lower priority tasks are then performed in the null-space of the higher priority tasks [Khatib 87, Nakamura 87]. In other words, the task of the second level is executed as well as possible without interfering with the first level. The task of the third level is then executed without disturbing the two higher priority tasks, and so forth. The idea is based on the use of the limited Jacobian of a low priority task, which is projected into the null-space of the higher priority tasks by the application of a null-space projector [Liégeois 77]. Nowadays these techniques are applied in kinematic control [Baerlocher 04, Mistry 08] and in joint torque based control [Sentis 04, Sentis 10, Dietrich 12b, Dietrich 15]. Projected inverse dynamics schemes are developed for constrained systems in [Aghili 05, Khatib 08], where the dynamics equations is projected into the null-space of the Jacobian of constraint equations. The work in [Flacco 14] provides a reverse priority approach to handle multiple prioritized tasks. This approach, called Reverse Priority (RP), is based on the idea of computing the solution starting from the lowest priority one, and adding iteratively the contributions of higher priority task.

Recently many approaches based on Quadratic Programming (QP) are proposed to deal with multiple prioritized tasks. A basic quadratic programming problem to handle one task can be formulated by optimizing a quadratic objective function at time t, which is associated with the task, subject to linear equality and inequality constraints [Zhang 04a, Decré 09]:

$$\boldsymbol{\tau}_t^* = \begin{cases} \arg\min\limits_{\mathbb{X}_t} & \|T(\mathbb{X}_t, t)\|_Q^2 + \|\mathbb{X}_t\|_R^2 \\ \text{subject to} & \mathbf{A}(t)\mathbb{X}_t = \boldsymbol{b}(t) \\ & \mathbf{G}(t)\mathbb{X}_t \leq \boldsymbol{h}(t) \end{cases} . \tag{2.12}$$

The notation $\|\boldsymbol{a}\|_Q^2$ is the shorthand of the form $\boldsymbol{a}^T Q \boldsymbol{a}$, which shows that the objective function is quadratic. The weights Q and R govern the importance between the task objective and the regulation term. As the regulation term may increase task error, R is usually very small compared with Q. In (2.12), the dynamic motion equation (2.1) constitutes an equality constraint (2.8) to ensure physical realism. By solving this quadratic programming problem (2.12) at time t, joint torques $\boldsymbol{\tau}_t^*$ can be obtained.

In order to manage multiple prioritized tasks simultaneously, hierarchical quadratic programming (HQP) is developed to obtain the solution by solving a sequence of QPs in cascade at each time step. The computing process of HQP is to solve the first QP to get the solution for the highest priority task T_1, then to solve the second QP for the second highest priority task T_2 without increasing the obtained minimum of the task objective of T_1, until the lowest priority task T_{n_t} is resolved. This process is shown in Figure 2.5. This prioritization process corresponds to solving lower priority tasks in the null-space of higher priority tasks while trying to satisfy lower priority tasks at best. The computation is time consuming because HQP needs to solve n_t QPs while n_t tasks are performed at each time step. Specific work on the solver linear decomposition is proposed in [Escande 14] to avoid repetitive computations, reducing the computation cost of HQP.

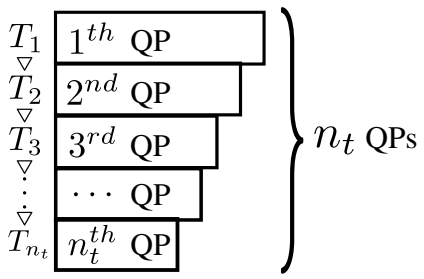


Figure 2.5: Hierarchical quadratic programming.

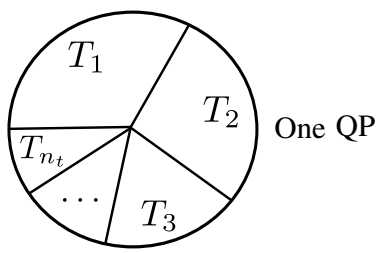


Figure 2.6: Weighting strategy.

In [Abe 07, Collette 07, Salini 11, Liu 12], a weighting strategy based on QP is proposed to handle multiple tasks with different priorities. Within this approach, each task $T_i(\mathbb{X}_t, t)$ is given a relative importance by assigning a weight $Q_i(t)$ and the solution is resolved by only one QP at each time step, the quadratic objective function of which is the sum of the weighted task objectives in (2.13). As shown in Figure 2.6, the weighting strategy provides a trade-off among task objectives with different importance levels. Therefore, the performances of higher priority tasks may not be guaranteed by simply adjusting the weights of task objectives. This means that a lower priority task may affect the performances of higher priority tasks.

$$\boldsymbol{\tau}_t^* = \begin{cases} \arg\min\limits_{\mathbb{X}_t} & \sum\limits_{i=1}^{n_t} (\|(T_i(\mathbb{X}_t, t)\|^2_{Q_i(t)}) + \|\mathbb{X}_t\|^2_R \\ \text{subject to} & \mathbf{A}(t)\mathbb{X}_t = \boldsymbol{b}(t) \\ & \mathbf{G}(t)\mathbb{X}_t \leq \boldsymbol{h}(t) \end{cases} . \tag{2.13}$$

In many contexts, task transitions are needed to cope with changing situations. However, task transitions may cause large instantaneous changes in joint torques due to the inversion of null-space projectors by using analytical methods, the sudden rearrangement of the computation sequence with HQP, or changes of task weights using weighting strategy. Recently, several approaches are developed to cope with task transitions. The work in [Lee 12] proposes a strategy called intermediate desired value approach to achieve continuous task transitions. Despite the effectiveness of this method, the computation cost increases exponentially with respect to the number of tasks in transitions. Another solution to achieve continuous priority rearrangements between only two levels of tasks is proposed using continuous null-space projector in [Petrič 13, Dietrich 12a]. Weighting strategy realizes continuous task transitions by continuously modifying the weight $Q_i(t)$ over time [Keith 11, Salini 11]. But weighting strategies still have their limitations that the lower priority tasks can affect the performance of the higher priority tasks. Therefore, a novel control framework is still needed to achieve continuous task transitions among arbitrary prioritized tasks.

2.3 Constraints

When performing tasks, the robot has to satisfy constraints, such as joint limits, actuator capacities, obstacle avoidance and contact constraints. These constraints are generally expressed in the form of equality (2.8) and inequality (2.9). All of them have to be taken into account when solving the control problem.

The analytical method by using the null-space projection matrices [Liégeois 77] can easily integrate equality constraints as task objectives with highest priorities to solve joint torques. In contrast, analytical method do not allow to explicitly take into account inequality constraints. A common method is to transform inequality constraint into a potential function [Khatib 86] as task objective, which is projected into the null-space of real task objectives. The potential function can generate repulsive forces to prevent the robot from entering into a forbidden region and to push it away from the constraint limits when the robot is coming closer [Chaumette 01, Stasse 08]. This solution can be applied not only to avoid collision with joint limits and saturation of actuator capacities, but also to avoid obstacles. However, in this case, inequality constraints may not be fully satisfied. To ensure that inequality constraints can be satisfied whatever the situation is, several solutions have been proposed to specify inequality constraint as higher priority tasks. In [Chan 95], the joint limit avoidance is used as a damping function to stop the motion of a joint, which is close to the limit. In the cases where damping is not sufficient, clamping is proposed in [Raunhardt 07]. However, it is difficult to relax a DoF that was clamped. The approaches presented in [Mansard 09] integrates unilateral constraints at any priority level. This approach has achieved interesting results, but the computation of some specific inverse operator is complex and time consuming. In general, the analytical method to handle inequality constraints can avoid the discontinuities cased by the switches between activation and inactivation of inequality constraints. The potential function works a buffer where the equality constraints are gradually activated. However, this analytical method is not capable to regulate multiple inequality constraints simultaneously, because it is complex to formulate a potential function for each inequality constraint and arrange their priorities.

As shown in (2.12), optimal methods based on QP can easily consider multiple equal-

ity and inequality constraints when solving the control problem. A QP is generally composed of two terms: the objective function and the constraints. It intrinsically has hierarchy between these two terms. The constraints always take priority over the objectives. This means that the constraints can be satisfied at any situation. Among inequality constraints (2.10d), joint limits and joint velocity limits are expressed based on a linear approximation with a time step δt in order to constrain the dynamic variable (optimal variable in QP). However, the linear approximation leads to a truncation error when solving the QP. When the robot reaches joint limits or joint velocity limits, chattering on joint torques occurs. In [Rubrecht 10] a constraint compliant control law is proposed to this problem, but it is only applied in the case of kinematic control. The approach proposed in [Park 98, Salini 12a] increases the time step δt in the constraint (not in the control time step) properly to prevent chattering for dynamic control problem. In fact, δt works as an anticipation coefficient to compute the estimated values of the future state $(\boldsymbol{q}, \dot{\boldsymbol{q}})$ in order to react to the limits in advance.

The bounds on joint limits and actuator capacities generally constant because they are determined by the robots' kinematics and the actuator's properties. In contrast, obstacle avoidance constraints and contact constraints are usually changing in the dynamic environment. The obstacle avoidance constraints may be updated within the control problem, *e.g.* when the distance to an obstacle is updated based on some sensor information. Contact constraints can be added to or removed from the control problem, *e.g.* when a contact is established or broken. The controller reactively adapt to the addition, removal or change of a constraint, which can result in large instantaneous changes in joint torques. To solve this problem, the changes of constraints are also needed to be taken into account in the robot control.

2.4 Conclusion

In this chapter, the current state-of-the-art studies involved with different types of causes of discontinuous control inputs have been described. Trajectory planning can efficiently produce a smooth trajectory to track for the robot and guarantee the continuity of control

actions, when task targets change due to changing requirement. Moreover, many approaches successfully handle the discontinuities due to task mode transitions. However, several key issues remain unsolved in the study of cases of discontinuous control inputs. Firstly, a framework to continuously change the priority of arbitrary tasks is still missing. Secondly, the reactive controller may not deal with suddenly changing constraints, especially obstacle avoidance constraints and contact constraints. Solving these two problems is a necessary condition to prevent large instantaneous changes in joint torques. With the aim of handling continuous task transitions and changing constraints, a generic control framework is presented in this thesis. In the following chapter, Generalized Hierarchical Control is proposed to achieve continuous task transitions among arbitrary prioritized tasks.

Chapter 3
Generalized Hierarchical Control

In this chapter, in order to solve the problem due to task transitions, a generalized projector is developed to regulate to what extent a lower-priority task is projected into the null-space of a higher-priority task. Based on the generalized projector, a generic dynamic hierarchical control framework is proposed to solve a single quadratic program (QP) to account for continuous task transitions among a set of prioritized tasks. Section 3.1 reviews recent techniques to solve the problem problem caused by task transitions. Section 3.2 proposes the GHC framework with detailed explanations of the generalized projector. Some experimental results are presented to demonstrate the framework capabilities, and comparisons with results using the HQP approach in Section 3.3. The continuity aspects of this approach are discussed in Section 3.4. Finally, Section 3.5 concludes the chapter by summarizing the contributions. This work was published in [Liu 16] with a major contribution of Mme. LIU.

3.1 Introduction

The redundancy of robots makes it possible to perform multiple tasks simultaneously, meanwhile all of these tasks may not be fulfilled at the same time due to task conflicts. Several existing approaches take advantage of task priorities to regulate multiple tasks. The classical prioritized approach based on the null-space projection ensure that lower priority tasks are executed in the null-space of higher priority tasks and do not affect the performances of higher priority tasks [Khatib 87, Sentis 04, Khatib 08]. Hierarchical quadratic programming [Saab 13, Escande 14] and weighting strategy [Abe 07, Liu 12] are used to regulate a set of tasks according to their priorities using quadratic programming (QP). However, they cannot directly be used to deal with task transitions because large instantaneous changes in joint torques may occur. Recently, three main kinds of methods have been developed to deal with task transition problem.

First of all, the approach proposed in [Lee 12] is called intermediate desired value. The general idea of this approach relies on the modification of the desired end-effector target of each task which is to be inserted, removed or switched. By continuously changing the desired target during the transition, the continuity of the control inputs is ensured. Despite the effectiveness of this approach, the computation cost increase exponentially with respect to the number of tasks in transitions.

Secondly, the approaches presented in [Keith 11, Salini 11, Jarquin 13] are based on the weighing strategy. Continuous task transitions are achieved by continuously modifying the weight assigned to the task objective. The insertion of a task is realized by gradually increasing its weight to bring the task from the last priority to the desired one. In contrast, the inverse process is executed to remove a task. The obtained continuity of the control inputs are determined by the duration of the transition period, which is a trade-off between reactiveness and smoothness. Besides, this method cannot guarantee the strict task hierarchies.

Thirdly, an analytical approach to handle task transitions is developed in [Mansard 09]. A specific inverse operator is used to guarantee the continuous task transitions by smooth activation/deactivation process of tasks. In [Petrič 13, Dietrich 12a], continuous null-space projector is proposed to implement continuous task transitions. The null-space projectors of two prioritized tasks continuously change to smooth rearrangement of priorities by a pair of coupled coefficients in [Petrič 13]. In the approach presented in [Dietrich 12a], an activator associated to directions in the right singular vectors of a task Jacobian matrix is regulated to activate or deactivate these directions. However, the design of such an activator makes this approach difficult to be implemented for the separate handling of different task directions.

Although the techniques reviewed above can solve large instantaneous changes in joint torques induced by task transitions, they have their respective limitations and cannot be implemented in a generic way. With the aim of achieving continuous task transitions among an arbitrary number of tasks, a novel QP reactive controller, called Generalized Hierarchical Control (GHC), is proposed in this chapter (see Figure 3.1). This new controller can successfully maintain strict and non-strict hierarchies of multiple tasks by

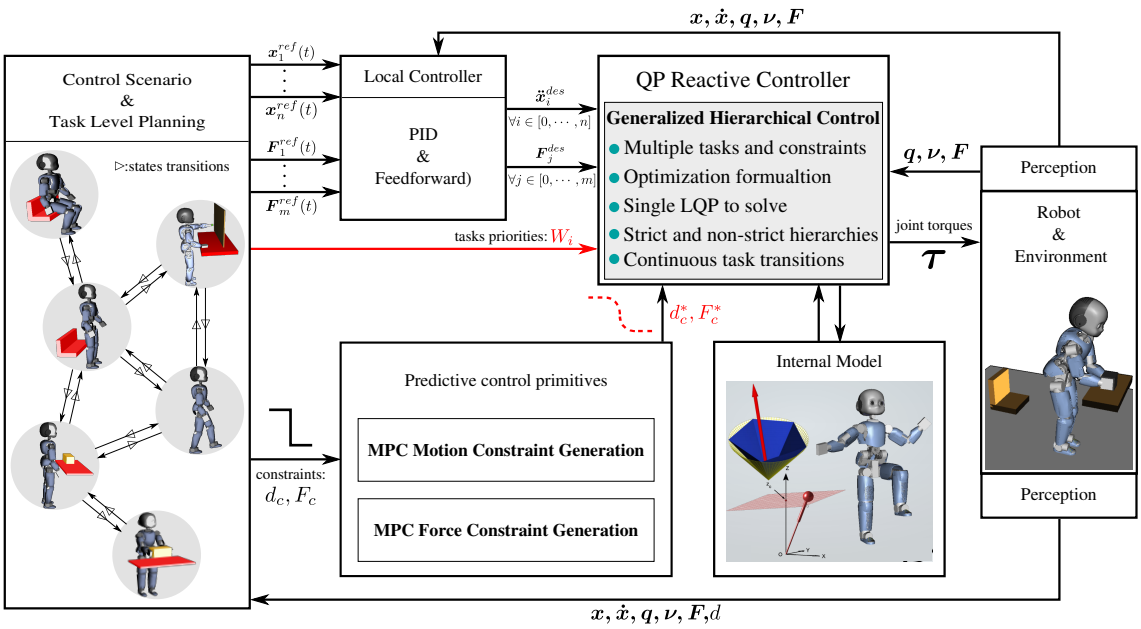


Figure 3.1: Overview of the methodology developed for achieving continuous tasks and constraints transitions. The features of Generalized Hierarchical Control are highlighted.

solving only one QP problem while satisfying various constraints simultaneously.

3.2 Generalized hierarchical control

The hierarchical control proposed in this Chapter is based on a new generalized projector, which can both precisely regulate how much a task is affected by other tasks, and continuously achieve arbitrary task transitions among a set of prioritized tasks. The following part of this section first reviews several forms of projectors, the analysis of which leads to the development of the generalized projector.

3.2.1 Basic Null-space Projector

Task priorities can be handled by analytical methods using a null-space projector $\mathbf{N}_j = \mathbf{I} - \mathbf{J}_j^{\dagger}\mathbf{J}_j$, where $\mathbf{J}_j^{\dagger}$ is the Moore-Penrose pseudo-inverse of the Jacobian $\mathbf{J}_j$[1]. The projection of a task i into the null-space of another task j can ensure that the lower-priority task i is

[1]The dependence to q is omitted for clarity reasons.

performed without producing any motion for the higher-priority task j. To handle priorities between one task i and a set of other tasks with higher priorities, task i is projected into the null-space of an augmented Jacobian composed of all the higher priority tasks [Siciliano 91, Baerlocher 98].

To achieve smooth priority transitions, the null-space projector is replaced by the following matrix in [Keith 11, Petrič 13]

$$\mathbf{N}_j^{'}(\alpha_{ij}) = \mathbf{I} - \alpha_{ij}\mathbf{J}_j^{\dagger}\mathbf{J}_j \,, \tag{3.1}$$

where a scalar parameter $\alpha_{ij} \in [0,1]$ is used to regulate the priority between two tasks i and j. The greater α_{ij} is, the more task i is projected into the null-space of task j. This method can handle priority transitions between only two levels of tasks, and it can hardly be extended to the case of simultaneous transitions among multiple priority levels.

The following projection matrix $\mathbf{N}^{''}$ is proposed in [Dietrich 12a] for continuous null-space projections

$$\mathbf{N}^{''} = \mathbf{I} - \mathbf{V}\mathbf{\Theta}\mathbf{V}^T \,, \tag{3.2}$$

where $\mathbf{V} \in \mathbb{R}^{n\times n}$ is composed of the right-singular vectors of $\mathbf{J}_j$, the Jacobian of a higher priority task, and $\mathbf{\Theta} \in \mathbb{R}^{n\times n}$ is a diagonal activation matrix. The k-th diagonal element of $\mathbf{\Theta}$, $\theta_{kk} \in [0,1]$, refers to the k-th column vector in $\mathbf{V}$: when $\theta_{kk} = 1$, the k-th direction in $\mathbf{V}$ is activated in $\mathbf{N}^{''}$; when $0 < \theta_{kk} < 1$, the k-th direction in $\mathbf{V}$ is partially deactivated; when $\theta_{kk} = 0$, the k-th direction in $\mathbf{V}$ is deactivated. As mentioned in [Dietrich 12a], for any one-dimensional task j ($\mathbf{J}_j \in \mathbb{R}^{1\times n}$), the matrix (3.2) becomes

$$\mathbf{N}_j^{''} = \mathbf{I} - \theta_{11}\frac{\mathbf{J}_j^T}{\|\mathbf{J}_j\|}\frac{\mathbf{J}_j}{\|\mathbf{J}_j\|} \,, \tag{3.3}$$

where only the first element θ_{11} of $\mathbf{\Theta}$ is relevant. $\mathbf{N}_j^{''}$ can be applied to achieve activation or deactivation of task j direction in the projection matrix by the variation of the scalar θ_{11}. When extended to a task (or a set of tasks) of m directions ($\mathbf{J}_j \in \mathbb{R}^{m\times n}$), this method allows one to apply the same transition to all the m directions of $\mathbf{J}_j$, but its application for achieving the separate regulation of each task direction is not easy. This is because each activator θ_{kk} is directly referred to the k-th direction in the right singular vectors of $\mathbf{J}_j$, but

not directly referred to a specific direction in $\mathbf{J}_j$.

3.2.2 Generalized Projector

In order to achieve continuous transitions of multiple task priorities among an arbitrary number of tasks, an approach to the computation of a generalized projector is developed in this section. First of all, a set of relative importance levels with respect to n_t tasks, including task i, characterized by a priority matrix

$$\mathbf{W}_i = \begin{bmatrix} \alpha_{i1}\mathbf{I}_{m_1} & 0 & 0 & 0 & 0 \\ 0 & \ddots & 0 & 0 & 0 \\ 0 & 0 & \alpha_{ij}\mathbf{I}_{m_j} & 0 & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & \alpha_{in_t}\mathbf{I}_{m_{n_t}} \end{bmatrix} \tag{3.4}$$

where $\mathbf{W}_i$ is a diagonal matrix, the main diagonal blocks of which are square matrices: $\alpha_{ij}\mathbf{I}_{m_j}$. m_j is the dimension of the task j. $\mathbf{I}_{m_j}$ is the $m_j \times m_j$ identity matrix, and $\alpha_{ij} \in [0, 1]$. By convention, the coefficient α_{ij} indicates the priority of task j with respect to task i.

- $\alpha_{ij} = 0$ corresponds to the case where task j has strict lower priority with respect to task i.
- $0 < \alpha_{ij} < 1$ corresponds to a compromise between the two tasks: task j is not restricted in the null-space of task i. The greater the value of α_{ij}, the higher the importance level of task j with respect to task i.
- $\alpha_{ij} = 1$ corresponds to the case where task j has a strict higher priority with respect to task i.

Similarly to the form of matrix $\mathbf{N}''$ in (3.2) in the case of considering a one-dimensional task (3.3), the form of $\mathbf{P}_i(\mathbf{W}_i) \in \mathbb{R}^{n\times n}$ is obtained without the necessity of the computation of pseudo-inverse matrices. Here the subscript $_i$ in $\mathbf{P}_i$ indicates that the projector takes into account the priorities of a set of tasks with respect to task i. The dependence of $\mathbf{P}_i$

to $\mathbf{W}_i$ is sometimes omitted hereafter for clarity reasons. Moreover, the new projector allows one to regulate the activation of each task directions in a more intuitive way, by regulating the priority matrix $\mathbf{W}_i$ that is more closely related to task directions than the activator $\mathbf{\Theta}$ in (3.2).

First, look at the following matrix, which extends $\mathbf{N}_j^{''}$ defined by (3.3) from the handling of one task direction to the handling of the directions of n_t tasks

$$\mathbf{N}^{'''} = \mathbf{I} - \sum_{j=1}^{n_t} \alpha_{ij} \frac{\mathbf{J}_j^T}{\|\mathbf{J}_j\|} \frac{\mathbf{J}_j}{\|\mathbf{J}_j\|} , \tag{3.5}$$

where, without loss of generality, each task dimension is supposed to be 1, and α_{ij} with $j = 1, 2, 3, \ldots, n_t$ parameterizes the priority of each of the n_t tasks with respect to a certain task i. For any task k among the n_t tasks with $\alpha_{ik} = 1$, which means that task k is of the highest priority, the product of $\mathbf{N}^{'''}$ with $\mathbf{J}_k$ leads to

$$\begin{aligned} \mathbf{J}_k\mathbf{N}^{'''} &= \mathbf{J}_k - \mathbf{J}_k \frac{\mathbf{J}_j^T}{\|\mathbf{J}_j\|} \frac{\mathbf{J}_j}{\|\mathbf{J}_j\|} - \sum_{j \neq k} \mathbf{J}_k \alpha_{ij} \frac{\mathbf{J}_j^T}{\|\mathbf{J}_j\|} \frac{\mathbf{J}_j}{\|\mathbf{J}_j\|} \\ &= - \sum_{j \neq k} \mathbf{J}_k \alpha_{ij} \frac{\mathbf{J}_j^T}{\|\mathbf{J}_j\|} \frac{\mathbf{J}_j}{\|\mathbf{J}_j\|} \end{aligned} . \tag{3.6}$$

In (3.6), $\mathbf{J}_k\mathbf{N}^{'''} = 0$ if $\mathbf{J}_k\mathbf{J}_j^T = 0$ for each $j \neq k$ and $\alpha_{ij} > 0$. This means that the highest priority of task k may not be satisfied if it is interfered by a lower priority task j, which has a component along task k direction. On the contrary, task priorities can be maintained if such task interferences disappear, or in other words, if all the lower priority task directions become orthogonal to all the higher priority task directions. Based on this observation, the computation of the generalized projector $\mathbf{P}_i$ is divided into three steps.

Step one is a preliminary processing of the matrices $\mathbf{J}$ and $\mathbf{W}_i$, where

$$\mathbf{J} = \left[\mathbf{J}_1^T \ldots \mathbf{J}_j^T \ldots \mathbf{J}_{n_t}^T \right]^T \tag{3.7}$$

is the augmented Jacobian concatenating the Jacobian matrices of all the n_t tasks. The processing of $\mathbf{J}$ and $\mathbf{W}_i$ is carried out according to the priorities of the n_t tasks with respect to task i. As each row of $\mathbf{J}$ is associated to α_{ij}, the rows of $\mathbf{J}$ can be sorted in descending order with respect to the values of the diagonal elements in $\mathbf{W}_i$. The resulting matrix $\mathbf{J}_{s_i}$ is

thus constructed so that tasks which should be the least influenced by task i appear in its first rows, while tasks which can be the most influenced by task i appear in its last rows. The values in $\mathbf{W}_i$ are sorted accordingly, leading to $\mathbf{W}_i^s$, the diagonal elements of which are organized in descending order starting from the first row.

Step two consists in the computation of a matrix $\mathbf{B}_i(\mathbf{J}_{s_i}) \in \mathbb{R}^{r \times n}$ by using $\mathbf{J}_{s_i}$, where r is the rank of $\mathbf{J}_{s_i}$. The rows of $\mathbf{B}_i(\mathbf{J}_{s_i})$ form an orthonormal basis of the joint space obtained using elementary row transformations on $\mathbf{J}_{s_i}$. Algorithm (1) describes this computation using Gram-Schmidt process [Cheney 09, p. 544]. As in any numerical scheme, tolerances are used here for numerical comparison, such as ϵ in line #11 of Algorithm (1), which is defined as a small positive value. As the use of ϵ may lead to rank jumps in $\mathbf{B}_i$, it is suggested to assign the smallest value greater than zero to ϵ to avoid large variation of $\mathbf{B}_i$.

Algorithm 1: Orthonormal basis computation - $GetOrthBasis(\mathbf{J})$

```
   Data: J, ϵ
   Result: B, origin, r
1  begin
2      n ⟵ GetNbCol(J)
3      m ⟵ GetNbRow(J)
4      i ⟵ 0
5      for k ← 0 to m − 1 do
6          if i ≥ n then
7              break
8          B[i, :] ⟵ J[k, :]
9          for j ← 0 to i − 1 do
10             B[i, :] ⟵ B[i, :] − (B[i, :]B[j, :]^T) B[j, :]
11         if norm(B[i, :]) > ϵ then
12             B[i, :] ⟵ B[i, :]/norm(B[i, :])
13             origin[i] ⟵ k
14             i ⟵ i + 1
15     r ⟵ i
16     return B, origin, r
17 end
```

Step three is to compute the generalized projector, which is given by

$$\mathbf{P}_i(\mathbf{W}_i) = \mathbf{I}_n - \mathbf{B}_i(\mathbf{J}_{s_i})^T \mathbf{W}_i^{r,s}(\mathbf{W}_i, \boldsymbol{origin}) \mathbf{B}_i(\mathbf{J}_{s_i}) \ , \tag{3.8}$$

where $\mathbf{W}_i^{r,s}$ is a diagonal matrix of degree r. The vector $\boldsymbol{origin} \in \mathbb{R}^r$ is a vector of the row

indexes of $\mathbf{J}_{s_i}$ selected during the construction of the orthonormal basis $\mathbf{B}_i$. Each of these r rows in $\mathbf{J}_{s_i}$ is linearly independent to all the previously selected ones. The diagonal elements of $\mathbf{W}_i^{r,s}$ are restricted to the r diagonal elements of $\mathbf{W}_i^s$, which correspond to the r rows of $\mathbf{J}_{s_i}$, the row indexes of which belong to ***origin***. Algorithm (2) summarizes the construction of the generalized projector.

Algorithm 2: Generalized projector computation - task i

Data: $\mathbf{W}_i, \mathbf{J}$
Result: $\mathbf{P}_i$
begin
  $n \longleftarrow GetNbCol(\mathbf{J})$
  $\boldsymbol{index} \longleftarrow GetRowsIndexDescOrder(\mathbf{W}_i)$
  $\mathbf{W}_i^s \longleftarrow SortRows(\mathbf{W}_i, \boldsymbol{index})$
  $\mathbf{J}_{s_i} \longleftarrow SortRows(\mathbf{J}, \boldsymbol{index})$
  $\mathbf{B}_i, \boldsymbol{origin}, r \longleftarrow GetOrthBasis(\mathbf{J}_{s_i})$ ▷Alg. (1)
  $\mathbf{W}_i^{r,s} \longleftarrow GetSubDiagMatrix(\mathbf{W}_i^s, \boldsymbol{origin})$
  $\mathbf{P}_i \longleftarrow \mathbf{I}_n - \mathbf{B}_i^T \mathbf{W}_i^{r,s} \mathbf{B}_i$
  return $\mathbf{P}_i$
end

Note that the interference of lower priority tasks with higher priority tasks, which exists in (3.5) if two task directions of different priorities are not orthogonal ($\mathbf{J}_k \mathbf{J}_j^T \neq 0$), is avoided in $\mathbf{P}_i$. Indeed, each row in $\mathbf{B}_i$ corresponds to the component of a task direction that is effectively accounted for by the projector $\mathbf{P}_i$. The row sorting in step one ensures that higher priority task directions are accounted for in $\mathbf{B}_i$ prior to any lower priority task direction, and the orthonormalization process in step two ensures that each direction (or row) of $\mathbf{B}_i$ is orthogonal to previous rows associated to all the higher priority task directions.

By varying the value of each α_{ij} in $\mathbf{W}_i$, one can regulate the priority of each task j with respect to task i separately. Indeed, during the execution of task i, the projector $\mathbf{P}_i$ can be configured such that

- for tasks having strict priority over task i, the movement along their task directions is completely forbidden by setting corresponding $\alpha_{i\bullet}$ to 1;
- for tasks over which task i has a strict priority, the movement along their directions is completely allowed by setting corresponding $\alpha_{i\bullet}$ to 0;

- and for tasks with non strict priorities, the movement along their task directions is partially allowed according to the value of their priority parameters. The increase of the values of corresponding $\alpha_{i\bullet} \in (0, 1)$ leads to the increase of the priorities of the associated tasks with respect to task i, and thus stronger restriction of task i movements along their task directions.

There is a particular case induced by the proposed formulation and corresponding to the influence of task i on itself. Even though not intuitive, this self-influence has to be interpreted in terms of task existence, modulated by α_{ii}. If $\alpha_{ii} = 1$ then task i is projected into its own null-space, *i.e.* it is basically cancelled out. Indeed, α_{ij} evolves over the entire execution timeline to regulate the priority of each task j with respect to task i. The priority matrix $\mathbf{W}_i$ and the projector $\mathbf{P}_i$ are formulated based on α_{ij}. Therefore, they are all time-variant and can be rewritten as: $\alpha_{ij}(t)$, $\mathbf{W}_i(t)$ and $\mathbf{P}_{i,t}$. During the execution of tasks, the continuous task priority transitions are achieved through continuously modifying the value of $\alpha_{ij}(t)$ over time. For example, decreasing $\alpha_{ii}(t)$ continuously to 0 activates task i gradually. Conversely, increasing $\alpha_{ii}(t)$ continuously from 0 to 1 deactivates the task gradually.

3.2.3 Generalized Hierarchical Control Framework

The control framework based on weighting strategy (2.13) can qualitatively regulate the relative priorities of tasks by weighting task objectives, but it cannot ensure strict task priorities. The GHC framework proposed here extends the framework (2.13) through the implementation of generalized projectors defined by (3.8) to handle continuous transitions among prioritized tasks.

Consider the control problem for solving n_t tasks. The operating principle of GHC is summarized by the following LQP problem, which takes into account the desired task priorities parameterized by the priority matrix $\mathbf{W}_i(t)$.

$$\boldsymbol{\tau}_t^* = \begin{cases} \underset{\dot{\boldsymbol{v}}'_t, \boldsymbol{\tau}_t, \boldsymbol{F}_{c,t}}{\arg\min} & \sum_{i=1}^{n_t} (\|(T_i(\dot{\boldsymbol{v}}'_{i,t}, \boldsymbol{\tau}_t, \boldsymbol{F}_{c,t}, t)\|^2) + \left\| \begin{bmatrix} \dot{\boldsymbol{v}}'_t \\ \boldsymbol{\tau}_t \\ \boldsymbol{F}_{c,t} \end{bmatrix} \right\|_R^2 & (3.9a) \\ \text{subject to} & \mathbf{A}(t) \begin{pmatrix} \dot{\boldsymbol{v}}_t \\ \boldsymbol{\tau}_t \\ \boldsymbol{F}_{c,t} \end{pmatrix} = \boldsymbol{b}(t) & (3.9b) \\ & \mathbf{G}(t) \begin{pmatrix} \dot{\boldsymbol{v}}_t \\ \boldsymbol{\tau}_t \\ \boldsymbol{F}_{c,t} \end{pmatrix} \leq \boldsymbol{h}(t) & (3.9c) \\ & \dot{\boldsymbol{v}}_t = \mathbf{P}_t \dot{\boldsymbol{v}}'_t = \sum_{i=1}^{n_t} \mathbf{P}_{i,t}(\mathbf{W}_i(t)) \dot{\boldsymbol{v}}'_{i,t} & (3.9d) \end{cases}$$

with $\dot{\boldsymbol{v}}'_t = \left[\dot{\boldsymbol{v}}'^{\,T}_{1,t} \ldots \dot{\boldsymbol{v}}'^{\,T}_{n_t,t}\right]^T$ and $\mathbf{P}_t = [\mathbf{P}_{1,t}(\mathbf{W}_1(t)) \ldots \mathbf{P}_{n_t,t}(\mathbf{W}_{n_t}(t))]$. Each $\dot{\boldsymbol{v}}'_{i,t}$ in (3.9) is an intermediate system acceleration variable associated to each task i and $\dot{\boldsymbol{v}}_t$ is the overall system accelerations accounting for the sets of desired task priorities $(\mathbf{W}_1(t), \ldots, \mathbf{W}_{n_t}(t))$. Assuming a perfect model, $\dot{\boldsymbol{v}}_t$ is the system accelerations resulting from the application of the joint torques computed by solving (3.9).

This optimization problem minimizes the objective function of each task as well as the magnitude of the control input, subject to constraints. Each task objective function is expressed in terms of the intermediate system acceleration variable $\dot{\boldsymbol{v}}'_{i,t}$. Note that in GHC, task priorities are handled by using the generalized projectors $\mathbf{P}_{i,t}$ in (3.9d) instead of task weights. Therefore, here the task weighting matrix Q_i in (2.13) is set to the identity matrix, which is omitted in (3.9a).

A solution to the equation of motion in (2.8) can be ensured as long as there exists a highest priority task i such that $\mathbf{P}_{i,t}(\mathbf{W}_i(t)) = \mathbf{I}_n$ ($\mathbf{W}_i(t)$ being the zero matrix), which means that this task is not projected in the null-space of any other task. Indeed, the equation of motion in (2.8) can be expressed in terms of intermediate joint accelerations as

$$\mathbf{S}\boldsymbol{\tau}_t + \mathbf{J}_c(\boldsymbol{q}_t)^T \boldsymbol{F}_{c,t} = \mathbf{M}(\boldsymbol{q}_t)\mathbf{P}_t \dot{\boldsymbol{\nu}}'_t + \boldsymbol{n}(\boldsymbol{q}_t, \boldsymbol{\nu}_t) \ , \tag{3.10}$$

with $\mathbf{P}_t = [\mathbf{P}_{1,t}(\mathbf{W}_1(t)) \dots \mathbf{P}_{n_t,t}(\mathbf{W}_{n_t}(t))]$. As the inertia matrix $\mathbf{M}$ is positive definite, a solution to (3.10), and thus (3.9b), can be ensured if $\mathbf{P}_t$ has full row rank. A sufficient condition to ensure this property of $\mathbf{P}_t$ is that there exists at least one $\mathbf{P}_{i,t}$ which equals the identity matrix, and this is the case for the highest priority task in a hierarchy.

Since the constraints have a higher priority than the objectives in LQP, and in (3.9) the constraints are expressed in terms of the overall system accelerations $\dot{\boldsymbol{\nu}}$, it is ensured that the solution accounting for desired task hierarchies satisfies the constraints. Or in other words, the GHC framework ensures that the constraints, such as the equation of motion in equality constraints (3.9b) and the other physical limitations in inequality constraints (3.9c), have a higher priority over task hierarchies. Moreover, this GHC framework can handle strict task hierarchies represented by standard lexicographic orders. The proof is provided in Appendix A.

Another property of GHC is that it is robust to both kinematic and algorithmic singularities. In this framework based on LQP, tasks are expressed in a forward way and most LQP solvers do not require the explicit inversion of Jacobian matrices. Therefore, GHC does not have problems of numerical singularities due to kinematic singularities. Moreover, unlike approaches using the pseudo-inverse of limited Jacobians ($\mathbf{J}_i\mathbf{N}_j$), which requires special treatment for handling algorithmic singularities when the limited Jacobians drop rank [Sadeghian 13], GHC does not require the inversion of priority consistent Jacobians. Therefore, the framework does not have to handle such kind of algorithmic singularities.

3.3 Results

The proposed GHC framework (3.9) is applied to the control of a 7-DoF KUKA LWR robot. The experiments are conducted in the Arboris-Python simulator [Salini 12b], which is a rigid multibody dynamics and contacts simulator written in Python. The LQP problem is solved by a QP solver included in CasADi-Python [Andersson 12], which is a

symbolic framework for dynamic optimization.

In the experiments, three tasks are defined: task 1 for the control of the three dimensional position of the end-effector, task 2 for the control of the three dimensional position of the elbow, and task 3 for the control of the 7-DoF posture. The elbow task target is a static target position and the posture task target is a static posture. For each task i, an optimization variable $\ddot{\boldsymbol{q}}_i' \in \mathbb{R}^7$ is defined. A local proportional-derivative controller is used to ensure the convergence of each task variable towards its target $\ddot{\boldsymbol{x}}^d = k_p(\boldsymbol{x}^{goal} - \boldsymbol{x}) + k_d(\dot{\boldsymbol{x}}^{goal} - \dot{\boldsymbol{x}})$. When a task target is static, $k_p = 30s^{-2}$ and $k_d = 20s^{-1}$. When tracking a desired trajectory, $k_p = 100s^{-2}$ and $k_d = 20s^{-1}$. The priority parameter matrices associated with the three tasks are: $\mathbf{W}_i(t) = diag(\alpha_{i1}(t)\mathbf{I}_3,\ \alpha_{i2}(t)\mathbf{I}_3,\ \alpha_{i3}(t)\mathbf{I}_7)$ with $i = 1, 2, 3$. The regularization weight R is chosen as 0.01. The following function is used for the smooth variation of $\alpha_{ij}(t)$ (conversely $\alpha_{ji}(t)$) from 0 to 1 during the transition time period ($[t_1, t_2]$)

$$\begin{aligned}
\alpha_{ij}(t) &= 0.5 - 0.5\cos\left(\frac{t - t_1}{t_2 - t_1}\pi\right), \quad t \in [t_1, t_2], \\
\alpha_{ji}(t) &= 1 - \alpha_{ij}(t).
\end{aligned}$$

3.3.1 Insertion and Removal of Tasks

In this experiment, the end-effector task is inserted into the control framework with the highest priority while the elbow task and the posture task are operating. The end-effector task is removed after a while. The evolution of the task hierarchy is thus defined as: $2 \triangleright 3 \Rightarrow 1 \triangleright 2 \triangleright 3 \Rightarrow 2 \triangleright 3$.

The process of the insertion and the removal of the end-effector task is performed by the continuous change of α_{11}. Moreover, as the end-effector task is inserted with the highest priority, the values of α_{12} and α_{31} increase smoothly from 0 to 1 during the insertion period. Figure 3.2 illustrates the variation of the αs and the task errors during this operation. The end-effector task error decreases to zero when it becomes the highest priority task, and no abrupt changes of task errors are observed. This result illustrate the fact that a task can be inserted into a set of tasks with a desired priority level, or be removed from

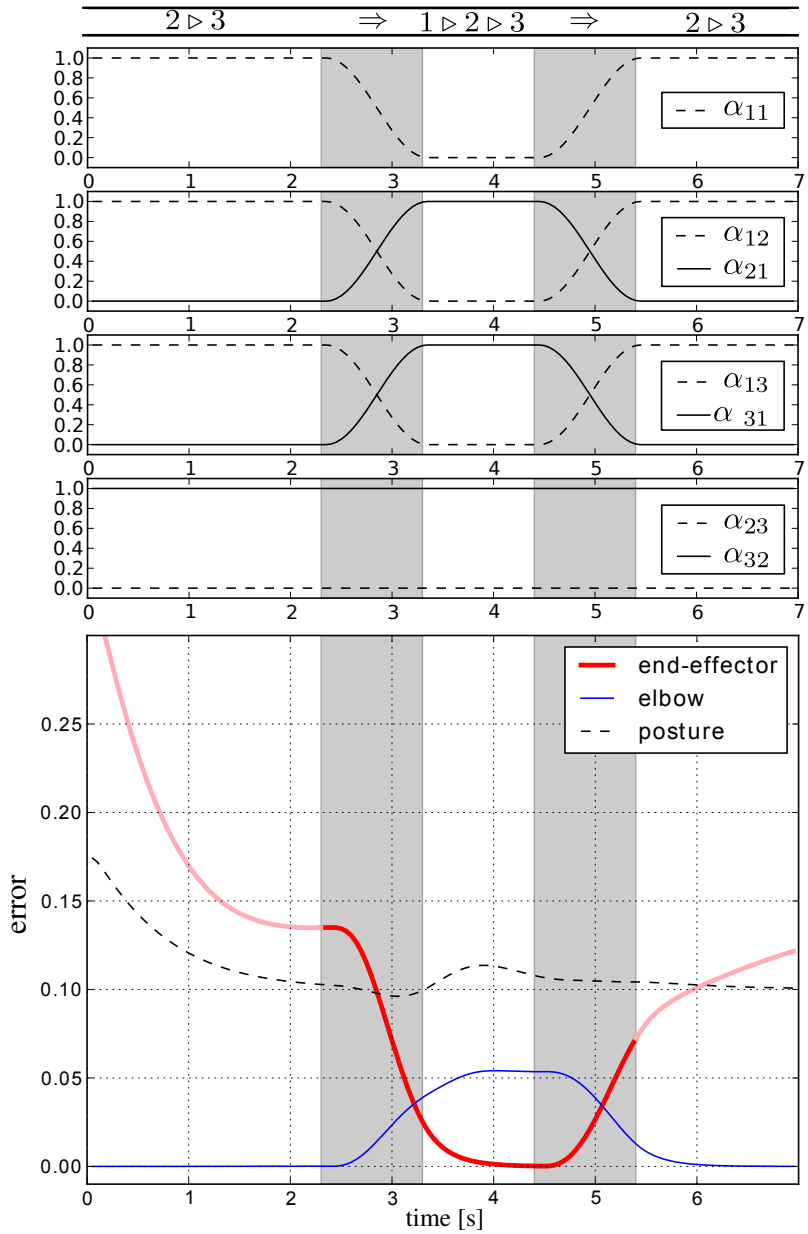


Figure 3.2: Evolution of αs (top) and task errors (bottom) during the insertion and the removal of the end-effector task. The end-effector task (task 1) is firstly inserted to be the task with the highest priority, then removed from the set of tasks. The parameter α_{11} is regulated for the insertion and removal of the end-effector task. The other parameters $\alpha_{12}, \alpha_{21}, \alpha_{13}$, and α_{31} are regulated for handling the priorities between the end-effector task and the other two tasks. α_{23} and α_{32} are constant to maintain the priority of task 2 over the task 3. The insertion and removal of task 1 occurs in a continuous manner and the prescribed priorities are well realized.

the set of tasks, continuously and smoothly.

3.3.2 Task Transitions Subject to Constraints

This experiment is carried out to demonstrate that GHC allows handling task priorities subject to a variety of constraints. The end-effector task is to move along a lemniscate-shaped trajectory, periodically, with an orbital period of 2π seconds. The elbow task and posture task is static. In addition, the joint velocity and torque bounds are imposed.

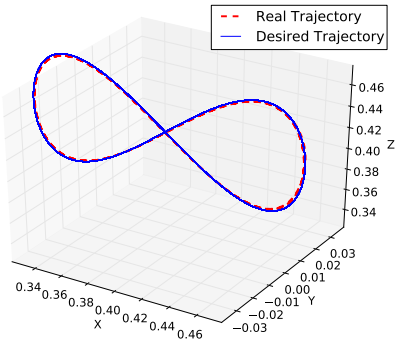


Figure 3.3: The desired and the resulting end-effector trajectory provided by GHC, when the end-effector task has the highest priority. The end-effector moves along the lemniscate-shaped trajectory with an orbital period of 2π seconds.

The evolution of task hierarchy is $3 \triangleright 2 \triangleright 1 \Rightarrow 1 \triangleright 2 \triangleright 3 \Rightarrow 1 \triangleright 3 \Rightarrow 1 \triangleright 2 \triangleright 3$. In the beginning, the tasks, in the priority level decreasing order, are the posture task, the elbow task, and the end-effector task. Then the end-effector task priority increases and the posture task priority decreases simultaneously. Afterwards, the priorities of the end-effector task and the elbow task are switched. Then the elbow task is removed. Finally, the elbow task is inserted with its priority level between those of the end-effector task and the posture task. The desired and the resulting end-effector trajectory is shown in Figure 3.3. The resulting task errors using GHC is presented in Figure 3.4. The resulting joint velocities and joint torques are shown in Figure 3.5.

Figure 3.4 shows that when the end-effector task has the highest priority, it can track its desired trajectory precisely. Moreover, Figure 3.5 shows that joint velocity and joint

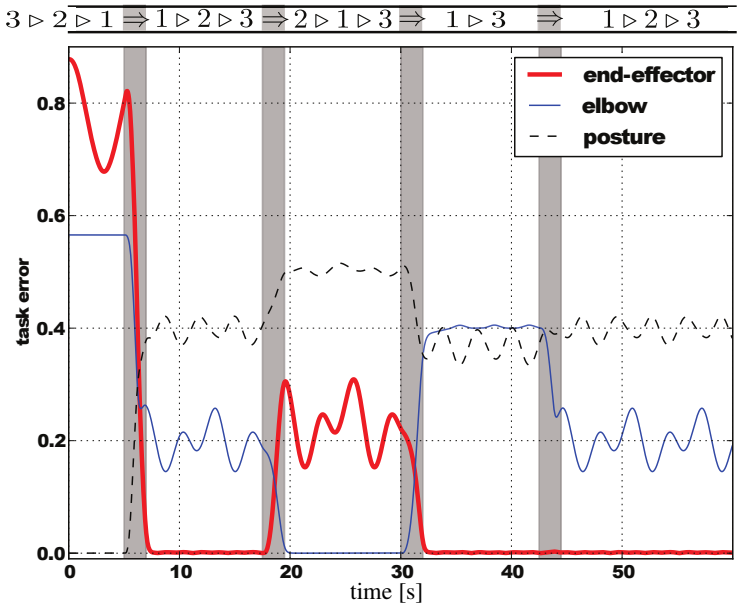


Figure 3.4: Task errors using GHC, with the end-effector tracking a lemniscate-shaped trajectory. Desired priority transitions as well as the insertion and removal of the elbow task are achieved. Strict task hierarchies are maintained.

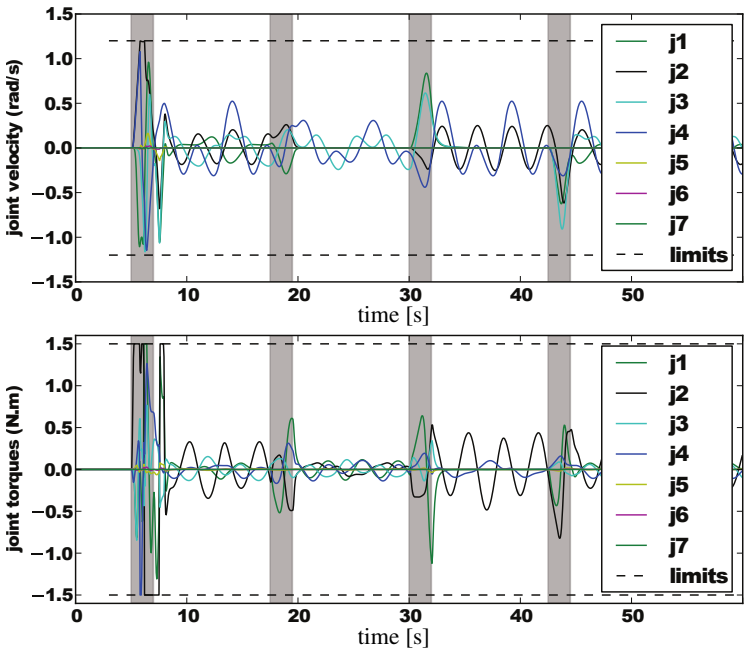


Figure 3.5: Evolution of joint velocities and joint torques. The upper and lower bounds of $\dot{\boldsymbol{q}}$ are 1.2 rad/s and -1.2 rad/s, respectively. The upper and lower bounds of $\boldsymbol{\tau}$ are 1.5 $N \cdot m$ and -1.5 $N \cdot m$, respectively. These bounds are voluntarily set low in order to easily illustrate the fact that they are satisfied.

torque limits are satisfied, which demonstrate that GHC can maintain desired task hierarchies and achieve continuous task transitions while satisfying constraints.

3.3.3 Comparisons with HQP

In this experiment, the changes of the task hierarchy are the same as the experiment in Section 3.3.2. The process of task transitions is shown in Figure 3.6. The experiment is carried out using static task targets for steady state error analysis. The performance of GHC is compared with the HQP approach [Kanoun 11].

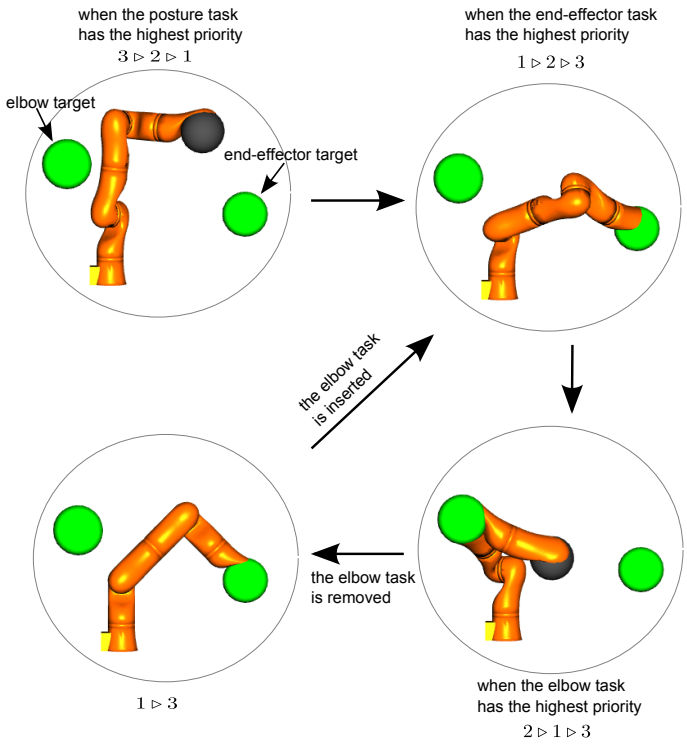


Figure 3.6: Experiment of priority switching.

The results corresponding to the use of static task targets are presented in Figure 3.7 to 3.10. Task errors by using HQP (Figure 3.7) as well as those by using GHC with different hierarchy rearrangement durations (Figure 3.8 and Figure 3.9) are shown. The hierarchy rearrangement duration is 0.005 seconds in Figure 3.8 and 2 seconds in Figure 3.9. Figure 3.10 shows the integration of the absolute values of the resulting joint jerks $\sum_{i=1}^{n}\left(\int_0^t \frac{|\mathrm{d}^3 q_i|}{\mathrm{d}t^3} dt\right)$ by using HQP that performs instantaneous hierarchy rearrangements,

as well as by using GHC with faster and slower hierarchy rearrangements. Steady state task errors for each task hierarchy configuration are shown in Table 3.1, where the results using GHC and HQP are included.

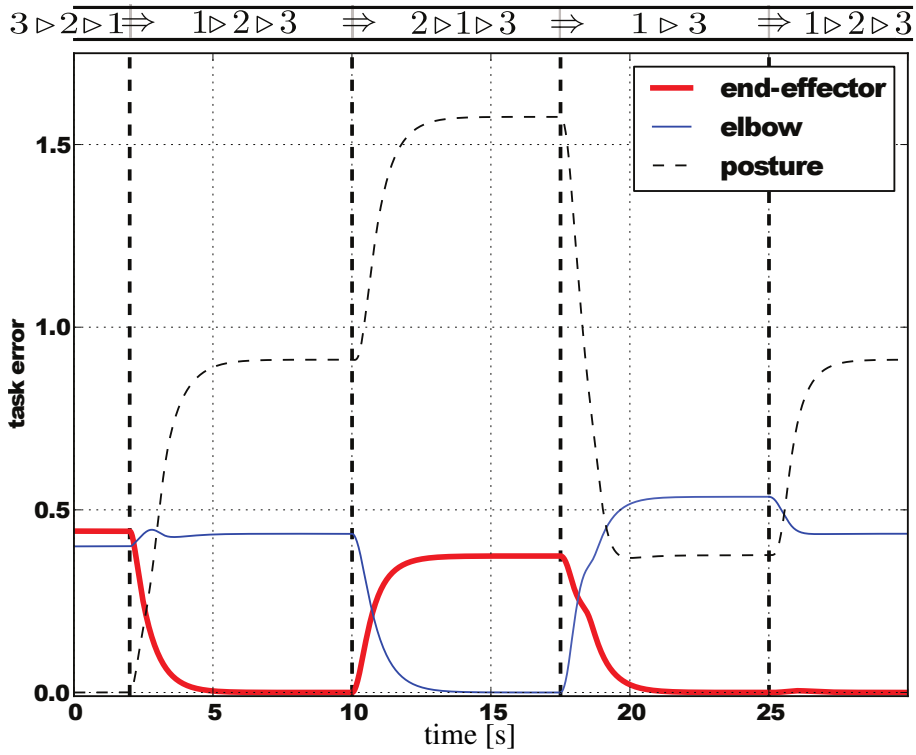


Figure 3.7: Task errors using HQP, with fixed task targets. Priority transitions as well as the insertion and removal of the elbow task are performed. The hierarchy rearrangement is instantaneous.

GHC provides similar results in terms of task errors compared with HQP, as can be observed in Figure 3.8 and 3.7. The results of task errors in Table 3.1 show that both GHC and HQP can ensure strict task hierarchies. When controlled by either GHC or HQP, errors of the tasks with the highest priority are very small. Moreover, GHC can perform slower and smoother hierarchy rearrangements that require less joint jerks. This can be seen in Figure 3.10, which shows that GHC can generate smaller joint jerks than HQP does.

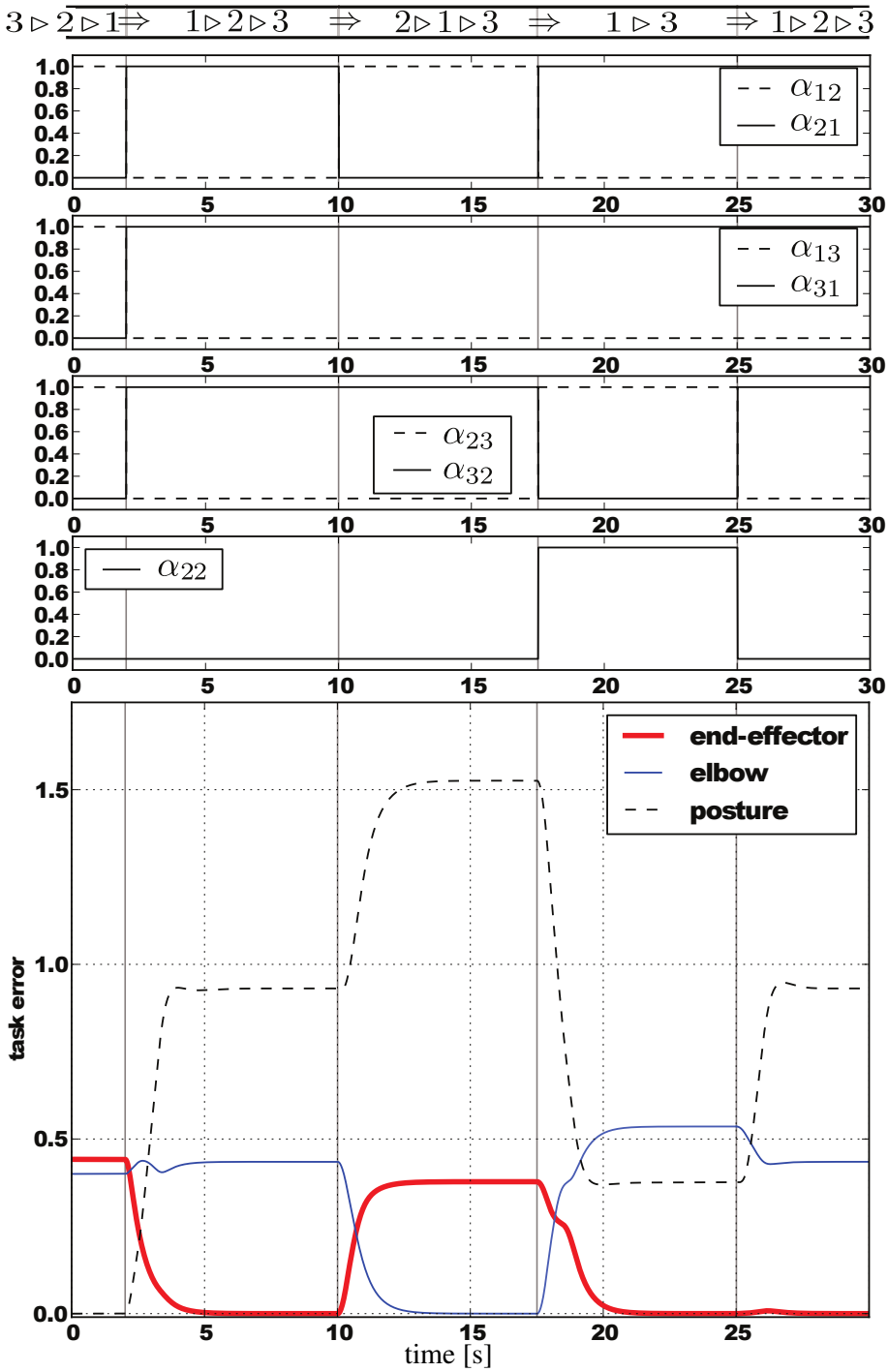


Figure 3.8: Evolution of αs (top) and task errors (bottom) using GHC, with fixed task targets. Priority transitions as well as the insertion and removal of the elbow task are performed. The hierarchy rearrangement duration is 0.005 seconds.

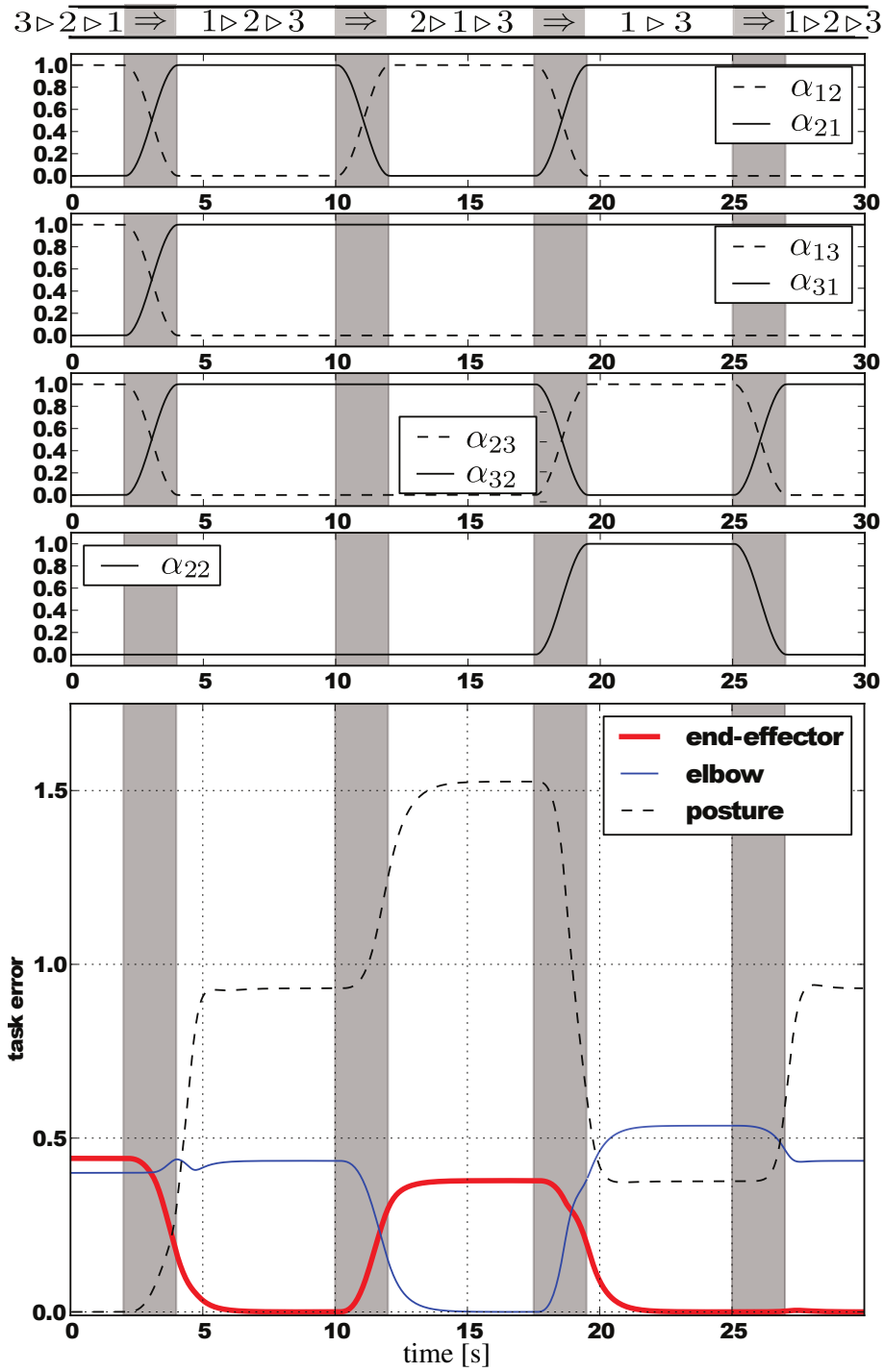


Figure 3.9: Evolution of αs (top) and task errors (bottom) using GHC, with fixed task targets. Priority transitions as well as the insertion and removal of the elbow task are performed. The hierarchy rearrangement duration is 2 seconds.

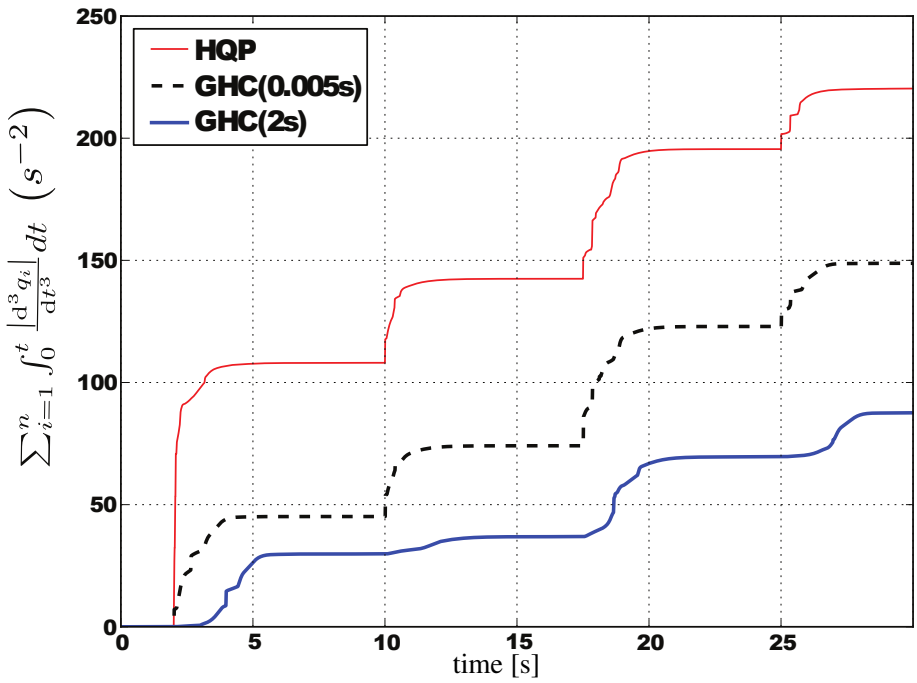


Figure 3.10: Integration of the absolute values of joint jerks using GHC and HQP, with fixed task targets. The value is increased each time the task hierarchy is changed. GHC generates less amount of joint jerks by performing slower hierarchy rearrangements; while HQP, which performs instantaneous hierarchy rearrangements generates larger joint jerks.

Table 3.1: Steady state task errors for each task hierarchy configuration

priority	$3 \triangleright 2 \triangleright 1$			$1 \triangleright 2 \triangleright 3$		
task	1	2	3	1	2	3
GHC	0.46	0.40	2.2e-30	1.0e-6	0.46	1.8
HQP	0.46	0.40	2.8e-10	4.5e-7	0.46	1.8
priority	$2 \triangleright 1 \triangleright 3$			$1 \triangleright 3$		
task	1	2	3	1	2	3
GHC	0.42	2.6e-6	3.0	3.9e-6	0.55	0.79
HQP	0.42	2.7e-6	3.1	4.5e-6	0.55	0.79

3.4 Discussion

It can be seen in Figure 3.10 that GHC generates smaller joint jerks than HQP does, which implies that GHC provides smoother priority transitions. Basically, the solution of GHC is continuous, even during hierarchy rearrangement, if the vector ***origin*** in (3.8) remains the same before and after the rearrangements. Indeed, in this case, the basis $\mathbf{B}_i$ used to compute the generalized projector varies continuously with $\mathbf{J}_i$, and the generalize projector varies continuously with $\mathbf{B}_i$ and α_i. However, similarly to the HQP algorithm, discontinuity may still occur during the switch of priorities or during the insertion and removal of tasks, In GHC, such a discontinuity is due to the change of the basis $\mathbf{B}_i$ during hierarchy rearrangement.

3.5 Conclusion

This chapter proposes a generalized hierarchical control approach for handling continuous task transitions. A generalized projector is developed. It can precisely regulate how much a task can influence or be influenced by other tasks through the modulation of a priority matrix: a task can be completely, partially, or not at all projected into the null-space of other tasks. Multiple simultaneous changes of task priorities can be achieved by using this generalized projector and, using the same mechanism, tasks can be easily inserted or removed. Moreover, the GHC approach can maintain and switch task priorities while satisfying a set of equality and inequality constraints.

In this chapter, the GHC approach is illustrated at the dynamic level; however, the generalized projector introduced here is not restricted to this case. In fact, it can also be used in other types of controllers, such as a velocity kinematics controller or a quasi-static controller. The idea is to associate each task with an intermediate task variable in joint space ($\dot{\boldsymbol{q}}_i'$, $\ddot{\boldsymbol{q}}_i'$, $\boldsymbol{\tau}_i'$, etc.), then to apply generalized projectors to these task variables, and finally the global joint space variable is the sum of each projected task variables($\mathbf{P}_i(\mathbf{W}_i)\dot{\boldsymbol{q}}_i'$, $\mathbf{P}_i(\mathbf{W}_i)\ddot{\boldsymbol{q}}_i'$, $\mathbf{P}_i(\mathbf{W}_i)\boldsymbol{\tau}_i'$, etc.).

One question remains unanswered: how to choose the right amount of time for transitions? In non dynamic and non time critical situations, this time can be chosen arbi-

trarily long. However, in many cases, a compromise has to be found between the required reactivity of the robot (short transitions) and the continuity of control inputs (long transitions). This compromise cannot be obtained by reasoning on the instantaneous behaviour of the robot. Indeed, choosing the right compromise requires to preview the consequences of the control action over a time horizon. For example, when a robot has to recover balance using an extra contact with one of its hands, there may be no time for smooth transitions if we want the robot to avoid falling, but this cannot be known based on the current state of the robot. This advocates for predictive approaches.

While this work of optimally choosing the transition time using predictive approaches has not been carried out in this thesis, the interest of such approaches is exploited in the following chapters. In the next chapter, a predictive control primitive is indeed proposed to deal with moving obstacles. Model Predictive Control is employed to generate continuous obstacle avoidance constraints for the QP reactive controller, effectively minimizing the rate of change in joint torques.

Chapter 4
Motion Constraint Generation

In this chapter, a predictive primitive is proposed to minimize large instantaneous changes in joint torques. This predictive primitive can produce continuous obstacle avoidance constraint for a QP reactive controller using Model Predictive Control (MPC). The MPC can generate continuous position, velocity and acceleration profiles of obstacles based on the sensed sudden movements of obstacles. Furthermore, a continuous obstacle avoidance constraint can be obtained. Section 4.1 reviews techniques to deal with obstacle avoidance for robots. Section 4.2 introduces a new obstacle avoidance constraint including position, velocity and acceleration profiles of the obstacle as well as the reaction time and safe distance, which enable the robot to have time to react to the sudden movement of the obstacle. Section 4.3 reviews fundamental principles of Model Predictive Control. Section 4.4 proposes a MPC scheme to generate continuous obstacle avoidance constraint by taking advantage of the position information of the obstacle from distance sensors and the reaction time to prevent large instantaneous changes in joint torques. A sensitivity analysis is carried out to find the effects of the parameters of the MPC. To illustrate the effectiveness of the proposed approach, Section 4.5 shows the simulation results on the iCub robot. Section 4.6 concludes the chapter by summarizing the contributions.

4.1 Introduction

Robot collision avoidance is traditionally treated as a planning problem, and research in this area has focused on the development of collision-free path planning algorithm [Gouzenes 84, Lumelsky 87, Hsu 02, González-Banos 06]. These algorithms provide a path that will enable the robot to accomplish its assigned task free from any risk of collisions. The advantage of the planning approach is that it can obtain a global solution. However, the robot is limited to the execution of elementary operations for which the paths have been precisely specified. Additionally, global trajectory planning is not only a time-consuming technique, but also incapable of dealing with trajectory replanning due

to changing situations. Therefore, using planning algorithms to avoid collisions is not sufficient if robots are operating in an unstructured and dynamic environment.

A complementary way to achieve obstacle avoidance is to integrate environment sensing feedback to the reactive controller . The advantage of reactive obstacle avoidance is to compute control actions by introducing the sensor information within the control loop. The artificial potential filed (APF) approach in [Khatib 86] is a well-established approach to deal with obstacle avoidance. The created virtual field around the obstacle induces repulsive forces applied on the robot when it gets too close to the obstacle. These potential fields are built around the obstacle and allow real-time reactive control. The Gradient Projection Method (GPM) [Liégeois 77] has been widely used in the literature by utilizing the null-space projector to deal with inequality constraints. Based on this framework, a low-level controller is proposed with two components in [Maciejewski 85, Brock 02]: the principal goal is the desired task and the secondary is the obstacle avoidance projected into the null-space of the task. Herein, obstacle avoidance is realized in the null-space of the task objective. It is given a higher priority if that null-space is not sufficient to ensure collision-free motion. The same idea is applied in [Stasse 08, Sugiura 10, Dietrich 12b] to handle self-collision avoidance in a task hierarchy using the null-space projector. However, obstacle avoidance is treated as tasks in these analytical methods. Once the controller has to cope with other inequality constraints, the analytical methods becomes more complex and is not easy to be solved.

Quadratic Programming provides a mathematical framework particularly suitable to the formulation of the controller while accounting for tasks and constraints simultaneously. It allows to explicitly write the control problem as objectives to be reached under a certain set of constraints. In [Faverjon 87, Zhang 04b], a Quadratic Programming control structure is proposed to handle obstacle avoidances. In this controller, obstacle avoidances are treated as inequality constraints, which limit the robot's velocities towards obstacle. The work in [Salini 12a] integrates the position, velocity and acceleration of the robot to the obstacle avoidance constraint. The constraint with dynamic parameters of the system is created to keep the minimum distance between the robot and the obstacle positive.

All these solutions are adapted to local procedures and cannot be applied in more complex context, for example when robots have to reach some goals in highly cluttered spaces. Additionally, these approaches neglect the important case where the obstacle is moving. Even though this motion is continuous, its perception may no be. Indeed, if the sensor refresh rate is much smaller (*e.g.* $10Hz$) than the control sampling rate (*e.g.* $1KHz$), the perceived motion of the obstacle will be discontinuous from the control point of view. The limitation in the range of detection of the sensor may also lead to similar perceived discontinuities when obstacles "appear" or "disappear" from the field of view of the robot. As a consequence, the obstacle avoidance constraint stated above will be discontinuous and result in large instantaneous changes in joint torques when solving the control problem at time t_{k+1}. Moreover, in this scenario the collision is inevitable if the obstacle avoidance constraint is formulated as the minimum distance between the robot and the obstacle to be kept positive. In practice, obstacles are commonly uncontrolled and their movements are random. The robot has to adapt to their movements reactively. Nevertheless, the movements of obstacles can be measured by various distance sensors, and these information can be integrated to the controller, enabling the robot to have time to react to their sudden movements.

In this chapter, to deal with the abrupt movements of obstacles, a control framework including a predictive primitive and a QP reactive controller is proposed to minimize the rate of changes in joint torques (see in Figure 4.1). The new obstacle avoidance is formulated as the minimum distance between the robot and the obstacle based on the work in [Salini 12a]. It not only allows for considering the position, velocity and acceleration of both the robot and the obstacle, but also provides a safe distance for the robot to react to movements of the obstacle. Given the minimal time for the robot to react to these abrupt movements of obstacles, analytical methods, like polynomial, can be used to produce continuous position, velocity and acceleration of the obstacle based on the measured position information of the obstacle. However, polynomial approaches require manual choose of the start and end points of the segment to be smoothed and manual tuning of polynomial parameters. Moreover, they cannot handle complex constraint profiles on the movements of obstacles in an automatic and generic way. Therefore, Model

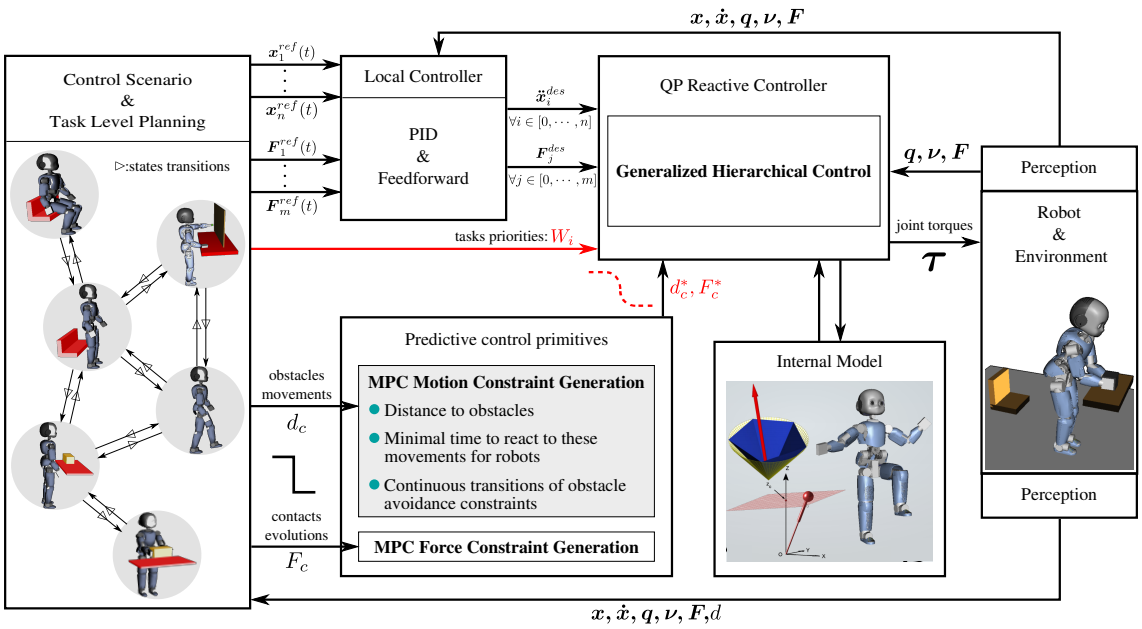


Figure 4.1: The control framework including a predictive primitive and a QP reactive controller is proposed to minimize large changes in joint torques due to abrupt movements of obstacles.

Predictive Control is employed automatically to produce continuous position, velocity and acceleration of the obstacle according to the measured position information of obstacles and the reaction time. The generated continuous position, velocity and acceleration of the obstacle are not real ones, but they can be used to yield continuous obstacle avoidance constraint the QP reactive controller. Therefore, large instantaneous changes in joint torques can be minimized despite abrupt movements of obstacles.

4.2 Obstacle avoidance constraint

As shown in Figure 4.2, the obstacle avoidance constraint can be defined as requiring the minimum distance $d_k = \|\boldsymbol{p}_{r,\,k} - \boldsymbol{p}_{o,\,k}\|$ between the robot body and the obstacle to be strictly non-zero [Faverjon 87, Kanehiro 08], where $\boldsymbol{p}_{\bullet,\,k} = [x_{\bullet,\,k}\; y_{\bullet,\,k}\; z_{\bullet,\,k}]^T$ represents the position in Cartesian space at time k. Subscript $_r$ and $_o$ represent the robot and the obstacle, respectively. However, this assumption is insufficient in dynamic environments where obstacles are moving. In the example of the obstacle abruptly moving towards the

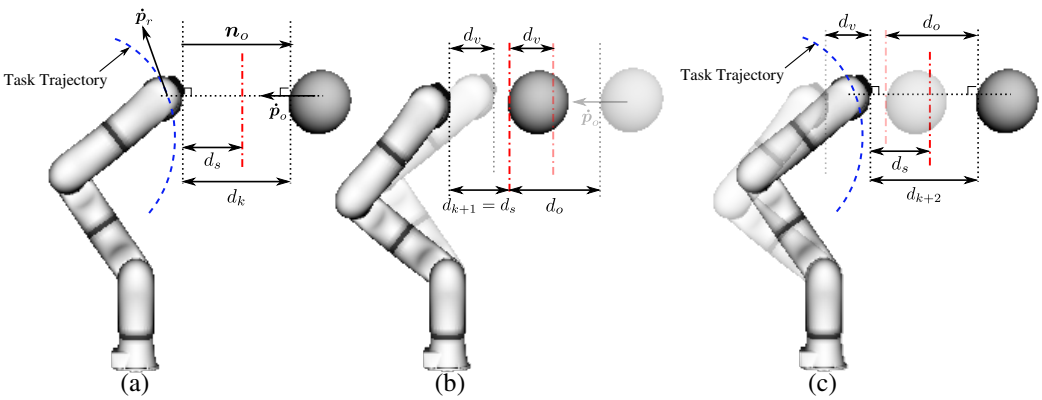


Figure 4.2: Description of the obstacle avoidance constraint.

robot at time $k+1$ under the condition that $d_{k+1} = 0$ was satisfied at time k, the collision will be inevitable. By considering moving obstacles, a safe distance between the robot and the obstacle d_s is proposed in this work to prevent any potential collision between the robot and moving obstacles. In this case, the obstacle avoidance constraint can be formulated as $d_k \geq d_s$. Using a discrete linear approximation with a time step of δt, the minimum distance d_{k+1} can be expressed as:

$$d_{k+1} = d_k + \boldsymbol{n}_o^T(\dot{\boldsymbol{p}}_{r,\,k} - \dot{\boldsymbol{p}}_{o,\,k})\delta t + \frac{\delta t^2}{2}\boldsymbol{n}_o^T(\ddot{\boldsymbol{p}}_{r,\,k} - \ddot{\boldsymbol{p}}_{o,\,k})\,, \tag{4.1}$$

where $\boldsymbol{n}_o$ is the vector associated to the shortest distance from the robot to the object. The vector $\boldsymbol{n}_o$ is assumed to evolve continuously in this thesis. $\boldsymbol{p}_{r,\,k}$ and $\dot{\boldsymbol{p}}_{r,\,k}$ are the position and velocity of the robot at time k, $\ddot{\boldsymbol{p}}_{r,\,k}$ is the acceleration resulting from the next control action. Using the robot kinematic relationship between the operational space velocity/acceleration and the generalized coordinates in (4.1), the obstacle avoidance constraint can be expressed linearly with respect to $\ddot{\boldsymbol{q}}_k$ as:

$$\begin{aligned} & d_k + \boldsymbol{n}_o^T(\mathbf{J}(\boldsymbol{q}_k)\dot{\boldsymbol{q}}_k - \dot{\boldsymbol{p}}_{o,\,k})\delta t + \frac{\delta t^2}{2}\boldsymbol{n}_o^T(J(\boldsymbol{q}_k)\ddot{\boldsymbol{q}}_k + \dot{J}(\boldsymbol{q}_k)\dot{\boldsymbol{q}}_k - \ddot{\boldsymbol{p}}_{o,\,k}) \geq d_s \\ \Leftrightarrow & -J(\boldsymbol{q}_k)\ddot{\boldsymbol{q}}_k \leq d_k - d_s + \boldsymbol{n}_o^T(J(\boldsymbol{q}_k)\dot{\boldsymbol{q}}_k - \dot{\boldsymbol{p}}_{o,\,k})\delta t + \frac{\delta t^2}{2}\boldsymbol{n}_o^T(\dot{J}(\boldsymbol{q}_k)\dot{\boldsymbol{q}}_k - \ddot{\boldsymbol{p}}_{o,\,k}) \end{aligned} . \tag{4.2}$$

According to (4.2), $h_i(k)$ can be expressed as:

$$h_i(k) = d_k - d_s + \boldsymbol{n}_o^T(\mathbf{J}(\boldsymbol{q}_k)\dot{\boldsymbol{q}}_k - \dot{\boldsymbol{p}}_{o,\,k})\delta t + \frac{\delta t^2}{2}\boldsymbol{n}_o^T(\dot{\mathbf{J}}(\boldsymbol{q}_k)\dot{\boldsymbol{q}}_k - \ddot{\boldsymbol{p}}_{o,\,k})\,. \tag{4.3}$$

$h_i(k)$ is related to the position $\boldsymbol{p}_{r,\,k}$ and velocity $\dot{\boldsymbol{p}}_{r,\,k}$ of both the robot. It also relies on the knowledge of $\boldsymbol{p}_{o,\,k}$, $\dot{\boldsymbol{p}}_{o,\,k}$ and $\ddot{\boldsymbol{p}}_{o,\,k}$, which are the position, velocity and acceleration of the obstacle at time k. Since the robot performs continuous motion tasks or continuous force tasks, its position $\boldsymbol{p}_{r,\,k}$ and velocity $\dot{\boldsymbol{p}}_{r,\,k}$ evolve continuously over time. In dynamic environment, obstacles are commonly uncontrollable and their movements are random. This means that the position, velocity and acceleration of the obstacle may change largely in a very shot time. If these instantaneous large movements of the obstacle occurs outside the safe distance d_s, $h_i(t)$ is inactive in the controller and it does not cause large instantaneous changes in joint torques. However, once the obstacle moves d_o towards the robot at time $k+1$ and violates the safe distance with the *violation distance* d_v (see in Figure 4.2(b)), the sudden changes of $\boldsymbol{p}_{o,\,k}$, $\dot{\boldsymbol{p}}_{o,\,k}$ and $\ddot{\boldsymbol{p}}_{o,\,k}$ result in discontinuous $h_i(t)$, which lead to switches of the constraint (4.2) from being inactive to being active instantaneously in the controller. As a result, large instantaneous changes in joint torques are generated and the robot moves d_v away from the obstacle instantaneously to maintain the safe distance correspondingly. It should be noted that the resulting changes in joint torques are closely related to the violation distance d_v. The larger the violation distance d_v is, the larger changes in joint torques are. But the violation distance d_v should be less than the safe distance d_s. If not, the robot cannot react to the movement of the obstacle and the collision occurs. In contrast, once the obstacle moves away from the robot abruptly as shown in Figure 4.2(c), the constraint (4.2) will switch from being active to being inactive. Similarly, the resulting large instantaneous changes in joint torques make the robot reach its task objective immediately.

The reactive adaptation to the sudden movement of the obstacle results in large instantaneous changes in joint torques through the discontinuous obstacle avoidance constraint. Generally, the instantaneous movements of the obstacle occur in a short time, particularly within a control time step. It is not easy to measure velocity and acceleration of the obstacle, but its final position can be easily measured by sensors. When the obstacle approaches the robot, the violation distance d_v can be obtained as illustrated in Figure 4.3. Since the violation distance d_v is within the safe distance d_s, this enables the robot to react to the sudden movement of the obstacle within a *reaction time* t_r instead of within

only one control step δt, meanwhile preventing any collision after the obstacle moves. The minimum reaction time is determined by the violation distance and the maximum expected velocity of the robot in Cartesian space v_{max}:

$$min(t_r) = d_o / v_{max} \tag{4.4}$$

Oppositely, if the obstacle moves away from within the safe distance, the possibility of collisions is reduced. Therefore, here is no need for the controller to react to this movement instantaneously but continuously.

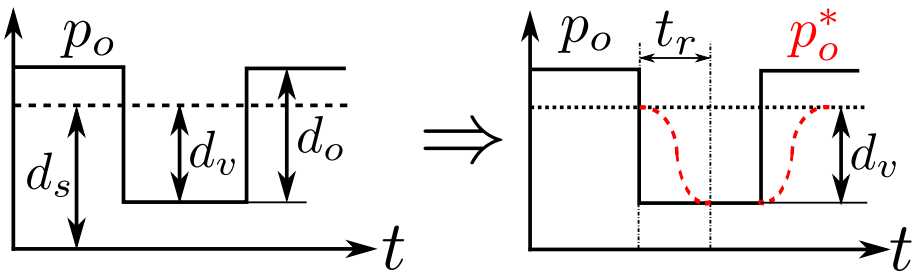


Figure 4.3: The conception of continuously reacting to sudden movements of obstacles.

Using distance sensors, the instance when the robot enters into or leaves away the safe region and the position of the obstacle can be known. In order to continuously react to the sudden movements of the obstacle, continuous evolution of $h_i(t)$ is required to be generated with the reaction time t_r and the violation distance d_v. The key is to generate virtual continuous position, velocity and acceleration of the obstacle for $h_i(t)$. One way to generate a continuous curve such as shown in Figure 4.3 is by a polynomial function, like cubic polynomial [Lin 83] and quintic polynomial [Andersson 89]. However, the use of polynomial approaches depends on the choices of the initial and final points of the segment to be smoothed and manual tuning of polynomial parameters. Polynomial approaches are difficult to account for the constraints applied on the position, velocity and acceleration of the obstacle when generating the trajectory. Moreover, as the order of polynomial increases, its oscillations increases. Among different methods, Model Predictive Control (MPC) stands out with many advantages to handle dynamical changes. MPC is an optimal control strategy based on numerical optimization. Given the reaction time t_r and the violation distance d_v, MPC can know the virtual final state of the obsta-

cle in advance and take advantage of the reaction time to generate continuous position, velocity and acceleration of the obstacle. Most importantly, MPC provides a systematic method of dealing with current and future constraints on inputs and states. Before the formulation of MPC for generating continuous trajectories of obstacle, Model Predictive Control is briefly reviewed in the following section.

4.3 Model Predictive Control

Model Predictive Control (MPC), also known as receding horizon control [Bellingham 03], usually contains the following three ideas [Camacho 13, p. 1]:

1) Explicit use of a model to predict the system output along a future time horizon;

2) Calculation of a control sequence by minimizing an objective function with respect to a reference trajectory;

3) A receding horizon strategy, so that at each instant the horizon is displaced towards the future, which involves the application of the first control signal of the sequence calculated at each step.

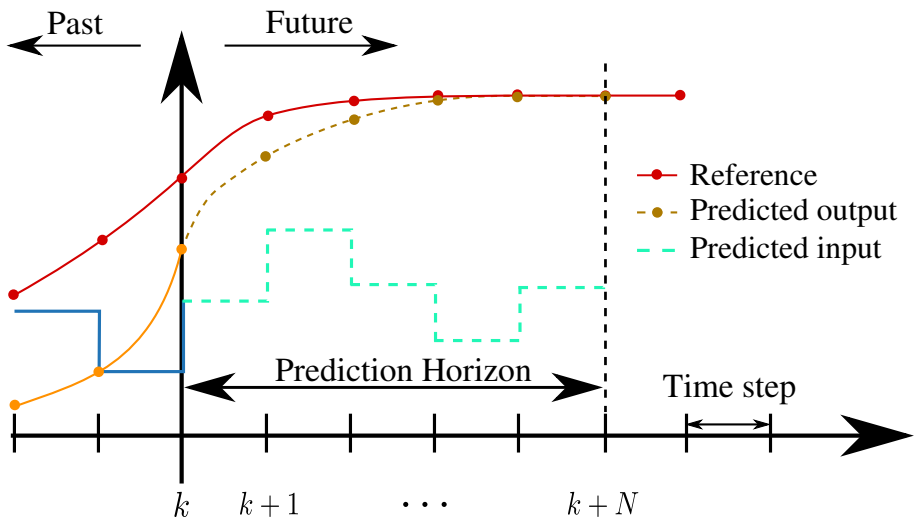


Figure 4.4: Concept of Model Predictive Control. At time step k, an optimal horizon of outputs is computed with respect to the output reference.

The calculation process of MPC is shown in Figure 4.4. At time step k, the current states are measured and a cost minimizing the error between the predicted output and the reference trajectory is computed over a time horizon $[k, k+N]$ to find a sequence of the system inputs. Only the first control in this sequence is sent to the system, then the calculation shifts to the next time step, yielding repeatedly a new set of system future inputs. The real power of MPC is that it can take into account the future constraints on both control inputs and system states.

A Model Predictive Control law contains the basic components: dynamic model, optimization and receding horizon implementation. This thesis is concerned mainly with the case of discrete time linear systems with state space representation:

$$\boldsymbol{x}(k+1) = \mathbf{A}\boldsymbol{x}(k) + \mathbf{B}\boldsymbol{u}(k) \ , \tag{4.5}$$

where $\boldsymbol{x}(k) \in \mathbb{R}^n$ and $\boldsymbol{u}(k) \in \mathbb{R}^m$ are the model state and the input vector at time k, respectively. Matrices $\mathbf{A}$ and $\mathbf{B}$ implicitly describe the linearity of the system. Given a predicted input sequence, the corresponding sequence of state predictions is generated by the model over a horizon of $N \in \mathbb{N}$ time steps. The prediction sequences of the state vector $\boldsymbol{x}$ and the input vector $\boldsymbol{u}$ are defined as follows:

$$\boldsymbol{X}_k = [\boldsymbol{x}(k+1|k) \ \ldots \ \boldsymbol{x}(k+N|k)]^T \ , \tag{4.6}$$

$$\boldsymbol{U}_k = [\boldsymbol{u}(k|k) \ \ldots \ \boldsymbol{u}(k+N-1|k)]^T \ . \tag{4.7}$$

Here, $\boldsymbol{x}(k+i|k)$ and $\boldsymbol{u}(k+i-1|k)$ denote the state and input vectors of the system at time $k+i$ that are predicted at time k, and $\boldsymbol{x}(k+i|k)$ therefore evolves according to the model:

$$\boldsymbol{x}(k+i|k) = \mathbf{A}\boldsymbol{x}(k+i-1|k) + \mathbf{B}\boldsymbol{u}(k+i-1|k) \ . \tag{4.8}$$

In practice, the system inputs and outputs are constrained, which are related to technological limitations of the system (saturation, limits, etc.). These constraints are frequently expressed in the form of linear inequality constraints on $\boldsymbol{U}_k$:

$$\mathbf{A}_c \boldsymbol{U}_k \leq \boldsymbol{b}_c \ . \tag{4.9}$$

Using quadratic programming, the predictive control problem can be formulated over a horizon of N time steps at time k.

$$\boldsymbol{U}_k^* = \begin{cases} \underset{\boldsymbol{U}_k}{\arg\min} & \sum_{i=1}^{N} [\boldsymbol{x}(k+i|k)^T \mathbf{Q} \boldsymbol{x}(k+i|k) + \boldsymbol{u}(k+i-1|k)^T \mathbf{R} \boldsymbol{u}(k+i-1|k)^T] \\ \text{subject to} & \mathbf{A}_c \boldsymbol{U}_k \leq \boldsymbol{b}_c \\ & \boldsymbol{x}(k+i|k) = \mathbf{A}\boldsymbol{x}(k+i-1|k) + \mathbf{B}\boldsymbol{u}(k+i-1|k) \end{cases}, \tag{4.10}$$

where $\mathbf{Q}$ and $\mathbf{R}$ are positive definite weighting matrices determining the weights between outputs and inputs of the system. The solution $\boldsymbol{U}_k^* = [\boldsymbol{u}(k|k)^* \ \ldots \ \boldsymbol{u}(k+N-1|k)^*]^T$ of MPC problem (4.10) can be numerically solved by the standard algorithms of quadratic optimization, and $\boldsymbol{u}(k|k)^*$ solely is applied to the system. Then the problem is updated and computed again at next time step $k+1$.

This MPC formulation in a quadratic form has been widely used in robot manipulators and humanoid robots. For example, many approaches based on model predictive control are presented to achieve trajectory tracking [Künhe 05, Makarov 11]. Stable walking patterns for humanoid robots are often generated using model predictive control approach [Kajita 03, Wieber 06]. In this chapter, Model Predictive Control is used to predict the changes of contact states and generated a smooth evolution of the contact force.

4.4 Motion constraint generation

Due to the rapid movements of the obstacle, large changes of h_i in (2.10d) can result in large instantaneous changes in joint torques when solving the control problem as stated in Section 2.1. In order to prevent large instantaneous changes in joint torques, an intuitive method is to modify the obstacle avoidance constraint when obstacles moves rapidly. Based on the formulation of the obstacle avoidance constraint in Section 4.2, the time and position information of the moving obstacle provide possibility to generate a continuous obstacle avoidance constraint by taking advantage of the MPC primitive.

4.4.1 MPC obstacle avoidance constraint generation

The movement of the obstacle along any axis in Cartesian space is independent. Without loss of generality, the position, velocity and acceleration along the $x-$axis for instance are formulated in the following part of this thesis. In this case, a linear discrete-time dynamic model of the obstacle can be created in state space form:

$$\underbrace{\begin{bmatrix} x_{o,\,k+1} \\ \dot{x}_{o,\,k+1} \\ \ddot{x}_{o,\,k+1} \end{bmatrix}}_{\boldsymbol{X}_{o,\,k+1}} = \underbrace{\begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} x_{o,\,k} \\ \dot{x}_{o,\,k} \\ \ddot{x}_{o,\,k} \end{bmatrix}}_{\boldsymbol{X}_{o,\,k}} + \underbrace{\begin{bmatrix} T^3/6 \\ T^2/2 \\ T \end{bmatrix}}_{\mathbf{B}} \dddot{x}_{o,\,k}\,, \tag{4.11}$$

where $\boldsymbol{X}_{o,\,k}$ and $\dddot{x}_{o,\,k}$ are the state vector of the obstacle and the control action at time k, respectively. Matrices $\mathbf{A}$ and $\mathbf{B}$ implicitly describe the linearity of the system.

Using the dynamic model (4.11) recursively, at time k, the relationships between the control action vector and the state vector over a finite time horizon NT is given by:

$$\hat{\boldsymbol{X}}_{o,\,k} = \hat{\mathbf{A}}\boldsymbol{X}_{o,\,k} + \hat{\mathbf{B}}\boldsymbol{U}_o\,, \tag{4.12}$$

where,

$$\hat{\boldsymbol{X}}_{o,\,k} = \begin{bmatrix} \boldsymbol{X}_{o,\,k+1|k} \\ \boldsymbol{X}_{o,\,k+2|k} \\ \vdots \\ \boldsymbol{X}_{o,\,k+N|k} \end{bmatrix}, \quad \hat{\mathbf{A}} = \begin{bmatrix} \mathbf{A} \\ \mathbf{A}^2 \\ \vdots \\ \mathbf{A}^N \end{bmatrix},$$

$$\hat{\mathbf{B}} = \begin{bmatrix} \mathbf{B} & 0 & \cdots & 0 \\ \mathbf{AB} & \mathbf{B} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}^{N-1}\mathbf{B} & \mathbf{A}^{N-2}\mathbf{B} & \cdots & \mathbf{B} \end{bmatrix}, \quad \boldsymbol{U}_o = \begin{bmatrix} \dddot{x}_{o,\,k|k} \\ \dddot{x}_{o,\,k+1|k} \\ \vdots \\ \dddot{x}_{o,\,k+N-1|k} \end{bmatrix}.$$

In order to continuously react to the abrupt movement of the obstacle, the discontinu-

ous position evolution of the obstacle can be used to generate virtual continuous position, velocity and acceleration of the obstacle in (4.3) to avoid large instantaneous changes in joint torques. The position of the obstacle $x^m_{o,\,k}$ can be obtained by distance sensors. Once the robot enters into the safe region at time t_e and moves away at time t_l, the violation distance $d_v > 0$ can be measured and the robot has to react to this violation distance to maintain the safe distance. In this case, the obstacle avoidance constraint switches between inactive and active, and changes of h_i occurs simultaneously. As stated in Section 2.1.3, two kinds of discontinuities appears. In order to handle these discontinuities, several assumptions are made:

1) when the obstacle is outside the safe distance, the obstacle is assumed to lie on the edge of the safe distance to ensure that the obstacle avoidance constraint is always active.

2) the obstacle stays still during the reaction time once the abrupt movement of the obstacle is measured.

Based on these assumptions, the position of the obstacle over the prediction horizon $\boldsymbol{x}^m_{o,\,k} = \left[x^m_{o,\,k+1|k} \;\cdots\; x^m_{o,\,k+N|k}\right]^T$ can be previewed with the safe distance d_s, the violation distance d_v, and the reaction time t_r at time k:

- before the obstacle entering the safe region, $x^m_{o,\,i|k} = d_s, \forall i \in [k+1, \cdots, k+N+1]$.
- after the obstacle entering the safe region, if $i < t_e + t_r, x^m_{o,\,i|k} = d_s$; if $i \geq t_e + t_r, x^m_{o,\,i|k} = d_s - d_v; \forall i \in [k+1, \cdots, k+N+1]$.
- after the obstacle moving away from the safe region, $x^m_{o,\,i|k} = d_s, \forall i \in [k+1, \cdots, k+N+1]$.

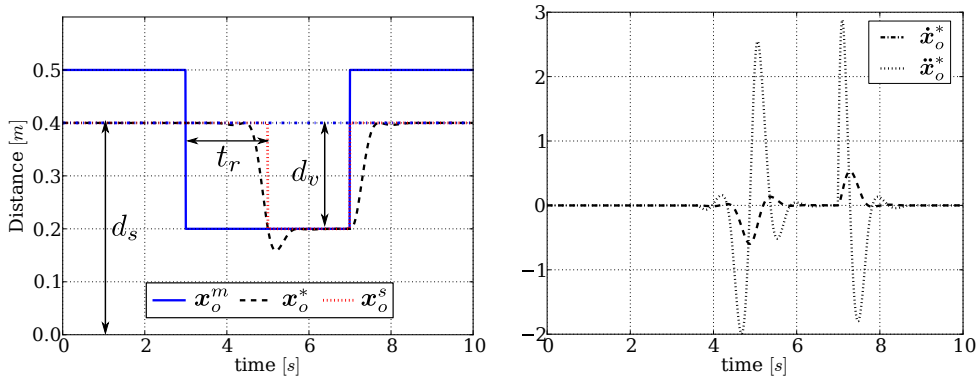
Actually, only the position information of the obstacle can be used as a reference to generate virtual position, velocity and acceleration of the obstalce. Therefore, the predicted positions over the prediction horizon can be extracted from $\hat{\boldsymbol{X}}^x_{o,\,k}$ in (4.12): $\hat{\boldsymbol{X}}_{o,\,x} = \left[x_{o,\,k+1|k} \;\cdots\; x_{o,\,k+N|k}\right]^T$. Similarly, $\hat{\mathbf{A}}_x$ and $\hat{\mathbf{B}}_x$ can be derived from $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$, respectively. Therefore, the predicted position of the obstacle over the horizon can be formulated as:

$$\hat{\boldsymbol{X}}^{x}_{o,\,k} = \hat{\mathbf{A}}_x \boldsymbol{X}_{o,\,k} + \hat{\mathbf{B}}_x \boldsymbol{U}_o \,. \tag{4.13}$$

To approximate the movement of the obstacle using the previewed position information $\boldsymbol{x}^m_o$, the optimization problem of the MPC is formulated similarly to (5.8):

$$\dddot{x}^*_{o,\,k|k} = \begin{cases} \underset{\boldsymbol{U}_o}{\arg\min} & \left\| \hat{\boldsymbol{X}}^{x}_{o,\,k} - \boldsymbol{x}^m_{o,\,k} \right\|^2 + \gamma \|\boldsymbol{U}_o\|^2 & \text{(4.14a)} \\ \text{subject to} & \hat{\boldsymbol{X}}^{x}_{o,\,k} = \hat{\mathbf{A}}_x \boldsymbol{X}_{o,\,k} + \hat{\mathbf{B}}_x \boldsymbol{U}_o & \text{(4.14b)} \\ & \hat{\boldsymbol{X}}^{x}_{o,\,k} \leq \boldsymbol{x}^m_{o,\,k} & \text{(4.14c)} \end{cases}$$

where constraint (4.14c) ensures that the robot has to maintain the safe distance at the end of the reaction time. By solving a QP at time k, $\dddot{x}^*_{o,\,k|k}$ can be calculated and it is used to update $\boldsymbol{X}^*_{o,\,k+1} = \mathbf{A}\boldsymbol{X}^*_{o,\,k} + \mathbf{B}\dddot{x}^*_{o,\,k|k}$. Therefore, the continuous evolution of $\boldsymbol{X}^*_o = [\boldsymbol{x}^*_o\ \dot{\boldsymbol{x}}^*_o\ \ddot{\boldsymbol{x}}^*_o]^T$ can be obtained (similarly for $\boldsymbol{Y}^*_o$ and $\boldsymbol{Z}^*_o$). $\boldsymbol{p}_o$, $\dot{\boldsymbol{p}}_o$ and $\ddot{\boldsymbol{p}}_o$ are thus continuous. Based on (4.3) continuous h^*_i can be obtained.



(a) Evolution of $\boldsymbol{x}^m_o$ and $\boldsymbol{x}^*_o$. (b) Evolution of $\dot{\boldsymbol{x}}^*_o$ and $\ddot{\boldsymbol{x}}^*_o$.

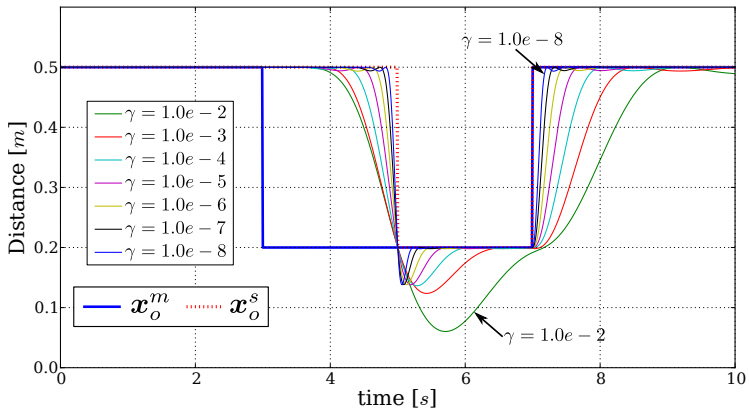
Figure 4.5: The generated continuous position, velocity and acceleration with respect to the discontinuous evolution the obstacle using the MPC with $T = 0.01s$, $NT = 1.5s$ and $\gamma = 10^{-5}$.

To illustrate this concept, a simple example is provided to handle abrupt movements of the obstacle. MPC (4.14) is used with time step $T = 0.01s$, constant ratio $\gamma = 10^{-5}$ and the prediction horizon $NT = 1.5s$. The safe distance d_s and the maximal velocity of the robot v_{max} are set to be $0.4m$ and $0.1m/s$, respectively. The position of the obstacle $x^m_{o,\,k}$ is measured by the distance sensor at each time step. In figure 4.5(a), the obstacle ap-

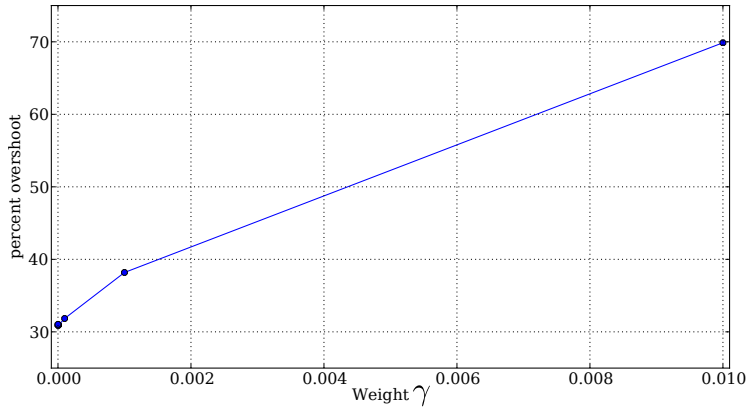
proaches the robot at $t = 3.0$, the sensor measures its position and the violation distance $d_v = 0.2m$ can be computed. Using the resulting reaction time $t_r = 2.0s$, the movement of the obstacle is assumed to occur at $t = 5.0s$. Therefore, the previewed position $\boldsymbol{x}^m_{o,\,k}, \forall k \in [3.0s,\ 5.0s]$ is updated by the delayed position information of the obstacle $\boldsymbol{x}^s_o$. At $t = 7.0s$ the obstacle moves outside of the safe region, the previewed position $\boldsymbol{x}^m_{o,\,k}$ is updated by the current position of the obstacle measured by the sensor. The MPC here is used to minimize the error between the solution $\boldsymbol{X}^x_{o,\,k}$ and the reference $\boldsymbol{x}^m_{o,\,k}$ over the time horizon as well as reduces the instantaneous variation of $\dddot{\boldsymbol{x}}_o$. The resulting $\dddot{\boldsymbol{x}}^*_{o,\,k|k}$ is used to update the virtual position, velocity and acceleration of the obstacle for the next control time. As a result, Figure 4.5(a) shows that the generated $\boldsymbol{x}^*_o$ is continuous compared with the original position evolution of the obstacle. Moreover, MPC also generates continuous $\dot{\boldsymbol{x}}^*_o$ and $\ddot{\boldsymbol{x}}^*_o$ in Figure 4.5(b). However, there exists an overshoot of $\boldsymbol{x}^*_o$ at $t = 5.0s$. It is inherently affected by the weight γ and the prediction horizon NT of the proposed MPC. The influences of the weight and the prediction horizon are analysed and summarized in the following section.

4.4.2 Analysis on the parameters of MPC for motion

The example in Figure 4.5 illustrates that the proposed approach (4.14) can generate continuous evolution of $\boldsymbol{x}^*_o$, $\dot{\boldsymbol{x}}^*_o$ and $\ddot{\boldsymbol{x}}^*_o$ with respect to a discontinuous evolution of the obstacle position $\boldsymbol{x}^m_o$. But the overshoot may degrade the reaction performance of the robot when $\boldsymbol{x}^m_o$ changes largely. Based on (4.14), the overshoot is closely related to the weight γ and the prediction horizon NT of the MPC. The evaluation of the overshoot is defined by the percent overshoot [Nise 07, p. 178]: the amount that the waveform overshoots the steady-state, expressed as a percentage of the steady-state value. In order to effectively minimize the percent overshoot or even eliminate the overshoot, the weight γ and the prediction horizon NT should be optimized. The criteria for choosing γ and NT is analysed and summarized in this section.



(a) Evolution of $\boldsymbol{x}_o^m$ and $\boldsymbol{x}_o^*$.



(b) Evolution of the percent overshoot.

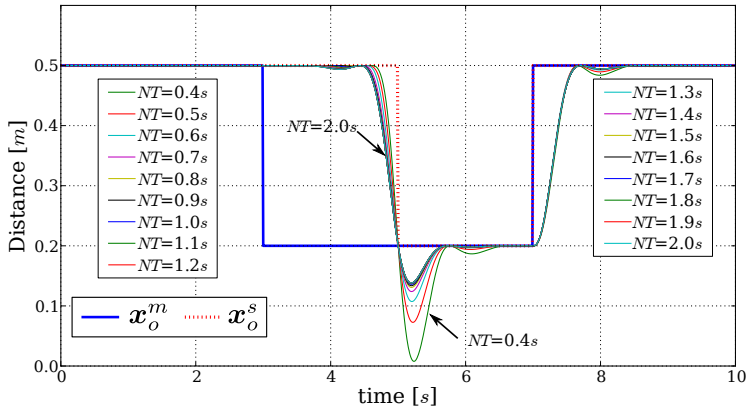
Figure 4.6: The resulting evolution of $\boldsymbol{x}_o^*$ and the percent overshoot using a set of different γs with the prediction horizon $NT = 1.5s$.

The weight

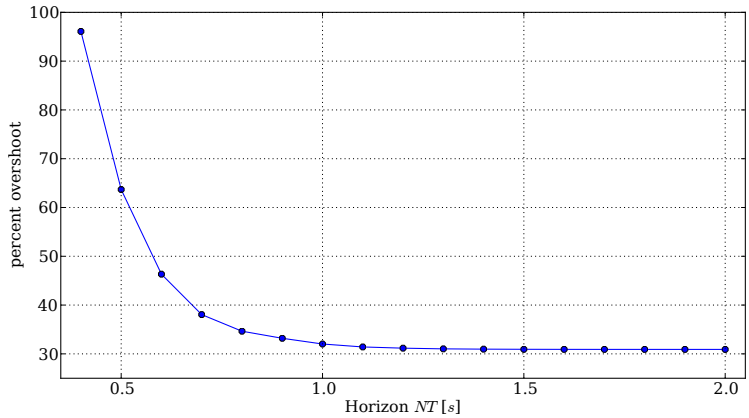
The weight γ regulates the importance between the regulation term $\|\boldsymbol{U}_o\|^2$ and the approximation term $\left\|\hat{\boldsymbol{X}}_{o,\,x} - \boldsymbol{x}_o^m\right\|^2$ in (4.14). Since the weight for the approximation term is 1, a relatively large weight γ means that the maximum jerk $\dddot{\boldsymbol{x}}_o$ is reduced, but the percent overshoot is increased. In this experiment, the simulation scenario is identical to the example shown in Figure 4.5. Given the prediction horizon $NT = 1.5s$, the resulting evolutions of $\boldsymbol{x}_o^*$ with a set of different γs are shown in Figure 4.6(a). Although the weight γ changes from 10^{-8} to 10^{-2}, the overshoot still exits. In Figure 4.6(b), the percent overshoot stays 31% when γ is between 10^{-8} and 10^{-5} and it increases largely as the weight increases from 10^{-4} to 10^{-2}. Therefore, it is impossible to avoid overshoots only

modifying the weight γ. In this case, $\gamma = 10^{-5}$ is proper choice for the MPC.

The length of the horizon window



(a) Evolution of x_o^m and x_o^*.



(b) Evolution of the percent overshoot.

Figure 4.7: The resulting evolution of x_o^* and the percent overshoot using a set of different NTs with the weight $\gamma = 10^{-5}s$.

The prediction horizon NT enables the MPC to react to future changes accordingly. Given the weight $\gamma = 10^{-5}$, Figure 4.7(a) shows the generated continuous evolution x_o^* with respect to the discontinuous x_o^m with a set of different NTs using the MPC. In Figure 4.7(b), the percent overshoot decreases as the prediction horizon increases from $NT = 0.4s$ to $NT = 1.1s$. Then although the prediction horizon continues to increase from $NT = 1.1s$ to $NT = 2.0s$, the percent overshoot stays at the value of 31%, rather than decrease. Therefore, the length of the prediction horizon is useful for decreasing of

the percent overshoot. Additionally, it is observed in Figure 4.7(a) that the MPC does not begin to react to the changes of the obstacle at $t = 3.0s$, although the position change of the obstacle can be known by the prediction horizon $NT = 2.0s$. In this case, the MPC does not fully take advantage of the prediction horizon, in contrast, a long horizon increases computation cost of the MPC. Given the weight $\gamma = 10^{-5}$, the prediction horizon can be between $NT = 1.0s$ and $NT = 1.5s$.

According to the analysis above, the weight γ and the prediction horizon NT play an important role in the reduction of the percent overshoot. The weight determines the instance when the MPC begins to react to the discontinuous changes of the obstacle, instead of reacting it once the changes is previewed within the prediction horizon. Therefore, it is essential to choose the prediction horizon after the weight is decided. However, the overshoot cannot be removed by tuning the weight γ and the prediction horizon NT. It is necessary to find a way to deal with this problem when solving the MPC problem (4.14).

Dynamic weighting matrix

Through the analysis on the weight γ and the prediction horizon NT, it is observed that the weight plays a more important role in the generated results than the prediction horizon. The weight strictly guarantees the approximation to the original position. Given a constant weight, the steady increase of prediction horizon cannot enable the MPC to eliminate the overshoots, but increase the computation cost. Moreover, it may not be realistic to know the information too much in advance. The constant weight γ limits the potential of the prediction horizon. In order to reduce the magnitude of the overshoots or even eliminate the overshoots, a dynamic weighting matrix $\mathbf{D}$ is added into (4.14):

$$\ddot{x}^{*}_{o,\,k|k} = \begin{cases} \underset{\boldsymbol{U}_o}{\arg\min} & \left\| \hat{\boldsymbol{X}}^{x}_{o,\,k} - \boldsymbol{x}^{m}_{o,\,k} \right\|^2_{\mathbf{D}} + \gamma \|\boldsymbol{U}_o\|^2 \\ \text{subject to} & \hat{\boldsymbol{X}}^{x}_{o,\,k} = \hat{\mathbf{A}}_x \boldsymbol{X}_{o,\,k} + \hat{\mathbf{B}}_x \boldsymbol{U}_o \\ & \hat{\boldsymbol{X}}^{x}_{o,\,k} \leq \boldsymbol{x}^{m}_{o,\,k} \end{cases} . \tag{4.15}$$

where $D = diag(d_j, d_{j+1}, \ldots, d_{j+N})$ is a diagonal matrix, and $d_j \in (0, 1], \forall j \in [k+1, k+N]$

determines the weight associated to each sample in the time horizon. Each d_j is computed with respect to the variation of $\boldsymbol{x}_o^m$:

$$d_j = \left(\frac{max(\boldsymbol{x}_{o,\,k}^m) - x_{o,\,j|k}^m + \lambda}{max(\boldsymbol{x}_o^m) - min(\boldsymbol{x}_o^m) + \lambda} \right)^g , \tag{4.16}$$

where λ is a regulation term to avoid a division by zero, g is the power to regulate the ratio between d_j and γ_j. d_j is dynamically changing in the time horizon $[k+1, k+N]$:

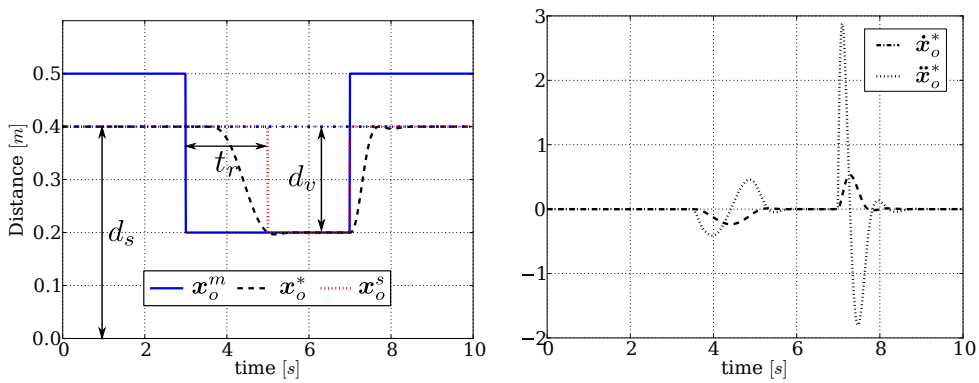
- if $max(\boldsymbol{x}_o^m) = min(\boldsymbol{x}_o^m)$, the obstacle dose not move during the prediction horizon $[k+1, k+K]$. In this case, $d_j = 1.0, \forall j \in [k+1, k+N]$. γ_j is set to be a very small value, for example 10^{-5}. This means that the approximation term is always more important than the minimization of $\boldsymbol{U}_o$. In this case, the generated $\boldsymbol{x}_o^*$ is identical to the reference.

- if $max(\boldsymbol{x}_o^m) \neq min(\boldsymbol{x}_o^m)$, the position of the obstacle changes in the time horizon. There are two extreme situations to determine d_j.

  1) When $x_{o,\,j|k}^m = max(\boldsymbol{x}_o^m)$, $d_j = \left(\frac{\lambda}{max(\boldsymbol{x}_o^m) - min(\boldsymbol{x}_o^m) + \lambda}\right)^g$. In order to easily regulate d_j compared with γ_j, λ is set to be $(max(\boldsymbol{x}_o^m) - min(\boldsymbol{x}_o^m))/9$. Thus d_j equals to $\left(10^{-1}\right)^g$. Then g can be chosen according to the value of γ to ensure that the minimization of $\boldsymbol{U}_o$ is more important than the approximation term within this prediction horizon. Therefore, the approximation to the original reference is sacrificed to reduce variations of $\boldsymbol{U}_o$ as soon as the large changes is previewed over the prediction horizon.

  2) When $x_{o,\,j|k}^m = min(\boldsymbol{x}_o^m)$, $d_j = 1.0$. The approximation to the large variations of the reference is guaranteed with respect to minimization of $\boldsymbol{U}_o$. The overshoots can be minimized due to sacrifice of the approximation to the invariant reference.

The results using the dynamic weighting matrix $\mathbf{D}$ are shown in Figure 4.8, which shows that the overshoots are effectively minimized on $\boldsymbol{x}_o^*$. Using dynamic weighting matrix, the difference between $max(\boldsymbol{x}_o^m)$ and $min(\boldsymbol{x}_o^m)$ within the prediction horizon can be known at $t = 3.5$. In this case, g is chosen to be 7 according to $\gamma_j = 10^{-5}$. Thus



(a) Evolution of x_o^m and x_o^*. (b) Evolution of $\dot{x}_o^*$ and $\ddot{x}_o^*$.

Figure 4.8: The generated continuous position, velocity and acceleration with respect to the discontinuous evolution the obstacle with the dynamic weighting matrix $\mathbf{D}$, given $T = 0.01s$, $NT = 1.5s$ and $\gamma = 10^{-5}$. The percent overshoot is effectively reduced.

d_j is not constant but variant within the prediction horizon: $d_j = 10^{-7}, \forall j \in [3.5s, 4.9s]$ and $d_j = 1.0, j = 5.0s$. This means the approximation term is less important than the regulation term and the approximation to the reference position is sacrificed to reduce the variation of the jerk from $t = 3.5s$ to $t = 4.9s$. Moreover, the approximation at $t = 5.0s$ is of the most importance and it is satisfied mostly. Therefore, the percent overshoot is effectively minimized.

4.5 Results

The proposed approach is applied to the control of a 38-DoF iCub robot in simulation. The experiments are carried out in Arboris-Python simulator. The robot is actuated by joint torques to perform tasks in the operational space while satisfying obstacle avoidance constraints. The results of the proposed approach (4.14) are compared with those of the baseline approach (2.10).

In this scenario, the robot is standing on the ground and its left hand is tracking a trajectory above a table. Meanwhile, the left hand has to avoid the collision with the object, which moves in front of the robot. The safe distance d_s is set to be $0.06m$ in this experiment. In order to reach a far position, its right hand is in contact with the table to obtain an additional support that increases its reaching ability (see Figure 4.9).

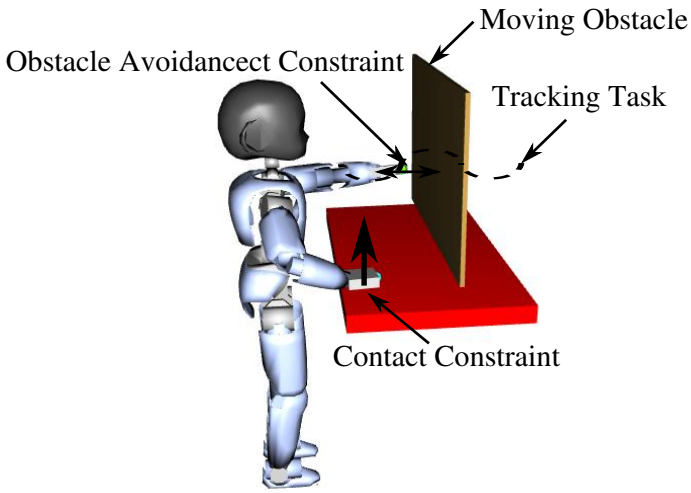
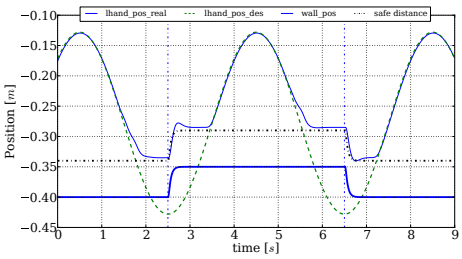


Figure 4.9: Snapshot of the robot tracking a trajectory with one hand supported by a table while avoiding collision with a moving object.

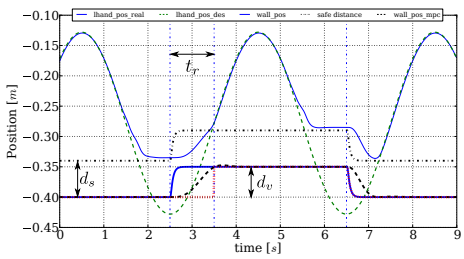
In this experiment, the left hand is tracking a sinusoid trajectory along x-direction in the operational space. The contact constraints handle the right hand contact with the table and the foot contacts with the ground. In order not to collide with the moving object, the obstacle avoidance constraint is created between the robot and the object as described in Section 4.2. The δt in (4.3) is chosen to be larger than the time step T to avoid the chattering when the obstacle constraint changes from being inactive to being active [Park 98, Salini 12a]. To demonstrate the effectiveness of the proposed approach, the results using the QP reactive control framework without and with predictive primitive are compared to deal with abrupt movements of the object.

Figure 4.10 shows the resulting trajectory of the left hand, the position of the obstacle, force evolution of the right hand, joint torques and torque derivatives *without* and *with* the proposed MPC approach. In Figure 4.10(a), the obstacle enters into the safe region of the robot with a violation distance $d_v = 0.05m$ at $t = 2.5s$ and then moves away at $t = 6.5s$. Under the obstacle avoidance constraint, the left hand of the robot moves instantaneously to maintain the safe distance between the robot and the obstacle. Thus the force on the right hand undergoes large changes in Figure 4.10(c) and large peak of joint torques and torque derivatives are clearly observed in Figure 4.10(e).

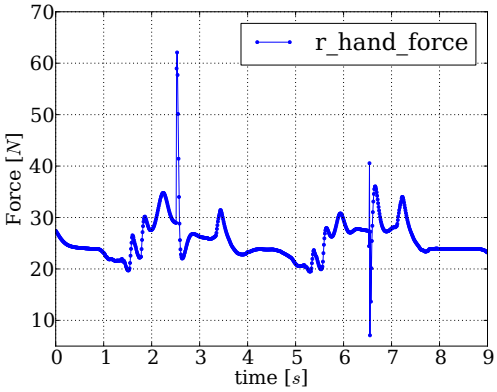
The result in Figure 4.10(b) shows that the proposed approach can smooth the ob-



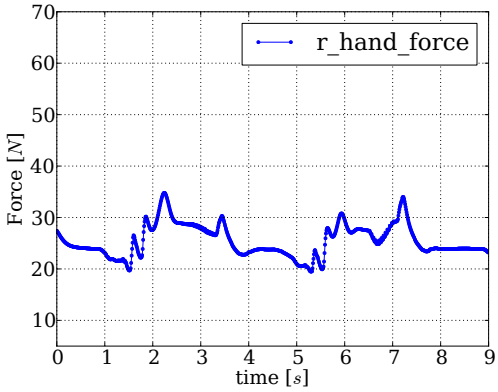
(a) The left hand moves rapidly and instantaneously to react to the movement of the obstacle.



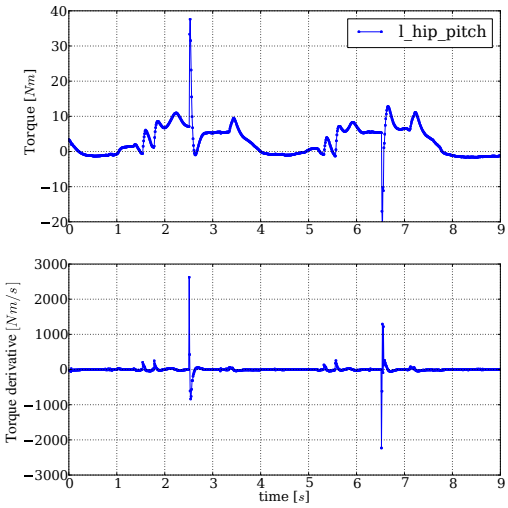
(b) The left hand moves smoothly with respect to the MPC generated continuous constraint.



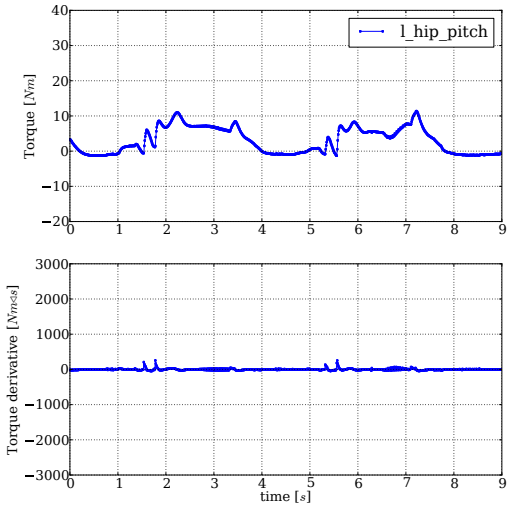
(c) Evolution of the force on the right hand. Large changes occur when the left hand rapidly moves.



(d) The right hand force evolves smoothly with respect to continuous obstacle avoidance constraint.



(e) The large torque changes and high torque derivatives are observed.



(f) The torque derivatives are significantly reduces.

Figure 4.10: The resulting trajectory, right hand force, joint torques and torque derivatives *without* MPC (left) and *with* MPC (right).

stacle avoidance constraint with respect to the rapid movement of the obstacle. Once the sudden movement of the robot is measured by the distance sensor at $t = 2.5s$ and $t = 6.5s$, the reaction time $t_r = 1.0s$ and the position of the obstacle is employed by the MPC to generate the virtual continuous evolution of the position, velocity and acceleration of the obstacle. With this new continuous obstacle avoidance constraint in the QP reactive controller, the left hand reacts to the constraint correspondingly and moves smoothly. The smooth evolutions of the right hand force, joint torques and torque derivatives demonstrate the effectiveness of the proposed approach in Figure 4.10(b, d and f). Large changes in joint torques are significantly reduced.

4.6 Conclusion

In this chapter, the proposed MPC is used to react to sudden movements of the obstacle over a receding finite horizon and generate continuous position, velocity and acceleration of the obstacle, which is used to produce a continuous obstacle avoidance constraint. Based on the overall control framework proposed in Figure 4.1, the generated continuous obstacle avoidance constraint is integrated into a reactive QP controller. Thus the rate of change in actuation torques is minimized. Simulations pertaining to the discontinuous movements of the obstacle show that the proposed approach effectively reduces the instantaneous changes in joint torques.

The additions and removals of contact constraints in the controller induced by establishing and breaking the contact can also cause large instantaneous changes in joint torques. In the following chapter, maximum and minimum allowable contact forces are proposed to restrict changes of contact states to handle the discontinuous changes of contact states. A predictive primitive focuses on the generation of continuous maximum and minimum contact force constraint, resulting in minimized changes in joint torques.

Chapter 5
Force Constraint Generation

In this chapter, a predictive primitive is proposed to minimize instantaneous changes in joint torques using Model Predictive Control. The MPC previews the evolution of contacts in time and generates continuous maximum and minimum allowable force constraints for a QP reactive controller. Section 5.1 introduces problems caused by changes of contact constraints and recent methods to solve them. Section 5.2 proposes a continuous contact force generation approach based on MPC and analyses the smoothness of the generated force constraint with respect to the parameters of MPC. An overall control framework is developed to integrate the generated smooth contact force constraint into a low-level reactive controller in Section 5.3. To illustrate the effectiveness of the proposed control framework, Section 5.4 shows the simulation results on the iCub robot in different scenarios, such as standing up from a sitting posture and the lifting and lowering of a foot while standing. Section 5.5 concludes the chapter by summarizing the contributions and limitations.

5.1 Introduction

Free-floating robots, such as humanoids, are required to make contacts with the environment. For example, when the robot stands on the ground, contacts are intrinsically required for balance. In the control problem, contacts can be controlled in two different ways. First of all, an effective way to realize direct force control [Whitney 77] is to regulate the contact force to a desired value [Raibert 81, Whitney 87, Chiaverini 99]. Secondly, contacts are usually accounted for as constraints in the controller in order to maintain contacts [Abe 07, Wensing 13, Herzog 14] and are expressed in two parts: one is the friction cone constraint [Klein 90, Muico 09]; and the other is the linear complementary condition [Pang 96]. For tasks such as walking, the contact between the feet and the ground is usually treated as constraints and must be established and broken in order to move

around in the environment. At the instant when a contact is broken, the contact force decreases to zero. On the contrary, when a contact is established, the contact force may suddenly increase from zero. The sudden addition or removal of a contact constraint in the controller due to the establishment or break of a contact may result in large sudden changes in joint torques.

To prevent large instantaneous changes in joint torques due to changes of contact constraints, one possible approach is to prevent large changes of contact forces in the controller. The approach presented in [Salini 12a] regulates the contact force as a task of the controller. The normal contact force gradually decreases to zero before the contact is broken. While this approach provides interesting results, it is strongly related to the method chosen to describe and solve task transitions. In this aspect, it does not provide a generic way of dealing with the problem due to changes of contact states. Moreover, these tasks are parametrizable but provide a stereotypical reaction, the dynamics of which is pre-planned and that is thus not well suited to deal with very dynamic situations.

Similarly to the work in [Salini 12a], high-level task planning and motion patterns are helpful to estimate the evolutions of contact states. For example, when a humanoid walks to a target, the planned walking pattern can be computed in advance [Kajita 03, Vukobratović 04]. The discrete contact evolutions thus can be known ahead of time. Moreover, motion patterns can be used to predict when the contact will be established. For example, when the upper-body of a robot sitting on a chair, moves forward and the contact force between the robot and the chair is decreasing, this can be seen as a signal that the robot will break the contact soon. The known future information of contact states enables the robot to react in advance.

Instead of using pre-planned strategies to manage contact transitions in a reactive controller, predictive approaches endow the reactive controller with both robustness and reactivity by taking advantage of the future contact states as illustrated in Figure 5.1. Using a receding finite horizon, MPC can preview the establishment and break of contacts in advance and generate continuous contact force constraints to achieve continuous contact transitions with a force model. Most importantly, MPC provide a systematic method of dealing with current and future constraints on inputs and states. In this chapter, MPC

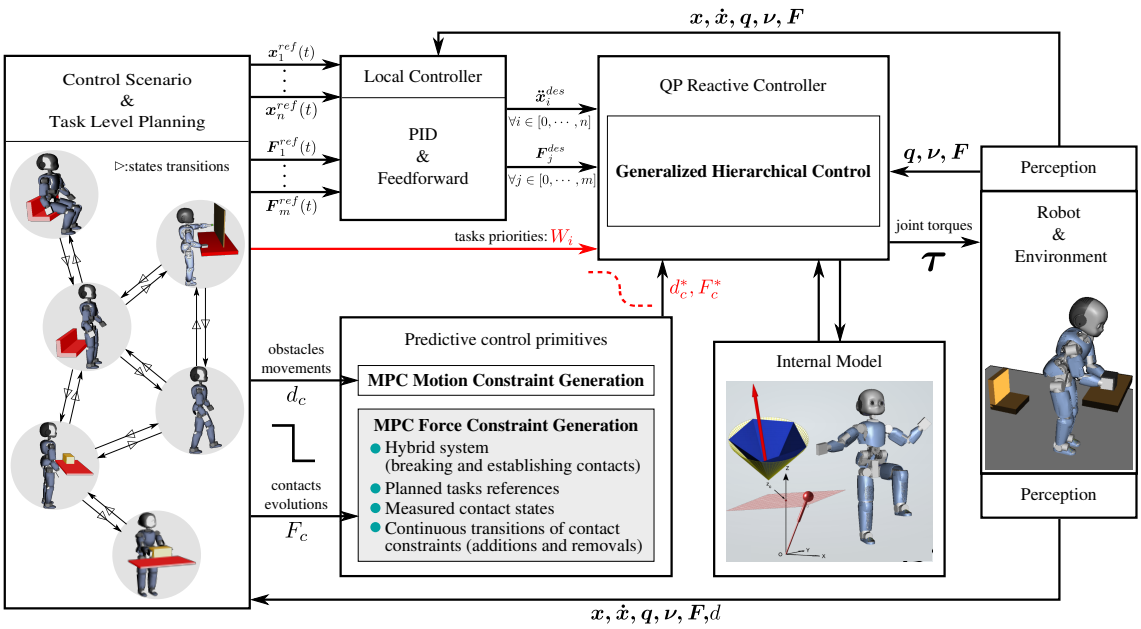


Figure 5.1: The control framework including a predictive control primitive and a QP reactive controller is proposed to minimize the rate of change in joint torques due to changes of contacts states.

is used to produce a time-varying allowable force constraint that is integrated into a QP reactive controller. With this approach, the rate of change in joint torques is effectively minimized.

5.2 Force Constraint Generation

Due to the establishment or break of a contact, the appearance or disappearance of a external contact force $\boldsymbol{F}_{c,\,i}$ in (2.10b) as well as contact constraints in (2.12) can result in large instantaneous changes in joint torques when solving the QP. In order to prevent this, an intuitive method is to minimize changes of contact forces when a contact is established or broken. In this section, an approach using MPC is proposed to preview changes in contacts in advance and have time to generate smooth contact forces, the rate of change of which is minimized.

5.2.1 Contact Constraint

For a single point in contact, the contact force $\boldsymbol{F}_{c,\,i}$ consists of the tangential component $\boldsymbol{F}_{t,\,i}$ and the normal component $\boldsymbol{F}_{n,\,i}$, with $\boldsymbol{F}_{n,\,i} = F_{n,\,i}\boldsymbol{n}$ and $\boldsymbol{n}$ being the normal vector to the contact surface. In order to maintain a non-sliding contact between two objects, two constraints have to be satisfied. One is contact existing constraint, and the other is friction cone constraint.

Contact Existing Constraint

Contact existing constraint is prerequisite to maintain a contact, which includes the kinematics condition and force condition. Kinematics condition means that there is no relative movement between two objects. Assuming that the environment is static and rigid, the contact is maintained only if the velocity of the contact point on the robot's body is zero: $\ddot{\boldsymbol{x}}_c = 0$. This condition can be expressed in terms of joint accelerations:

$$\ddot{\boldsymbol{x}}_c = \mathbf{J}_c(\boldsymbol{q})\dot{\boldsymbol{\nu}} + \dot{\mathbf{J}}_c(\boldsymbol{q},\boldsymbol{\nu})\boldsymbol{\nu} = 0\,, \tag{5.1}$$

At each contact point, the contact force condition is a complementary condition for the kinematics condition [Pang 96]. The contact force along the normal direction is non-negative. This means that either the robot pushes against the environment without any movement, or the robot does not apply a force on the environment.

$$F_n \geq 0\,. \tag{5.2}$$

These two conditions constitute the contact existing constraint, which can maintain a contact between the robot body and the environment.

Friction Cone Constraint

To ensure a non-sliding contact, the contact force for each contact point is constrained within the Coulomb friction cone: $\|\boldsymbol{F}_t\| \leq \mu\,\|\boldsymbol{F}_n\|$, where μ is the friction coefficient. The friction cone is usually approximated by a polygonal cone so that the non-sliding contact constraint can be expressed as a set of linear inequality constraints [Siciliano 08]:

$$\mathbf{C}\boldsymbol{F}_c \leq \mathbf{0}\,, \tag{5.3}$$

where $\mathbf{C}$ represents the linearized cone.

Allowable Force Limits

In order to prevent large instantaneous changes in joint torques, large changes of contact forces must be minimized when a contact is established or broken. In practice, contact forces can be constrained by various contact constraints according to different scenarios. For example, when there is no contact, contact force must be 0; and contact force is sometimes required to be limited in order to avoid damage to the environment, or to be larger than a certain value to be able to manipulate an object. In this case, constraint (5.2) can be modified by adding a maximum allowable force $F_{max} \geq 0$ and a minimum allowable force $F_{min} \geq 0$:

$$F_{min} \leq F_n \leq F_{max}\,, \tag{5.4}$$

where F_{max} and F_{min} are both related to the scenario.

5.2.2 MPC Force Constraint Generation

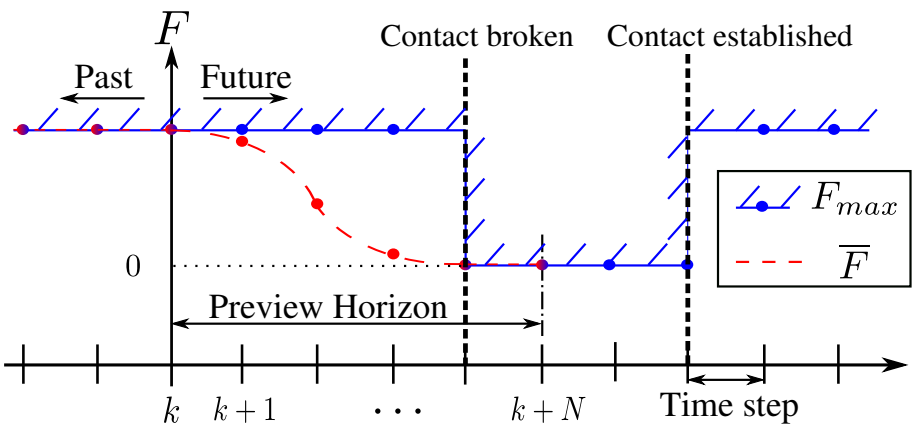


Figure 5.2: Generating continuous evolution of the maximum allowable force using MPC at time k.

In this chapter, MPC is used to reduce the changes of F_{max} and F_{min} online with the

goal to minimize the rate of change in joint torques. Given the evolution of F_{max} for example, the proposed MPC scheme is shown in Figure 5.2. The changes of F_{max} can be known in advance and accounted for by the MPC through a preview window of N steps with a sampling time T. The aim of the MPC is to generate continuous maximum allowable force $\overline{F}$ and continuous minimum allowable force $\underline{F}$. Then, they can replace F_{max} and F_{min} in constraint (5.4):

$$\underline{F} \leq F_n \leq \overline{F} \,. \tag{5.5}$$

The following part of this section explains the computation of $\overline{F}$ and $\underline{F}$.

A discrete-time linear model of the force can be expressed as:

$$\overline{F}_{k+1} = \overline{F}_k + \dot{\overline{F}}_k T \,, \tag{5.6}$$

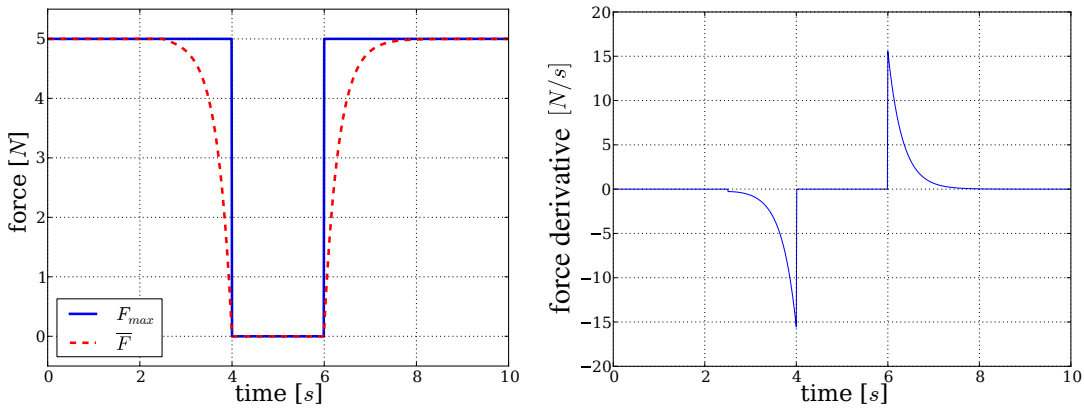
$$\underline{F}_{k+1} = \underline{F}_k + \dot{\underline{F}}_k T \,, \tag{5.7}$$

$\overline{F}_k$ and $\underline{F}_k$ are the maximum and the minimum allowable force at time step k, respectively. $\dot{\overline{F}}_k$ and $\dot{\underline{F}}_k$ are the first-order time derivative of $\overline{F}_k$ and $\underline{F}_k$, respectively. $\overline{F}_k$ and $\underline{F}_k$ should be optimized according to the original profile of F_{max} and F_{min}. Within the time horizon NT, the MPC for generating optimized $\overline{F}$ and $\underline{F}$ can be formulated as:

$$\begin{matrix} \dot{\overline{F}}_k^* \\ \dot{\underline{F}}_k^* \end{matrix} = \begin{cases} \underset{\dot{\overline{F}}_k, \ldots, \dot{\overline{F}}_{k+N} \\ \dot{\underline{F}}_k, \ldots, \dot{\underline{F}}_{k+N}}{\arg\min} & \sum_{j=k}^{k+N} \|\overline{F}_j - F_{max,j}\|^2 + \alpha \|\dot{\overline{F}}_j\|^2 \\ & \quad + \|\underline{F}_j - F_{min,j}\|^2 + \beta \|\dot{\underline{F}}_j\|^2 \qquad (5.8a) \\ \text{subject to} & \overline{F}_{j+1} = \overline{F}_j + \dot{\overline{F}}_j T, \ \forall j \in [k, k+N] \qquad (5.8b) \\ & \underline{F}_{j+1} = \underline{F}_j + \dot{\underline{F}}_j T, \ \forall j \in [k, k+N] \qquad (5.8c) \\ & F_{min,j} \leq \underline{F}_j \leq \overline{F}_j \leq F_{max,j}, \ \forall j \in [k, k+N] \qquad (5.8d) \end{cases}$$

where (5.8d) ensures that at each time step j every solution $\mathbb{X}^*$ that satisfies the constraint (5.5) also satisfies the constraint (5.4). The coefficient α is the weight, governing the importance between the changes of $\overline{F}$ and the deviation from the original maximum allowable force. β has the same effect on $\underline{F}$ as α on $\overline{F}$. For example, a larger value of α enhances the reduction of the changes of $\overline{F}$; however, the constraint (5.4) may turn to

be more conservative and actuation capacities may be restricted. In (5.8), at time step k, $\dot{\overline{F}}_k^*$ and $\underline{F}_k^*$ can be computed simultaneously by a Linear Quadratic Program solver [Nocedal 06]. The generated maximum allowable force is $\overline{F}_k = \overline{F}_{k-1} + \dot{\overline{F}}_k^* T$ and the generated minimum allowable force is $\underline{F}_k = \underline{F}_{k-1} + \dot{\underline{F}}_k^* T$.



(a) Evolution of $\overline{F}$. (b) Evolution of $\dot{\overline{F}}^*$.

Figure 5.3: The resulting continuous maximum allowable force with respect to the original reference.

To illustrate the proposed MPC approach, a simple example of generating a continuous evolution of $\overline{F}$ given a discontinuous F_{max} is provided with $T = 0.01s$, $\alpha = 10^{-1}$ and $NT = 1.5s$. In this example, F_{min} is 0 all the time. In Figure 5.3, it can be observed that the generated maximum allowable force $\overline{F}$ evolves continuously with less changes compared with the original F_{max}. $\overline{F}$ can gradually decrease to zero before a contact is broken and increase gradually after a contact is established. The MPC begins to reduce $\overline{F}$ at $t = 2.5s$ when it detects the changes of F_{max} in the preview window. At $t = 4.0s$ and $t = 6.0s$ when the large changes occur on F_{max}, the magnitude of force derivatives is significantly reduced from **500N/s** of the original evolution of force F_{max} to **15.5N/s** of the MPC generated force $\overline{F}$.

Note that an alternative method to address this problem is to generate continuous profile of F_{max} by applying a polynomial spline [Lin 83, Lambrechts 04]. However, the advantage of using MPC here is to be able to handle complex constraint profiles in an automatic and generic way to ensure that the optimized $\overline{F}$ constraint is compatible with $\underline{F}$ constraint as well (see Figure 5.4), whereas the use of polynomial approaches require

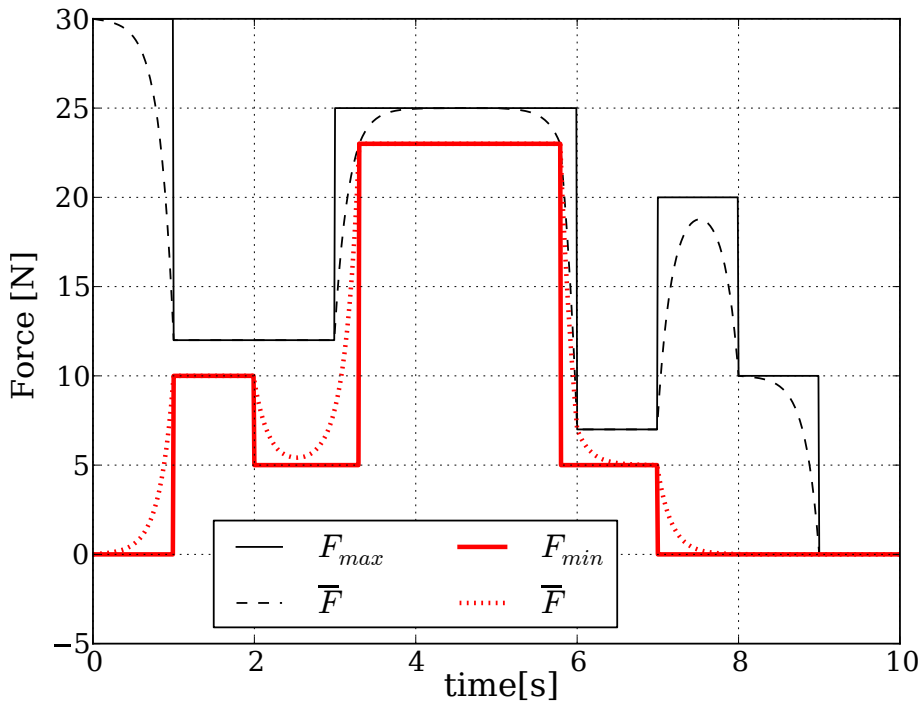


Figure 5.4: Continuous evolutions of the maximum and minimum allowable force are generated simultaneously using the MPC.

manual choose of the start and end points of the segment and manual tuning of polynomial parameters. Moreover, the use of polynomial may generate conflicts between one constraint and the other constraints.

5.2.3 Sensitivity Analysis of The Proposed MPC Scheme

The example in Figure 5.3 demonstrates that the proposed approach (5.8) can generate continuous $\overline{F}$ with respect to a discontinuous F_{max}. The maximum force derivative affects the magnitude of changes in joint torques: the larger the maximum force derivative is, the larger changes in joint torques will occur. In (5.8), the weight α and the prediction horizon NT have a great influence on the maximum force derivative. In order to effectively minimize the maximum force derivative, the weight α and the prediction horizon NT should be optimized. In this section, we take the generation of $\overline{F}$ for example to analyse and summarize the criteria for choosing α and NT. These criteria are also the same to the generation of $\underline{F}$.

Sensitivity Analysis on The Weight α (The same to β)

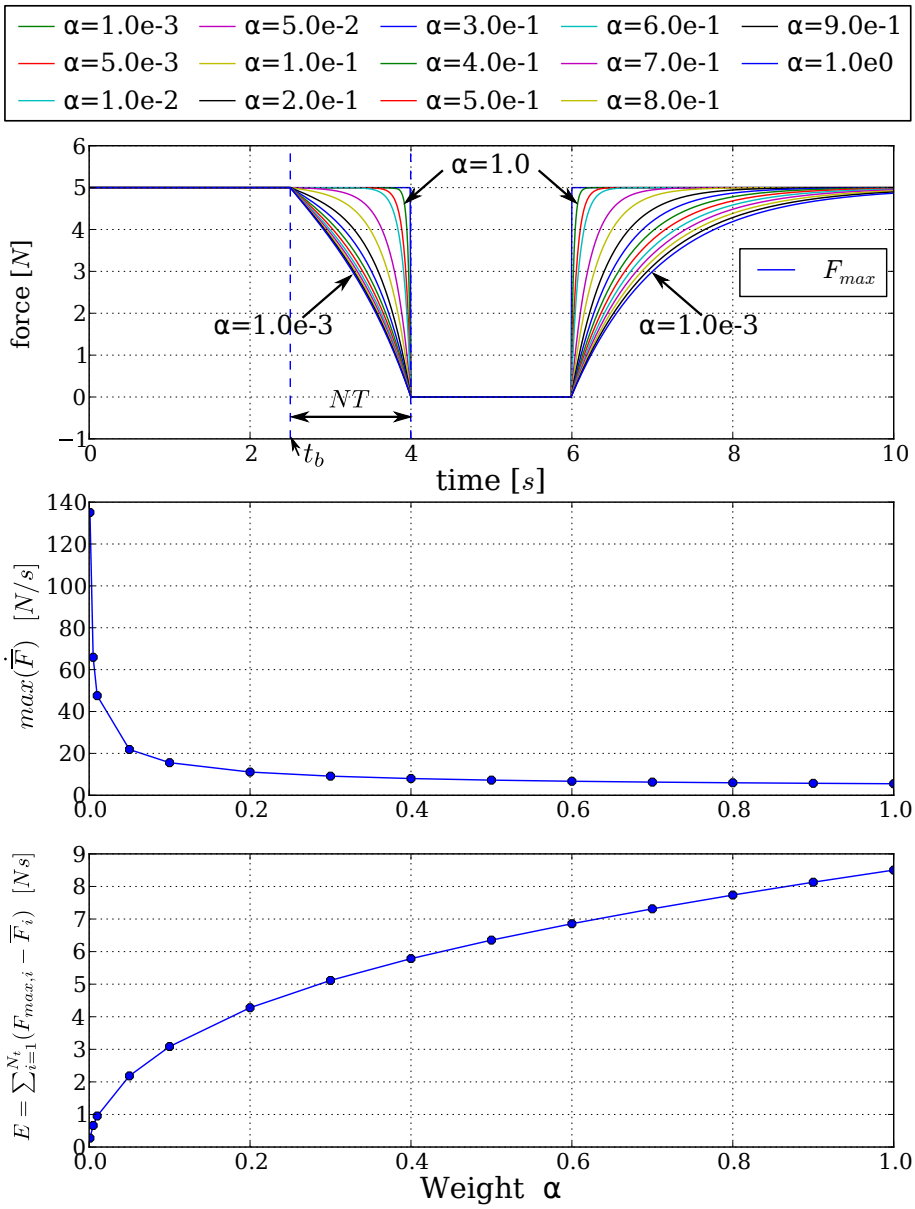


Figure 5.5: Evolution of $\overline{F}$, the maximum force derivative and the approximation error $E(F)$ using a set of different αs with the prediction horizon $NT = 1.5s$.

The weight α governs the importance between the changes of $\overline{F}$ and the approximation to the original maximum allowable force F_{max} in (5.8). In this thesis an approximation

error is defined by the variation between $\overline{F}$ and F_{max} over the whole time period $N_t T$:

$$E = \sum_{i=1}^{N_t} (F_{max,i} - \overline{F}_i) \ . \tag{5.9}$$

The results generated by the MPC using a set of different αs are shown in Figure 5.5. The F_{max} decreases suddenly to 0 at $t = 4.0s$. The MPC can detect this decrease at $t_b = 2.5s$, given the prediction horizon $NT = 1.5s$. The approximation error E increases with the increase of α, in contrast, the maximum force derivative decreases correspondingly. The reduction of the maximum force derivative is at the expense of the closeness to the reference F_{max}. In fact, when α increases from 0.05 to 1.0, the maximum force derivative is slightly reduced, but the approximation error E increases largely. Therefore, a larger value of α does not guarantee a better solution, on the contrary, the constraint (5.4) may turn to be more conservative, especially during the increase phase from $t = 6.0s$ to $t = 10.0s$, where the MPC takes too much time to approach to the original constraint. As a result, actuation capacities may be restricted by the new conservative constraint without more effective reduction on maximum force derivative.

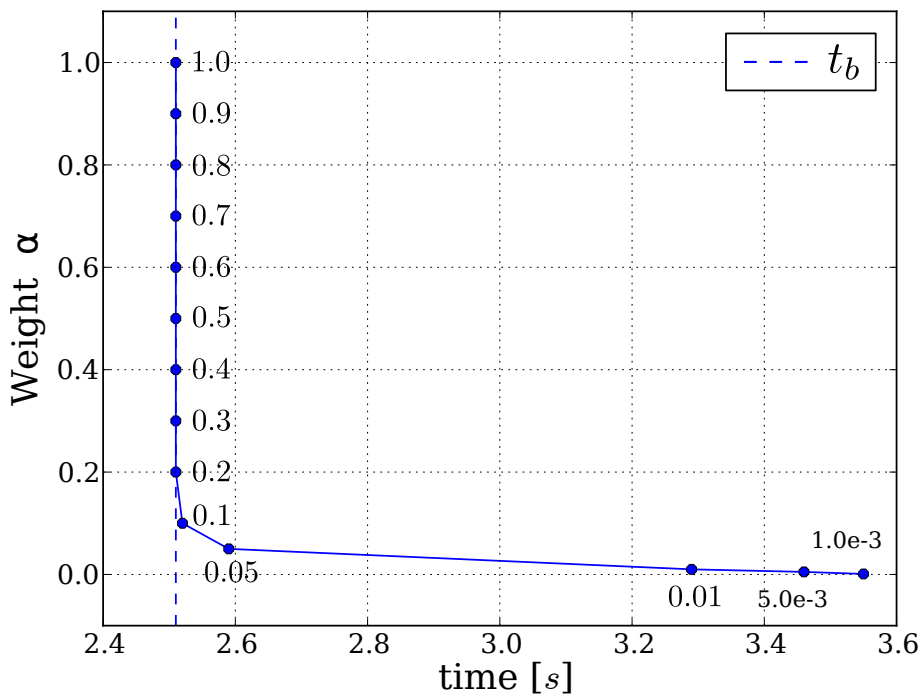


Figure 5.6: The instant when the MPC begins to reduce $\overline{F}$ with a set of different αs.

In addition, it can be observed in Figure 5.5 that the MPC does not begin to reduce $\overline{F}$ at $t_b = 2.5s$ when the decrease of F_{max} is detected using a relatively small weight. The precise instant when the MPC begins to reduce $\overline{F}$ using a set of different αs is shown in Figure 5.6. The MPC begins to react to the decrease of F_{max} at $t = 2.5s$ if the weight α is between 0.2 and 1.0. However, although the MPC detects the decrease of F_{max} at $t_b = 2.5s$, the MPC does not start to reduce $\overline{F}$ until it has to do if the weight is between 10^{-3} and 0.1. In this case, the MPC does not fully take advantage of the prediction horizon, and a long prediction horizon increases computational burden of the MPC. Therefore, in order to efficiently minimize the maximum force derivative, the prediction horizon should be optimized according to the weight.

Sensitivity Analysis on The Prediction Horizon

The prediction horizon NT enables the MPC to react to future changes. Given the weight $\alpha = 0.2$, Figure 5.7 shows the generated continuous evolution of $\overline{F}$, the maximum force derivative and the approximation error E with a set of different NTs using the MPC. The maximum force derivative decreases as the prediction horizon increases from $NT = 0.1s$ to $NT = 1.0s$. Then although the prediction horizon continues to increase from $NT = 1.0s$ to $NT = 3.0s$, the maximum force derivative just decreases a little bit. The approximation error E decreases to the minimum with the increase of the prediction horizon from $NT = 0.1s$ to $NT = 0.6s$, and it just increases a little bit when the prediction horizon increases from $NT = 0.6s$ to $NT = 3.0s$. In Figure 5.7, it is observed that the MPC begins to reduce $\overline{F}$ at $t = 2.5s$ and $t = 3.0s$ using $NT = 1.5s$ and $NT = 1.0s$, respectively. However, the maximum force derivatives are the same value of $11N/s$. The approximation error of $NT = 1.5s$ is slightly larger than that of $NT = 1.0s$. This demonstrates that a long prediction horizon is not the best choice for minimizing the maximum force derivative when the weight is fixed. Moreover, when the prediction horizon NT is $3.0s$, the MPC does not begins to reduce $\overline{F}$ at $t_b = 1.0s$. This result shows that the weight determines the maximum prediction horizon, which the MPC needs to begin to react to the changes of the reference.

The weight α and the prediction horizon NT both affect the evolution of the generated

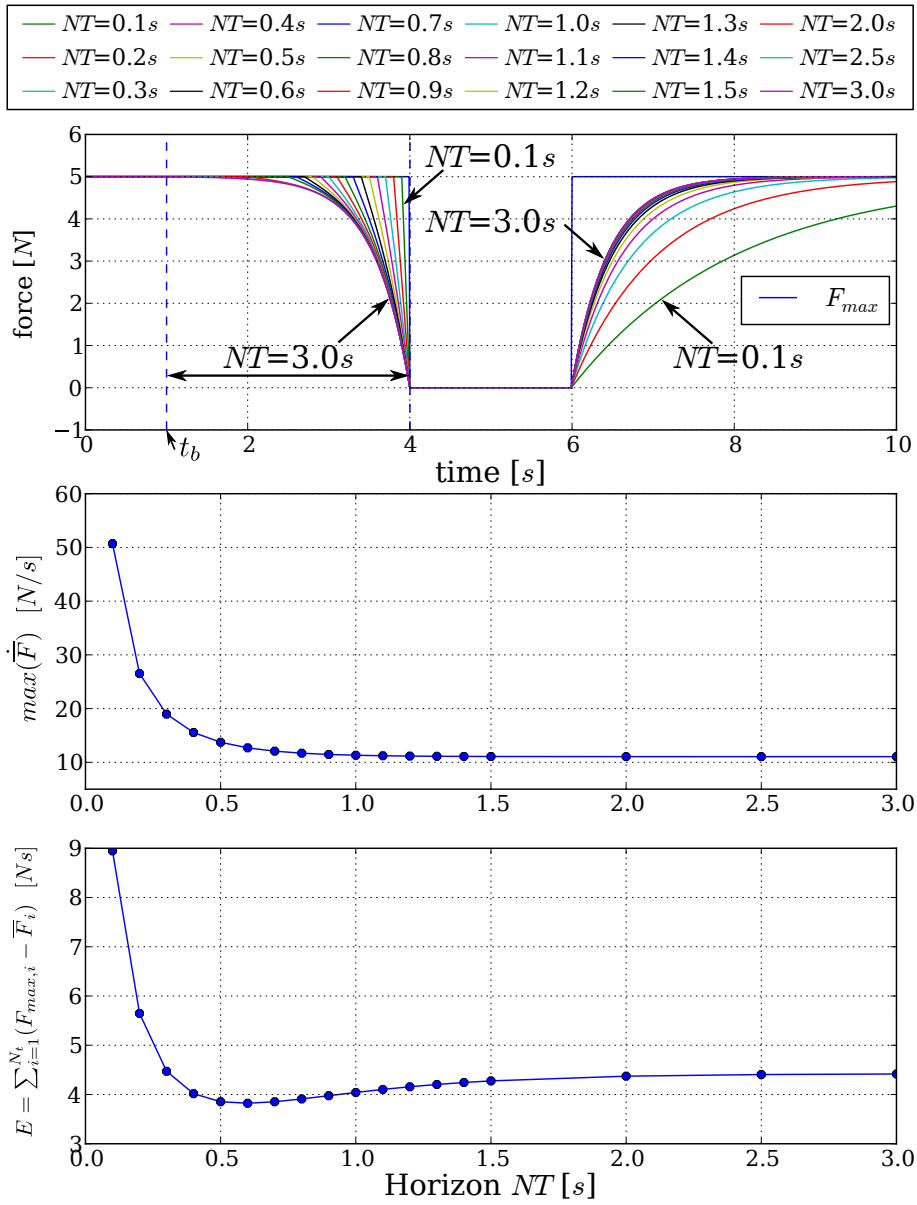
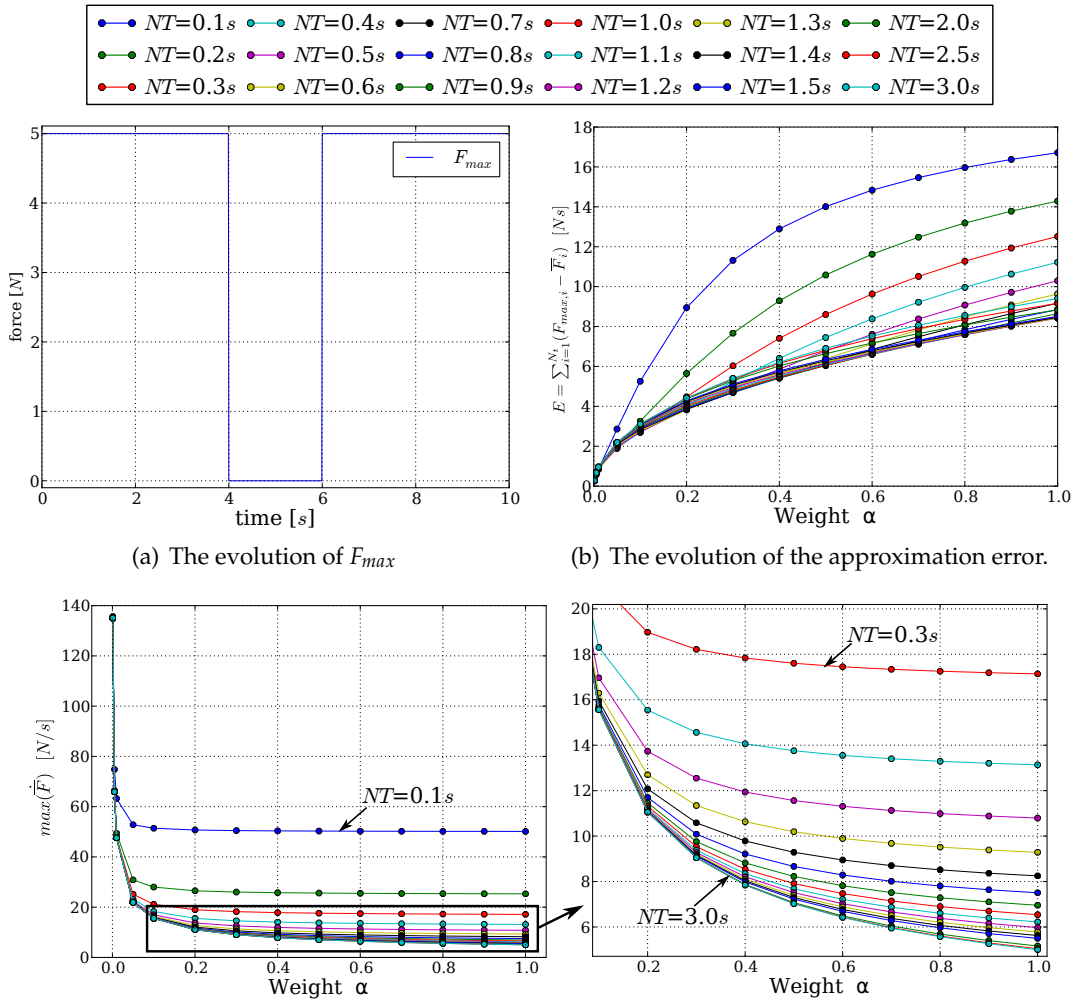


Figure 5.7: Evolution of $\overline{F}$, the maximum force derivative and the approximation error E using a set of different NTs with the weight $\alpha = 0.2$.

evolution of $\overline{F}$ in the term of maximum force derivative and the approximation error. Figure 5.8 shows the evolution of the maximum force derivative and the approximation error using a set of different αs and NTs with the same F_{max}. Combined with the results shown in 5.8 and the results discussed above, the rules for choosing the weight and the prediction are summarized as following:



(a) The evolution of F_{max}

(b) The evolution of the approximation error.

(c) The evolution of the maximum force derivative.

Figure 5.8: The evolution of the maximum force derivative and the approximation error E using a set of different αs and NTs with respect to the F_{max} with $5N$ instantaneous changes.

- The weight can be chosen between 0.05 and 0.8 according to the user's requirements. When the weight is less than 0.05, the maximum force derivative is too large. When the weight is larger than $0.8s$, the approximation error is too large because the MPC takes too much time to reach the reference, resulting a too conservative constraint.

- The weight determines the maximum prediction horizon that the MPC needs to react to the changes of references. With a certain weight, any longer prediction

horizon does not decrease the maximum force derivative efficiently, but increases the computation burden.

- The prediction horizon should not be less than $NT = 0.2s$, because short prediction horizon cannot enable the MPC to have enough time to react to the changes of the reference. In this case, the resulting maximum force derivative is still too large.

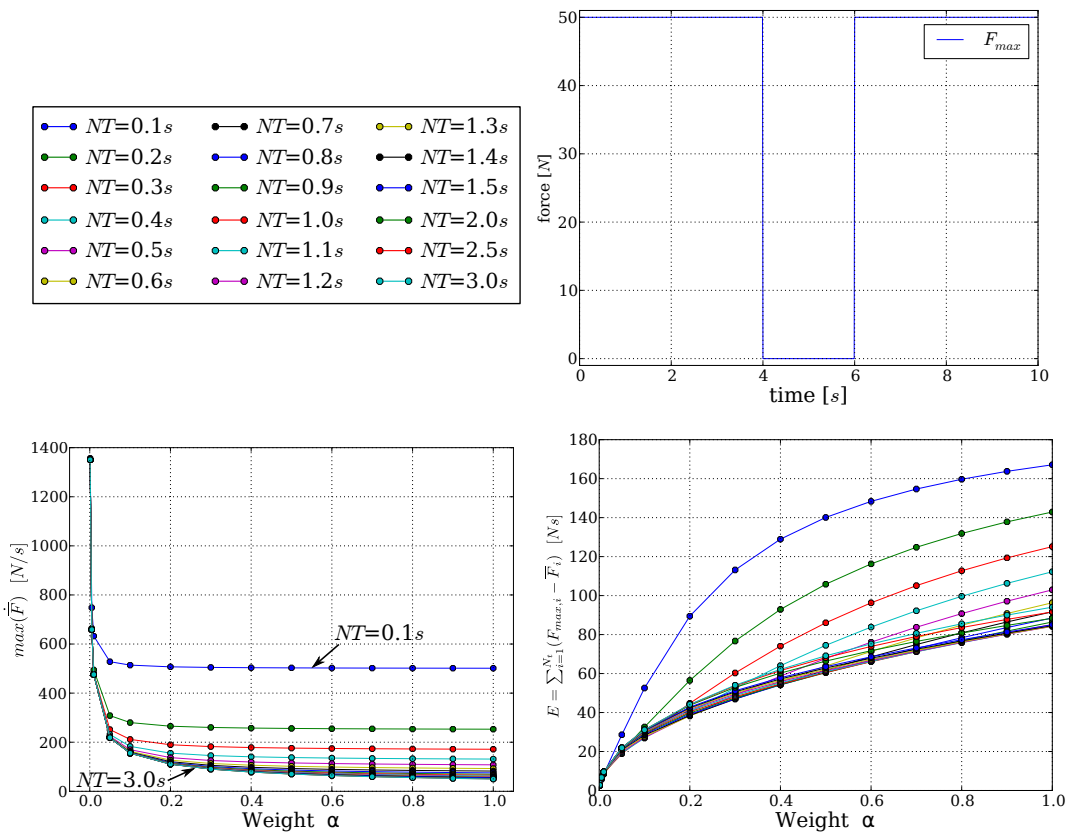


Figure 5.9: The evolution of the maximum force derivative and the approximation error E using a set of different αs and NTs with respect to the F_{max} with $50N$ instantaneous changes.

Sensitivity Analysis on The Value of F_{max}

Additionally, the instantaneous changes of F_{max} are inherent causes of large maximum force derivatives. The larger the instantaneous changes are, the larger the maximum force derivatives may be. To evaluate the effects of the changes of F_{max} on the maxi-

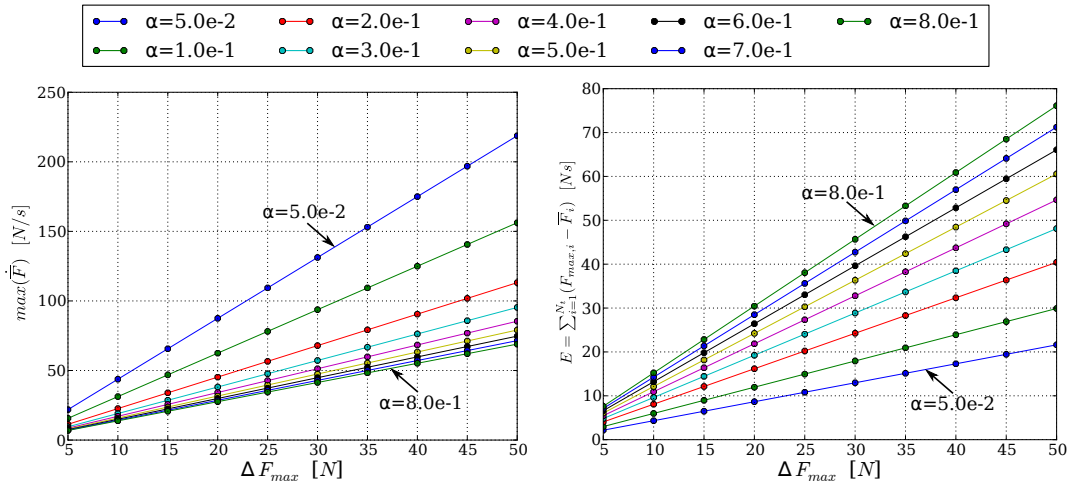


Figure 5.10: The evolution of the maximum force derivative and the approximation error with the instantaneous changes of F_{max}.

mum force derivative, the computation is carried out on the F_{max} with $50N$ instantaneous changes using a set of different NTs and αs. Figure 5.9 shows that the maximum force derivative does not decrease to the equivalent value by enlarging the prediction horizon and/or increasing the weight, compared with the F_{max} with $5N$ instantaneous changes. In fact, given the identical weight and prediction horizon, the maximum force derivative increases in proportion to the instantaneous changes of F_{max}. Figure 5.10 shows that the maximum force derivative increases linearly with the increase of instantaneous changes of F_{max}, given $\alpha = 0.4$ and $NT = 1.0s$ for example. This linear relationship is also true in other combinations of the weight and the prediction horizon. Therefore, the magnitude of the instantaneous changes of F_{max} does not affect the choice of the weight and the prediction. Threes rules can be applied to any instantaneous changes of F_{max}.

5.3 Control Framework

The overall control framework is shown in Figure 5.1. The proposed MPC, as an online approach, generates a continuous evolution $\overline{F}$ of the maximum allowable force F_{max} and a smooth evolution $\underline{F}$ of the minimum allowable force F_{min} as a high-level control. The results of the MPC is used by the QP reactive controller to minimize large instantaneous

changes in joint torques. To achieve this, the baseline QP reactive controller (2.10) is improved by implementing the continuous force constraints generated by the MPC in (5.10).

$$\boldsymbol{\tau}_t^* = \begin{cases} \arg\min\limits_{\mathbb{X}_t} & \|T(\mathbb{X}_t, t)\|_Q^2 + \|\mathbb{X}_t\|_R^2 \\ \text{subject to} & \mathbf{M}(\boldsymbol{q}_t)\dot{\boldsymbol{\nu}}_t + \boldsymbol{n}(\boldsymbol{q}_t, \boldsymbol{\nu}_t) = \mathbf{S}\boldsymbol{\tau}_t + \mathbf{J}_c(\boldsymbol{q}_t)^T \boldsymbol{F}_{c,t} \\ & \mathbf{J}_c(\boldsymbol{q}_t)\dot{\boldsymbol{\nu}}_t + \dot{\mathbf{J}}_c(\boldsymbol{q}_t, \boldsymbol{\nu}_t)\boldsymbol{\nu}_t = 0 \\ & C_t \boldsymbol{F}_{c,t} \leq 0 \\ & \overline{F}_t \leq F_{n,t} \leq \overline{F}_t \end{cases} . \tag{5.10}$$

5.4 Results

The proposed approach is applied to control a 7-DoF Kuka LWR robot and a 38-DoF iCub robot using the Arboris-Python simulator [Salini 12b], an open-source dynamic simulator written by Python. The Kuka robot and the iCub robot are both actuated by joint torques to perform tasks in operational space under contact constraints. The results of *the proposed approach* (5.10) are compared with those of *the baseline approach* (2.10).

5.4.1 Discontinuous evolution of the force constraint.

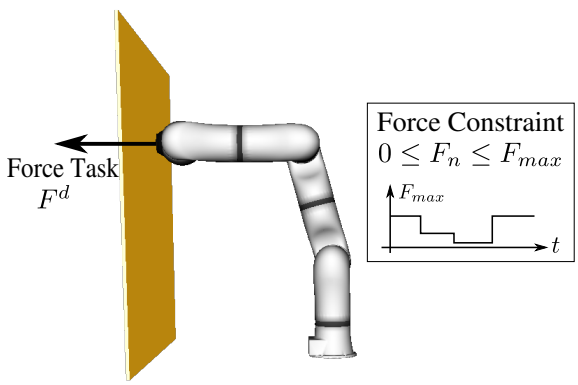
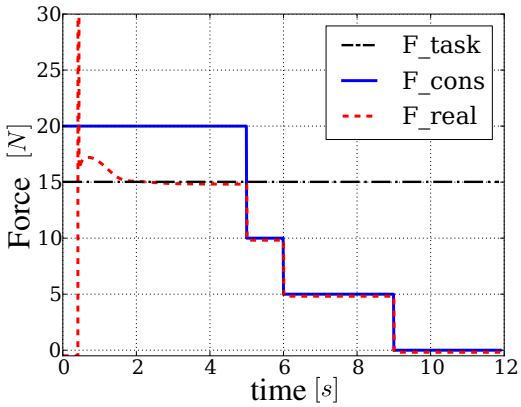
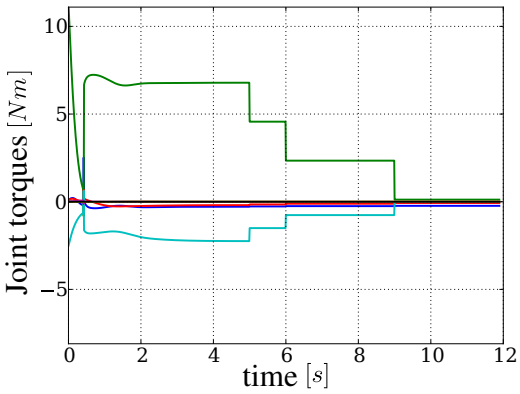


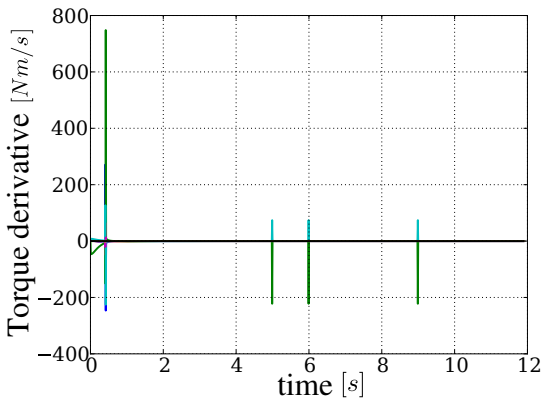
Figure 5.11: Scenario of Kuka performing a force task while satisfying discontinuous force constraint simultaneously.



(a) Evolution of the force.



(b) Evolution of joint torques.



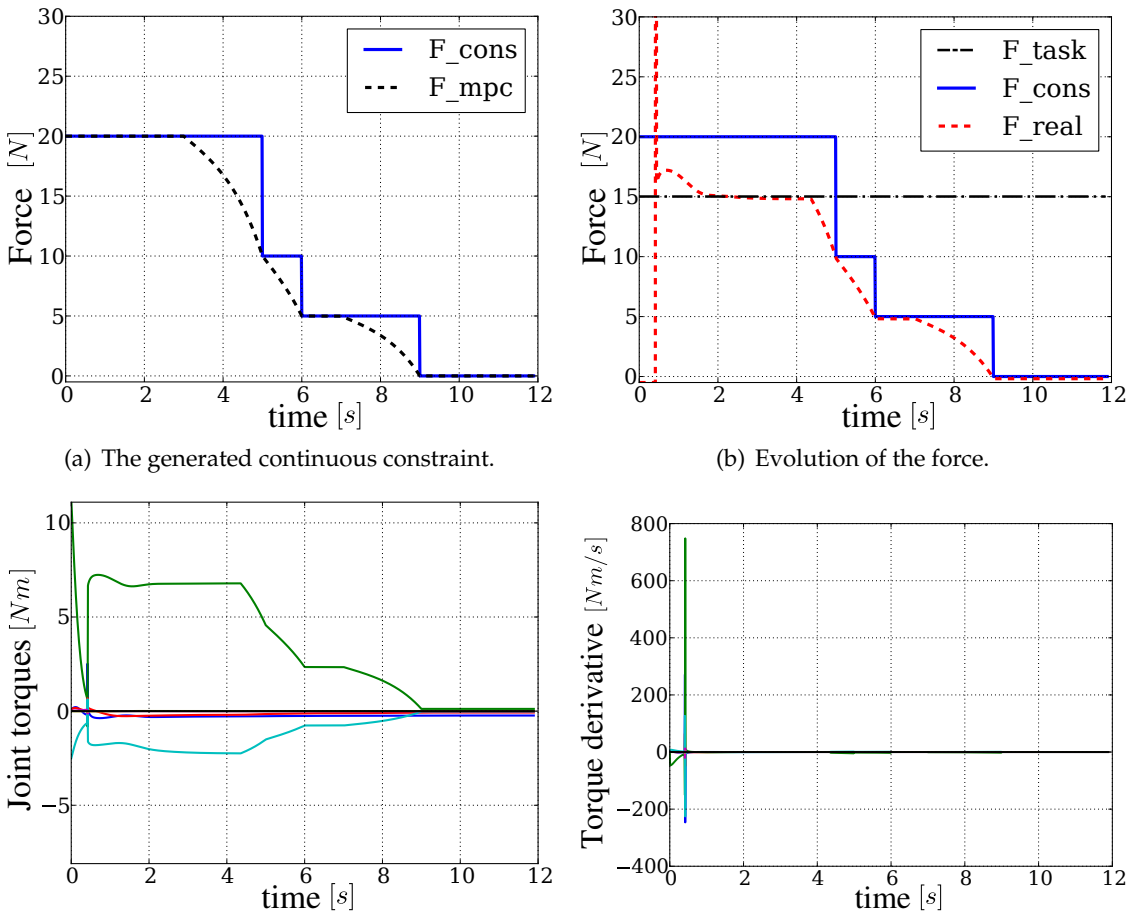
(c) Evolution of torque derivative.

Figure 5.12: Simulation results of the discontinuous force constraint *without* MPC smoothing. Large torque derivatives are clearly observed in (c).

In this experiment, an end-effector force task is defined to push against a fixed object with a constant force value. Meanwhile, the force constraint on the maximum allowed contact force evolves discontinuously. The results using the QP control framework without and with the proposed approach are compared.

In Figure 5.12, at the beginning the impact force results in a big peak on the force, which is not considered in this work. At $t = 5.0s$, the force constraint becomes suddenly active, causing torques discontinuities. At $t = 6.0s$ the force constraint changes discontinuously, and discontinuous torques and big torque derivative are observed. At $t = 9.0s$, the constraint decreases to zero suddenly, which means the contact constraint is removed, leading to discontinuities of torques and big torque derivatives.

In Figure 5.13(a), the new generated continuous force constraint using the proposed



(a) The generated continuous constraint.

(b) Evolution of the force.

(c) Evolution of joint torques.

(d) Evolution of torque derivative.

Figure 5.13: Simulation results of the discontinuous force constraint *with* the MPC approach. The large torque derivatives are significantly reduced in (d).

framework (5.10) is compared with original discontinuous force constraint. In Figure 5.13(b), it is shown that the new continuous force constraint is fully satisfied. With this continuous force constraint, the joint torques evolve smoothly and significant decreases of torque derivatives can be observed in 5.13(d).

5.4.2 Standing up from a chair

Figure 5.14 illustrates the 4 steps required for the robot to stand up from a chair:

1) The robot is sitting on the chair and the contact with the chair exists;

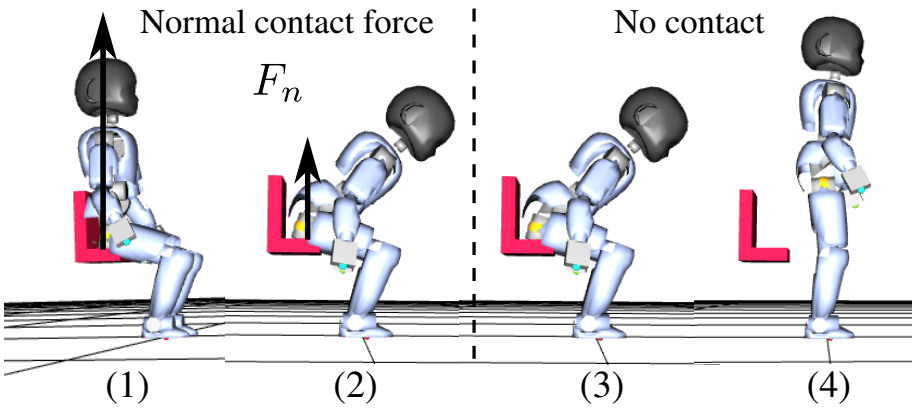


Figure 5.14: Snapshots of four steps as the robot stands up from a chair.

2) The robot moves its CoM into the support polygon of the feet in preparation for leaving the chair;

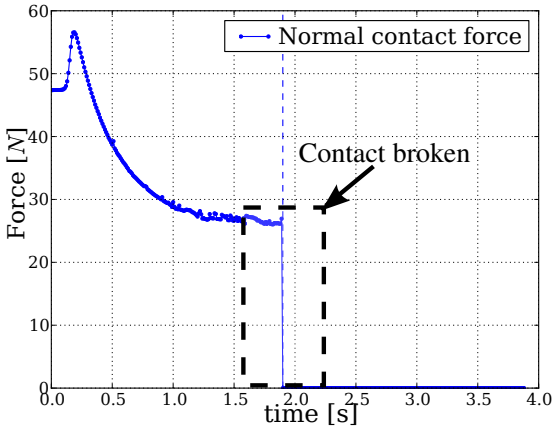
3) The contacts between the robot and the chair is broken;

4) The robot stands up.

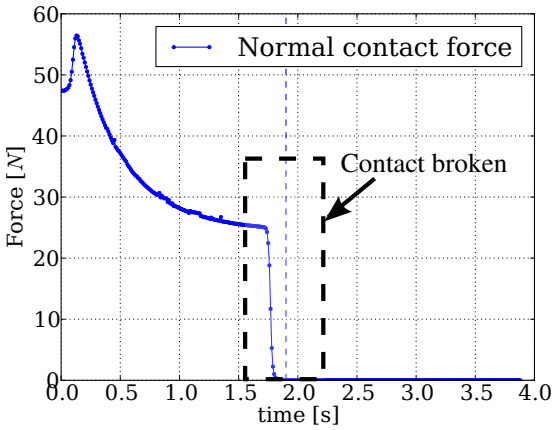
The resulting evolutions of the normal contact force between the robot and the chair using *the baseline approach*, *Salini's approach* [Salini 12a] and *the proposed approach* are shown in Figure 5.15(a), (b) and (c), respectively. The evolutions of the knee joint torque and its derivative are shown in Figure 5.16 for all three approaches.

Using *the baseline approach*, when the contact constraint between the chair and the robot is removed at $t = 1.9s$, the normal contact force instantaneously becomes zero (see Figure 5.15(a)). Among all of the joints in the iCub, the knee joint exhibits the largest instantaneous changes in joint torques and its torque derivative is $448Nm/s$ (see Figure 5.16(a) and (b)).

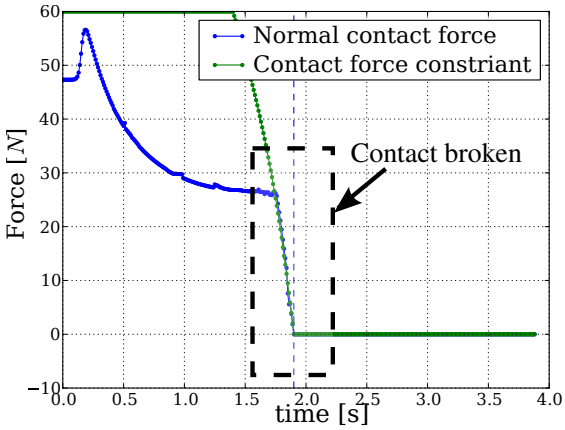
Using *Salini's approach*, the weight of the contact force task increases at $t = 1.4s$, which is set according to the scenario. The resulting contact force starts to decrease at $t = 1.7s$ and reaches zero at $t = 1.9$ when the contact is broken (see Figure 5.15(b)). However, Figure 5.16(d) shows the torque derivative reaches to $125N/s$ during the decrease of the contact force, because just increasing the weight of the contact force task cannot regulate the force derivatives. Moreover, when the contact is broken, joint torque decreases suddenly, resulting in $-74N/s$ torque derivative. This is because the contact force task is



(a) The normal contact force decreases to zero abruptly when the contact is broken.



(b) The normal contact force decreases to zero before the contact is broken



(c) The normal contact force decrease gradually to zero with respect to the maximum allowable contact force constraint.

Figure 5.15: Evolution of the normal contact force between the robot and the chair with *the baseline approach* (a), *Salini's approach* (b) and *the proposed approach* (c).

abruptly removed with a larger weight, resulting a discontinuous task transition.

Using *the proposed approach*, the evolution of the normal contact force is shown in 5.15(c) with the time horizon $NT = 0.5s$ and a normal contact force of $F_{max} = 60N$. Once the break of the contact is previewed by the MPC at $0.5s$ before it occurs, the MPC starts to decrease the contact force towards zero. The change of the contact force is significantly reduced at $t = 1.9s$ compared with the result using *the baseline approach* and *Salini's approach*, when the contact is broken. Furthermore, the torque of the knee joint evolves with less changes and its torque derivative is only $61Nm/s$ (see Figure 5.16(f)).

5.4.3 Lifting and putting down the foot

In this scenario, the robot lifts its right foot off the ground at $t = 2.0s$ and then puts it back down at $t = 5.0s$. The action includes five steps as shown in Figure 5.17:

1) The robot is standing on the ground;

2) The robot moves its CoM above the left foot to ensure that it doesn't fall down after the contact of the right foot is broken;

3) The contact of the right foot is broken and the robot lifts the right foot;

4) The robot puts down the right foot and the contact is re-established;

5) The robot goes back to initial posture.

Figure 5.18 shows the evolutions of the normal contact forces on both feet using *the baseline approach* and *the proposed approach*. Figure 5.19 shows the resulting evolutions of the knee joint torque and its derivative for both approaches.

Using *the baseline approach*, Figure 5.18(a) shows that the normal contact force abruptly changes at $t = 2.0s$ and $t = 5.0s$ when the right foot contact is broken and established, respectively. Once the right foot lifts off the ground, the whole-body weight shifts from the double feet support to one foot support. The sudden decrease of the normal contact force on the right foot directly leads to the increase of the normal contact force on the left foot. These sudden changes of contact forces result in the discontinuities in joint torques. The hip joint exhibits the largest changes in joint torques among all of the joints and its torque derivative is up to $920Nm/s$ (shown in Figure 5.19(a) and (c)).

Using *the proposed approach* with time horizon $NT = 0.5s$, the normal contact force of the right foot gradually decreases to zero before the contact is broken and smoothly increases after the contact is added (see Figure 5.18(b)). As a result, large changes in the contact force are avoided, and as shown in Figures 5.19(b) and (d), no large changes in hip torques occur. The torque derivative is significantly reduced from $920Nm/s$ using *the baseline approach* to $85Nm/s$ using *the proposed approach*.

In Figure 5.18, it is observed that a spike appears around $t = 5.0s$. This spike is the impact force due to the contact between two rigid objects. Reducing this impact force

peak would require to locally adapt the apparent impedance of the foot making contact [Walker 90, Pagilla 01]. This problem is not addressed by the proposed approach.

5.4.4 Walking

In order to verify the effectiveness of the proposed approach in a more dynamic situation, the proposed approach is extended to the scenario of walking. In this scenario, the desired walking pattern can be computed by ZMP planning [Kajita 03]. In this simulation, the robot spends one second to move one step and it has $0.3s$ for double support. The maximum allowable force of F_{max} is $300N$.

Figures 5.20(a) and (b) show the evolutions of the normal contact force on both feet using *the baseline approach* and *the proposed approach*, respectively. Figures 5.20(c) and (d) show the resulting evolutions of the hip joint torque and its derivative for both approaches, respectively.

Figure 5.20(b) shows that the changes of the normal contact forces are slightly reduced compared to *the baseline approach* shown in Figure 5.20(a). Among all of the joints in the iCub, the hip roll joint exhibits the largest changes in torques and large torque derivatives using *the baseline approach* when the contact is broken or established (see in Figure 5.20(c)). However, using *the proposed approach* the changes in joint torques and torque derivatives are not obviously reduced as shown in Figure 5.20(d).

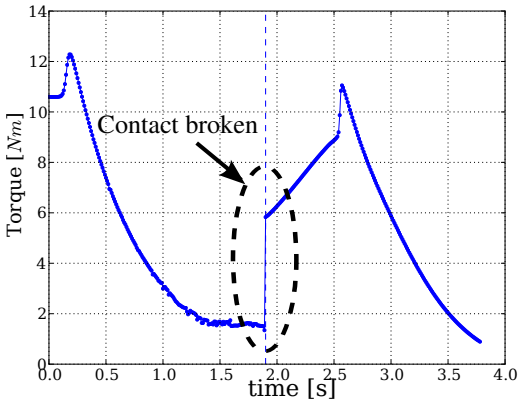
In dynamic walking, the whole-body weight shifts quickly between double feet support and one foot support. A large maximum allowable contact force F_{max} is required to support the heavy robot with one foot. Moreover, dynamic walking requires fast contact modifications. Each foot has a short time in contact with the ground between lifting and lowering phases. Due to these reasons, the ability of the proposed MPC to minimize the changes of contact forces is restricted. Indeed, the proposed scheme must decrease a large maximum force to zero in a short time, resulting in a large change of the contact force at the moment when the contact is broken. Additionally, the constraint imposed to F_{max} using MPC does not account for the fact that the position of the CoM has a large influence on contact forces. To summarize, the effects of the proposed approach are limited in very dynamic situations requiring a large variation of the force in a short time.

Nevertheless, the proposed approach has three main advantages. First, prediction allows smooth joint torques to be generated in situations where purely reactive approaches would fail to do so. Although global planning provides such a smoothing feature, prediction over a receding finite horizon can be seen as "short term reactive" and is thus better suited for dynamically changing situations. Second, the concept can be generalized to any constraint, the evolution of which can be known *a priori* (e.g. obstacle avoidance). Finally, the proposed method acts only on the constraints and can be directly used within any control scheme that handles constraints as inequalities.

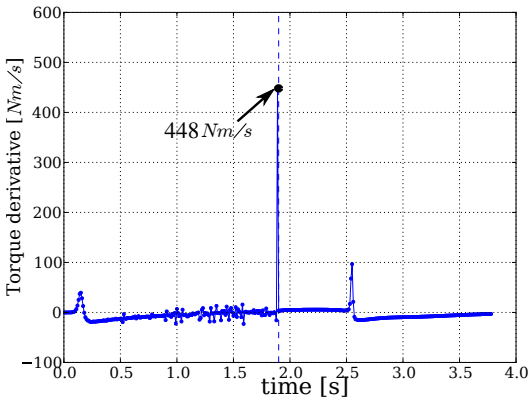
5.5 Conclusion

In this chapter, the proposed MPC approach is used to minimize the rate of change in joint torques. The contribution here is to endow a reactive control scheme, which handles constraints as inequalities, with the ability to anticipate and adapt more robustly to changes of contacts.

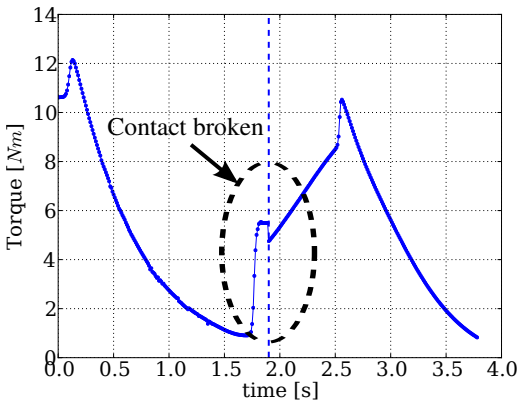
The proposed MPC can preview changes in contacts over a receding finite horizon and generate a continuous contact force constraint, the changes of which are minimized. The QP reactive controller uses this MPC generated constraint to reduce the changes of the actual contact forces. As a result, the rate of change in joint torques is minimized. Simulations involving breaking and establishing contacts show that the proposed approach can successfully minimize instantaneous changes in joint torques. The results in walking situations also show that the proposed approach has limitations related to the dynamics of the motions to be performed.



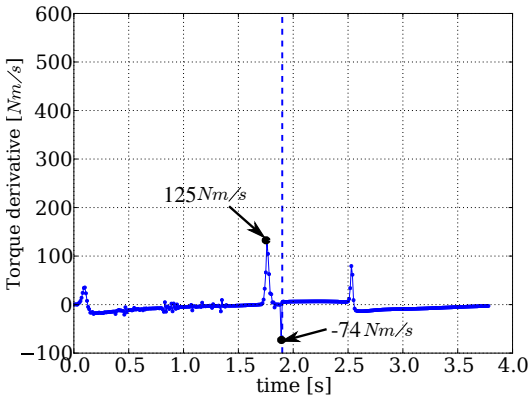
(a) A torque peak appears when the contact is broken.



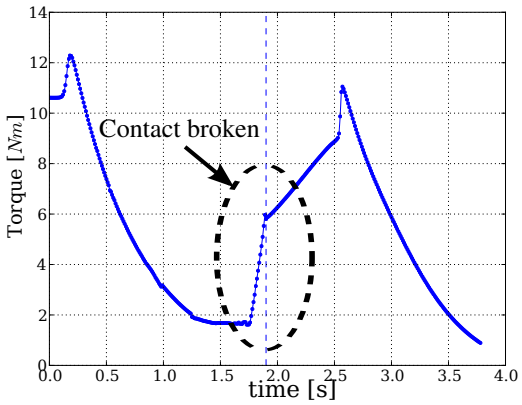
(b) The torque derivative reaches a maximum of $448 Nm/s$ when the contact is broken.



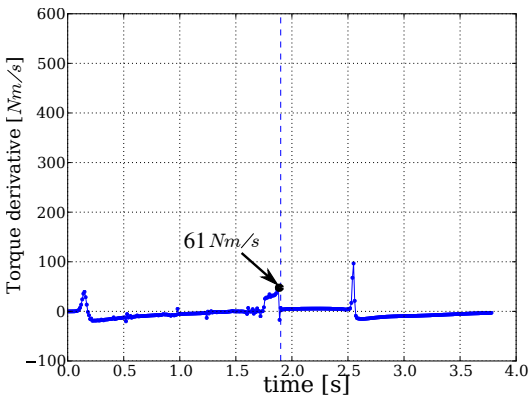
(c) The joint torque undergoes changes when the force task is deactivated.



(d) The torque derivative is $-74 Nm/s$ when the contact is broken. But the maximum $125 Nm/s$ occurs before the contact is broken



(e) The joint torque increases as expected in a smooth way.



(f) The torque derivative is only about $61 Nm/s$.

Figure 5.16: Evolution of the knee joint torque and its torque derivative with *the baseline approach* (top) *Salini's approach* (middle) and *the proposed approach* (bottom).

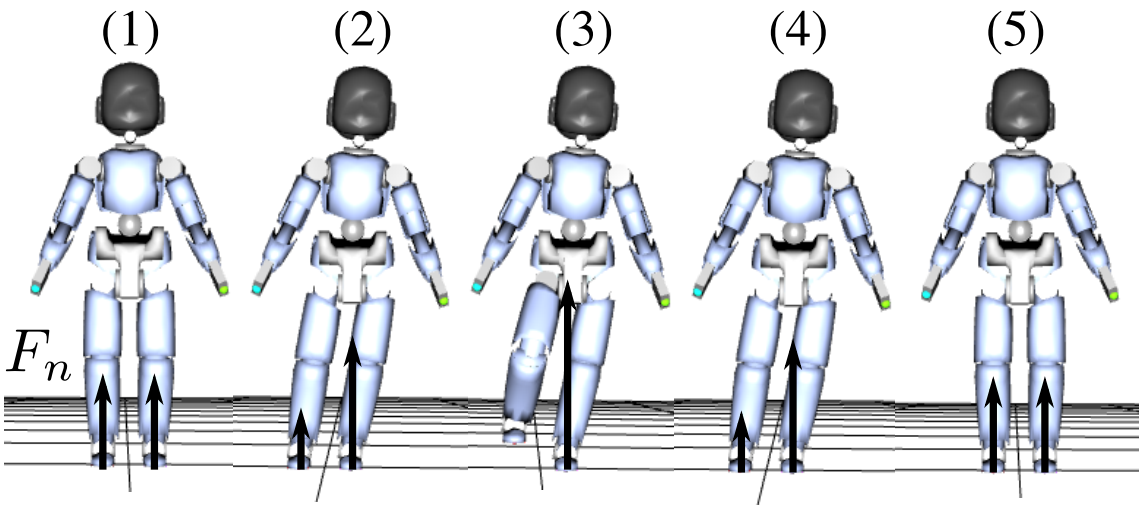
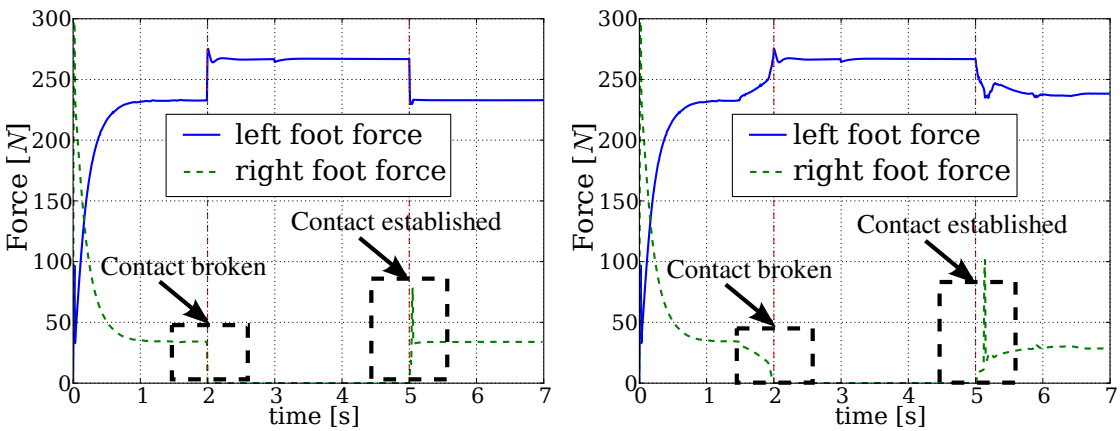


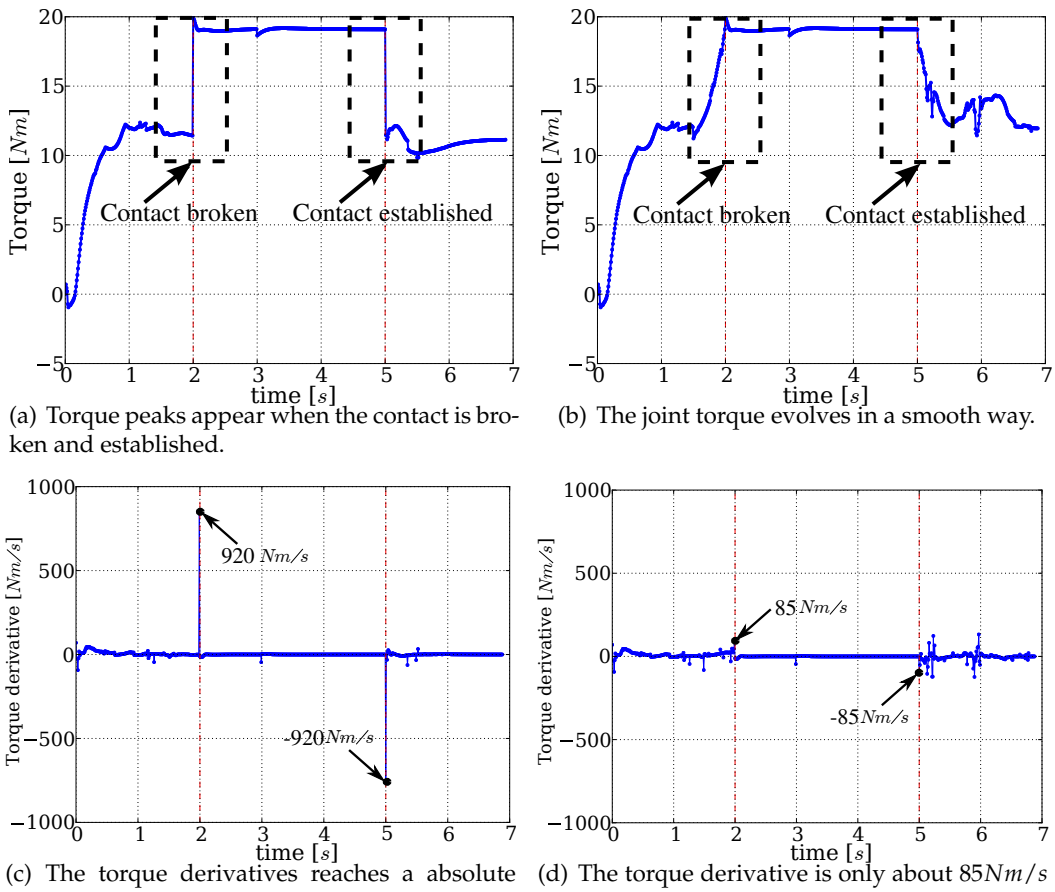
Figure 5.17: Snapshots of lifting and putting down the right foot with 5 steps.



(a) Large changes occur on the contact force of feet when the contact is broken and established.

(b) No large changes occur on the contact force of feet when the contact is broken and established.

Figure 5.18: Evolution of normal contact force on feet with *the baseline approach* (a) and *the proposed approach* (b).



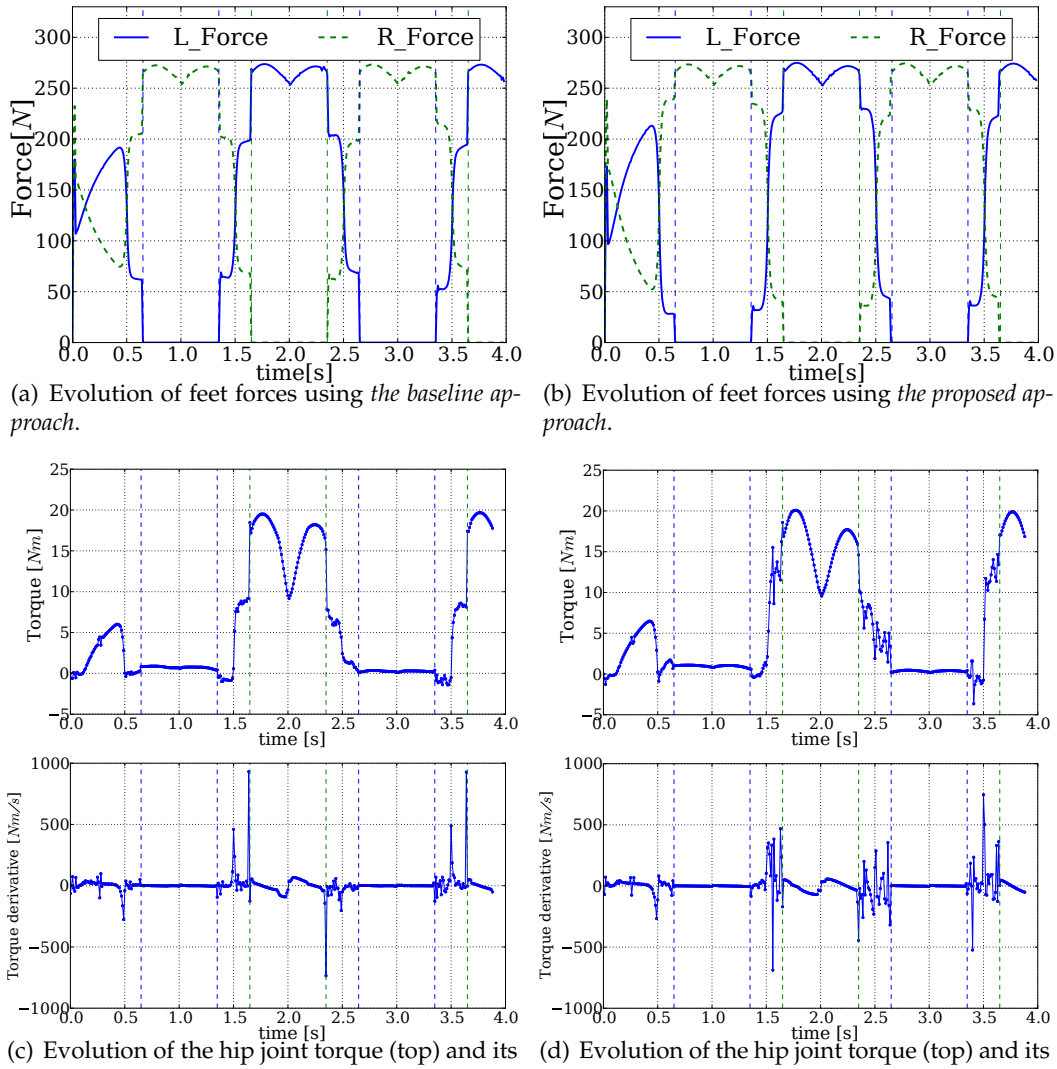
(a) Torque peaks appear when the contact is broken and established.

(b) The joint torque evolves in a smooth way.

(c) The torque derivatives reaches a absolute value maximum of $920Nm/s$ when the contact is broken and established.

(d) The torque derivative is only about $85Nm/s$ and $-85Nm/s$.

Figure 5.19: Evolution of hip joint torque and its torque derivative with *the baseline approach* (left) and *the proposed approach* (right).



(a) Evolution of feet forces using *the baseline approach*.

(b) Evolution of feet forces using *the proposed approach*.

(c) Evolution of the hip joint torque (top) and its derivative (bottom) using *the baseline approach*.

(d) Evolution of the hip joint torque (top) and its derivative (bottom) using *the proposed approach*.

Figure 5.20: Evolution of hip joint torque and its torque derivative with *the baseline approach* (left) and *the proposed approach* (right).

Chapter 6
Conclusions and Perspectives

In this chapter, the thesis is concluded with a summary of the contributions in Section 6.1 and an outlook on the future directions for this study in section 6.2.

6.1 Conclusion

Robotic systems with sophisticated mechanical structures are expected to perform complex activities in dynamic environments, especially in human environments. Although there are many advances in these system, controlling them is still very challenging. First of all, multiple task objectives have to be accounted for simultaneously in the controller. Secondly, a set of intrinsic and extrinsic constraints must be satisfied simultaneously. Last but not least, smooth actuator inputs are essential and necessary to avoid unpredictable and unstable behaviours of the system. However, changes in the environment, disturbances and evolutions of the objectives may result in large instantaneous changes in actuator inputs when solving the control problem. In this thesis, a new QP reactive controller called Generalized Hierarchical Control (GHC) is proposed to handle large instantaneous changes in actuator inputs due to transitions among multiple prioritized tasks, and a predictive control primitive based on Model Predictive Control (MPC) is developed to deal with large instantaneous change in actuator inputs as a result of discontinuous evolutions of contact constraints and obstacle avoidance constraints.

This thesis also proposes a novel and unifying Generalized Hierarchical Control approach for handling multiple tasks with both strict and nonstrict hierarchies. A generalized projector is developed. It can precisely regulate how much a task can influence or be influenced by other tasks through the modulation of a priority matrix: a task can be

completely, partially, or not at all projected into the null-space of other tasks. Continuous task transitions among a set of prioritized tasks can be achieved using this generalized projector and, using the same mechanism, tasks can be easily and smoothly inserted or removed. Moreover, the GHC approach can maintain and switch task priorities while satisfying a set of equality and inequality constraints. Several experiments are conducted to demonstrate that GHC allows tasks to be inserted and deleted smoothly, continuous task transitions, and the handling of strict and nonstrict task hierarchies subject to constraints. These experiments emphasize several characteristics of the GHC approach:

- Priorities among tasks can be maintained by applying the generalized projectors. Through the modulation of the priority matrices and consequently of the associated generalized projectors, GHC can behave as a controller that takes into account a strict hierarchy and as a controller that uses weighting strategy. In other words, the controller can be configured to control simultaneously tasks assigned with strict hierarchies, as well as tasks with different weights (nonstrict priorities).

- Continuous and simultaneous rearrangements of multiple task priorities can be achieved easily by the continuous variations of relevant entries in the generalized projectors associated to these tasks. The rate of change of resulting joint torques are significantly minimized.

- The GHC approach is not restricted at the dynamic level here. In fact, it can also be used in other types of controllers, such as a velocity kinematics controller. The basic idea is to associate each task with a task variable in joint space ($\dot{\boldsymbol{q}}_i'$, $\ddot{\boldsymbol{q}}_i'$, $\boldsymbol{\tau}_i'$, etc.), then to apply generalized projectors to these task variables, and finally the global joint space variable is the sum of each projected task variables ($\mathbf{P}_i\dot{\boldsymbol{q}}_i'$, $\mathbf{P}_i\ddot{\boldsymbol{q}}_i'$, $\mathbf{P}_i\boldsymbol{\tau}_i'$, etc.)

This thesis proposes a predictive control primitive based on Model Predictive Control (MPC), which can be integrated into a QP reactive controller, for example GHC, to minimize the rate of change in joint torques due to discontinuous evolutions of constraints, particularly for contact constraints and obstacle avoidance constraints. MPC is employed to preview changes in constraints over a finite horizon and generate a smooth constraint, the changes of which are minimized. The QP reactive controller thus uses this MPC gen-

erated continuous constraint to reduce the large instantaneous changes in joint torques compared with the original discontinuous constraints. According to the formulations of the contact constraint and the obstacle avoidance constraint, two different MPC continuous constraint generations are developed. For obstacle avoidance constraints, MPC takes advantage of the reaction time to preview the movements of the obstacle based on sensed position of the obstacle, generating virtual continuous position, velocity and acceleration of the obstacle. As a result, a continuous obstacle avoidance constraint is obtained and used in the QP reactive controller to minimize the large instantaneous changes in joint torques when the obstacle moves suddenly and rapidly. For contact constraints, MPC produces a time-varying maximum and minimum allowable contact force constraint to minimize the large instantaneous changes in joint torques when the robot establishes or breaks a contact with the environment. Several experiments are conducted to illustrate that the proposed control framework based on MPC can effectively and significantly reduce instantaneous changes in joint torques when contacts are suddenly established or broken, and obstacles moves abruptly and rapidly. In short, the predictive control primitive has several advantages as follows:

- One advantage of using MPC for handling discontinuous constraints here is to be able to explicitly account for complex bounds on these constraints themselves in an automatic and generic way. This can ensure that the generated continuous constraints are always compatible with the original constraints.

- The MPC scheme for constraints can preview changes in constraints over a future time horizon, react to the changes in advance and thus generate continuous evolutions of constraints. The contribution here is to endow a reactive controller, which handle constraints as inequalities, with the ability to anticipate and adapt more robustly and effectively to changes of constraints.

- One key feature of the proposed control framework is that the constraints of the reactive controller are modified rather than the task objectives. As a result, the proposed approach does not directly modify the task trajectory but ensure that the worst case torque changes are minimized. As such, it is a very generic approach

which can be applied independently from the way the control law and objective function is formulated.

6.2 Perspectives

In the future, some interesting points should be investigated to extend and improve the capabilities of the proposed approached in this thesis.

6.2.1 Toward reduction of the computation cost of GHC

The computational cost of the current GHC strategy is sensitive to the number of DoF of the robot and the number of tasks. For a fixed-based Kuka LWR robot with 7 DoFs performing n_1 motion tasks of different priority levels, a set of intermediate joint acceleration variables $\ddot{\boldsymbol{q}}_i' \in \mathbb{R}^{7n_1}$ and the joint torques $\boldsymbol{\tau} \in \mathbb{R}^7$ needs to be computed at each time step from a QP solver. For a floating-based humanoid robot iCub with 32 DoF performing n_2 tasks, the number of optimal variables would be $32(n_2 + 1)$. The enormous increase of the number of optimal variables brings the extra computational burden for the QP solver. Therefore, in order to achieve real-time control for a complex robot with a high number of DoF performing multiple tasks, immediate future work includes two aspects of work to reduce the computation cost of GHC. One possible way is to reduce the number of intermediate joint acceleration variables. Some joint acceleration variables of lower priority tasks do not need to be computed by analysing the generalized projector at each time step.

6.2.2 Toward intelligent task transitions

Currently, the task priorities are manually changed with respect to the goal objectives. The task transitions are achieved by the preplanned evolution of task priorities. However, this indeed limits the adaptability and robustness of the robot performance in a complex and dynamic environment. Therefore, one important future direction is to achieve intelligent task transitions to perform complex activities in a dynamic environment. Intelligent

task transitions mean to automatically adjust task priorities to achieve a long-term goal objectives. To improve the task transitions of GHC, one interesting approach presented in [Salini 12a] is to integrate a decision making engine based on fuzzy logic as a high-level controller to change task weights automatically. Another possible direction is to endow the controller with the ability to automatically realize task transitions in a predictive way, according to the anticipated future task objectives of the robot. The most interesting point would be the use of robot learning techniques to incrementally learn and improve the tuning of the relative influence of each task with respect to others before and during transitions [Lober 15].

6.2.3 Toward a generic predictive framework

In short-term perspective, high-level predictive formulation and parameterization can be further developed to reach faster control rate, and provide significant improvements in the overall control performance. Such improvements are envisaged in the implementation on the iCub robot. A long-term objective for the development of the proposed MPC scheme is to reach a complete and generic predictive framework for robotic systems stably and robustly performing complex activities in a dynamic environment. The control framework proposed in this work is not only restricted to handle constraints. It could be extended to the overall behaviours of the robot, including task performance, balance, walking, etc. The MPC scheme could be employed to parallelize the multi-objective problem and be able to make decisions for the robot in order to autonomously accomplish tasks [Ibanez 14]. One interesting point would also be to use a predictive approach for online trajectory planning [Bellingham 03, Makarov 11]. As stated in Section 6.2.2, the predictive approach can also be used to preview the future goal objectives and achieve task transitions automatically correspondingly. The most interesting direction is probably to find the possibility of directly integrating the proposed MPC scheme for contact forces as a part of locomotion generation algorithms for humanoids.

Bibliography

[Abe 07] **Yeuhi Abe**, **Marco da Silva et Jovan Popović**. *Multiobjective control with frictional contacts*. In Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation, pages 249–258, 2007.

[Aghili 05] **Farhad Aghili**. *A unified approach for inverse and direct dynamics of constrained multibody systems based on linear projection operator: applications to control and simulation*. IEEE Transactions on Robotics, vol. 21, n° 5, pages 834–849, 2005.

[Akella 94] **Prasad Akella**, **Vicente Parra-Vega**, **Suguru Arimoto et Kazuo Tanie**. *Discontinuous model-based adaptive control for robots executing free and constrained tasks*. In Proceedings of the International Conference on Robotics and Automation, pages 3000–3007, 1994.

[Altmann 05] **Simon L Altmann**. Rotations, quaternions, and double groups. Courier Corporation, 2005.

[Andersson 89] **Russell L Andersson**. *Aggressive trajectory generator for a robot ping-pong player*. Control Systems Magazine, vol. 9, n° 2, pages 15–21, 1989.

[Andersson 12] **Joel Andersson**, **Johan Åkesson et Moritz Diehl**. *CasADi: a symbolic package for automatic differentiation and optimal control*. In Recent Advances in Algorithmic Differentiation, pages 297–307. Springer, 2012.

[Baerlocher 98] **P. Baerlocher et R. Boulic**. *Task-priority formulations for the kinematic control of highly redundant articulated structures*. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, volume 1, pages 323–329 vol.1, 1998.

[Baerlocher 04] **Paolo Baerlocher et Ronan Boulic**. *An inverse kinematics architecture enforcing an arbitrary number of strict priority levels*. The visual computer, vol. 20, n° 6, pages 402–417, 2004.

[Bellingham 03] **John Bellingham**, **Yoshiaki Kuwata et Jonathan How**. *Stable receding horizon trajectory control for complex environments*. In Proceedings of the AIAA Guidance, Navigation, and Control Conference, 2003.

[Bobrow 85] **James E Bobrow**, **Steven Dubowsky et JS Gibson**. *Time-optimal control of robotic manipulators along specified paths*. The international journal of robotics research, vol. 4, n° 3, pages 3–17, 1985.

[Boyd 04] **Stephen Boyd et Lieven Vandenberghe**. Convex optimization. Cambridge university press, 2004.

[Brock 02] **Oliver Brock**, **Oussama Khatib et Sriram Viji**. *Task-consistent obstacle avoidance and motion behavior for mobile manipulation*. In Proceedings of the International Conference on Robotics and Automation, volume 1, pages 388–393, 2002.

[Brufau 05] **Jordi Brufau**, **Manel Puig-Vidal**, **Jaime Lopez-Sanchez**, **Josep Samitier**, **N Snis**, **U Simu**, **S Johansson**, **W Driesen**, **J-M Breguet**, **J Gao et others**. *MICRON: Small autonomous robot for cell manipulation applications*. In Proceedings of the International Conference on Robotics and Automation, pages 844–849, 2005.

[Camacho 13] **Eduardo F Camacho et Carlos Bordons Alba**. Model predictive control. Springer Science & Business Media, 2013.

[Chan 95] **Tan Fung Chan et Rajiv V Dubey**. *A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators*. IEEE transactions on Robotics and Automation, vol. 11, n° 2, pages 286–292, 1995.

[Chaumette 01] **François Chaumette et Éric Marchand**. *A redundancy-based iterative approach for avoiding joint limits: Application to visual servoing*. IEEE Transactions on Robotics and Automation, vol. 17, n° 5, pages 719–730, 2001.

[Cheney 09] **Ward Cheney et David Kincaid**. *Linear algebra: Theory and applications*. The Australian Mathematical Society, page 544, 2009.

[Chiaverini 99] **Stefano Chiaverini**, **Bruno Siciliano et Luigi Villani**. *A survey of robot interaction control schemes with experimental comparison*. IEEE/ASME Transactions On Mechatronics, vol. 4, n° 3, pages 273–285, 1999.

[Collette 07] **C. Collette**, **A. Micaelli**, **C. Andriot et P. Lemerle**. *Dynamic balance control of humanoids for multiple grasps and non coplanar frictional contacts*. In Proceedings of the IEEE/RAS International Conference on Humanoid Robots, pages 81–88, 2007.

[Decré 09] **Wilm Decré**, **Ruben Smits**, **Herman Bruyninckx et Joris De Schutter**. *Extending iTaSC to support inequality constraints and non-instantaneous task specification*. In Proceedings of the International Conference on Robotics and Automation, pages 964–971, 2009.

[Dietrich 12a] **A. Dietrich**, **A. Albu-Schaffer et G. Hirzinger**. *On continuous null space projections for torque-based, hierarchical, multi-objective manipulation*. In Proceedings of the International Conference on Robotics and Automation, pages 2978–2985, May 2012.

[Dietrich 12b] **Alexander Dietrich**, **Thomas Wimbock**, **Alin Albu-Schaffer et Gerd Hirzinger**. *Integration of reactive, torque-based self-collision*

avoidance into a task hierarchy. IEEE Transactions on Robotics, vol. 28, n° 6, pages 1278–1293, 2012.

[Dietrich 15] **Alexander Dietrich**, **Christian Ott et Alin Albu-Schäffer**. *An overview of null space projections for redundant, torque-controlled robots*. The International Journal of Robotics Research, page 0278364914566516, 2015.

[Escande 14] **Adrien Escande**, **Nicolas Mansard et Pierre-Brice Wieber**. *Hierarchical quadratic programming: Fast online humanoid-robot motion generation*. The International Journal of Robotics Research, page 0278364914521306, 2014.

[Faverjon 87] **Bernard Faverjon et Pierre Tournassoud**. *A local based approach for path planning of manipulators with a high number of degrees of freedom*. In Proceedings of the International Conference on Robotics and Automation, volume 4, pages 1152–1159, 1987.

[Flacco 14] **Fabrizio Flacco et Alessandro De Luca**. *A reverse priority approach to multi-task control of redundant robots*. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 2421–2427, 2014.

[Franklin 94] **Gene F Franklin**, **J David Powell et Abbas Emami-Naeini**. *Feedback control of dynamics systems*. Addison-Wesley, Reading, MA, 1994.

[Gasparetto 12] **Alessandro Gasparetto**, **Paolo Boscariol**, **Albano Lanzutti et Renato Vidoni**. *Trajectory planning in robotics*. Mathematics in Computer Science, vol. 6, n° 3, pages 269–279, 2012.

[Golub 12] **Gene H Golub et Charles F Van Loan**. Matrix computations, volume 3. JHU Press, 2012.

[González-Banos 06] **Héctor H González-Banos**, **David Hsu et Jean-Claude Latombe**. *Motion planning: Recent developments*. Autonomous Mobile Robots: Sensing, Control, Decision-Making and Applications, 2006.

[Gouzenes 84] **Laurent Gouzenes**. *Strategies for solving collision-free trajectories problems for mobile and manipulator robots*. The International Journal of Robotics Research, vol. 3, n° 4, pages 51–65, 1984.

[Herrera-Aguilar 06] **Ignacio Herrera-Aguilar et Daniel Sidobre**. *Soft motion trajectory planning and control for service manipulator robot*. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006.

[Herzog 14] **Alexander Herzog**, **Ludovic Righetti**, **Felix Grimminger**, **Peter Pastor et Stefan Schaal**. *Balancing experiments on a torque-controlled humanoid with hierarchical inverse dynamics*. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 981–988, 2014.

[Hogan 85] **Neville Hogan**. *Impedance control: An approach to manipulation: Part ITheory; Part IIImplementation; Part IIIAplications*. Journal of dynamic systems, measurement, and control, vol. 107, n° 1, pages 1–24, 1985.

[Hsu 02] **David Hsu**, **Robert Kindel**, **Jean-Claude Latombe et Stephen Rock**. *Randomized kinodynamic motion planning with moving obstacles*. The International Journal of Robotics Research, vol. 21, n° 3, pages 233–255, 2002.

[Ibanez 14] **Aurelien Ibanez**, **Philippe Bidaud et Vincent Padois**. *A distributed model predictive control approach for robust postural stability of a humanoid robot*. In Proceedings of the International Conference on Robotics and Automation, pages 202–209, 2014.

[Jarquin 13] **Gerardo Jarquin**, **Adrien Escande**, **Gustavo Arechavaleta**, **Thomas Moulard**, **Eiichi Yoshida et Vicente Parra-Vega**. *Real-time smooth task transitions for hierarchical inverse kinematics*. In Proceedings of the IEEE/RAS International Conference on Humanoid Robots, pages 528–533, 2013.

[Kajita 03] **Shuuji Kajita**, **Fumio Kanehiro**, **Kenji Kaneko**, **Kiyoshi Fujiwara**, **Kensuke Harada**, **Kazuhito Yokoi et Hirohisa Hirukawa**. *Biped walking pattern generation by using preview control of zero-moment point*. In Proceedings of the International Conference on Robotics and Automation, volume 2, pages 1620–1626, 2003.

[Kanehiro 08] **Fumio Kanehiro**, **Florent Lamiraux**, **Oussama Kanoun**, **Eiichi Yoshida et Jean-Paul Laumond**. *A local collision avoidance method for non-strictly convex polyhedra*. Proceedings of robotics: science and systems IV, 2008.

[Kanoun 11] **Oussama Kanoun**, **Florent Lamiraux et Pierre-Brice Wieber**. *Kinematic Control of Redundant Manipulators: Generalizing the Task-Priority Framework to Inequality Task*. IEEE Transactions on Robotics, vol. 27, n° 4, pages 785–792, 2011.

[Kazerooni 90] **H Kazerooni**, **BJ Waibel et S Kim**. *On the stability of robot compliant motion control: Theory and experiments*. Journal of Dynamic Systems, Measurement, and Control, vol. 112, n° 3, pages 417–426, 1990.

[Keith 11] **F. Keith**, **N. Wieber P.-B.and Mansard et A. Kheddar**. *Analysis of the discontinuities in prioritized tasks-space control under discreet task scheduling operations*. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 3887–3892, 2011.

[Khatib 86] **Oussama Khatib**. *Real-time obstacle avoidance for manipulators and mobile robots*. The international journal of robotics research, vol. 5, n° 1, pages 90–98, 1986.

[Khatib 87] **Oussama Khatib**. *A unified approach for motion and force control of robot manipulators: The operational space formulation*. IEEE Journal of Robotics and Automation, vol. 3, n° 1, pages 43–53, 1987.

[Khatib 08] **Oussama Khatib**, **Luis Sentis et Jae-Heung Park**. *A unified framework for whole-body humanoid robot control with multiple constraints and contacts*. In European Robotics Symposium 2008, pages 303–312. Springer, 2008.

[Klein 90] **Charles Klein et Sankon Kittivatcharapong**. *Optimal force distribution for the legs of a walking machine with friction cone constraints*. IEEE Transactions on Robotics and Automation, vol. 6, n° 1, pages 73–85, 1990.

[Künhe 05] **F Künhe**, **J Gomes et W Fetter**. *Mobile robot trajectory tracking using model predictive control*. In II IEEE latin-american robotics symposium, 2005.

[Kyriakopoulos 88] **Konstantinos J Kyriakopoulos et George N Saridis**. *Minimum jerk path generation*. In Proceedings of the International Conference on Robotics and Automation, pages 364–369. IEEE, 1988.

[Lambrechts 04] **Paul Lambrechts**, **Matthijs Boerlage et Maarten Steinbuch**. *Trajectory planning and feedforward design for high performance motion systems*. Feedback, vol. 14, page 15, 2004.

[Lee 12] **Jaemin Lee**, **N. Mansard et Jaeheung Park**. *Intermediate Desired Value Approach for Task Transition of Robots in Kinematic Control*. IEEE Transactions on Robotics, vol. 28, n° 6, pages 1260–1277, 2012.

[Liégeois 77] **A. Liégeois**. *Automatic Supervisory Control of the Configuration and Behavior of Multibody Mechanisms*. IEEE Transactions on Systems, Man and Cybernetics, vol. 7, n° 12, pages 868–871, 1977.

[Lin 83] **Chun-Shin Lin**, **Po-Rong Chang et Js Ye So Luh**. *Formulation and optimization of cubic polynomial joint trajectories for industrial robots*. IEEE Transactions on Automatic Control, vol. 28, n° 12, pages 1066–1074, 1983.

[Liu 12] **Mingxing Liu**, **A. Micaelli**, **P. Evrard**, **A. Escande et C. Andriot**. *Interactive Virtual Humans: A Two-Level Prioritized Control Framework With Wrench Bounds*. IEEE Transactions on Robotics, vol. 28, n° 6, pages 1309–1322, 2012.

[Liu 16] **Mingxing Liu**, **Yang Tan et Vincent Padois**. *Generalized hierarchical control*. Autonomous Robots, vol. 40, n° 1, pages 17–31, 2016.

[Lober 15] **R Lober**, **V Padois et O Sigaud**. *Variance Modulated Task Prioritization in Whole-Body Control*. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 1–6, 2015.

[Lumelsky 87] **Vladimir J Lumelsky et Alexander A Stepanov**. *Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape*. Algorithmica, vol. 2, n° 1-4, pages 403–430, 1987.

[Macfarlane 03] **Sonja Macfarlane**, **Elizabeth Croft et others**. *Jerk-bounded manipulator trajectory planning: design for real-time applications*. IEEE Transactions on Robotics and Automation, vol. 19, n° 1, pages 42–52, 2003.

[Maciejewski 85] **Anthony A Maciejewski et Charles A Klein**. *Obstacle avoidance for kinematically redundant manipulators in dynamically varying environ-*

ments. The international journal of robotics research, vol. 4, n° 3, pages 109–117, 1985.

[Makarov 11] **Maria Makarov**, **Mathieu Grossard**, **Pedro Rodriguez-Ayerbe et Didier Dumur**. *Generalized predictive control of an anthropomorphic robot arm for trajectory tracking*. In Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 2011.

[Mansard 09] **Nicolas Mansard**, **Oussama Khatib et Abderrahmane Kheddar**. *A unified approach to integrate unilateral constraints in the stack of tasks*. IEEE Transactions on Robotics, vol. 25, n° 3, pages 670–685, 2009.

[Mistry 08] **Michael Mistry**, **Jun Nakanishi**, **Gordon Cheng et Stefan Schaal**. *Inverse kinematics with floating base and constraints for full body humanoid robot control*. In Proceedings of the IEEE/RAS International Conference on Humanoid Robots, pages 22–27, 2008.

[Muico 09] **Uldarico Muico**, **Yongjoon Lee**, **Jovan Popović et Zoran Popović**. *Contact-aware nonlinear control of dynamic characters*. ACM Transactions on Graphics, vol. 28, n° 3, page 81, 2009.

[Murray 94] **Richard M Murray**, **Zexiang Li**, **S Shankar Sastry et S Shankara Sastry**. A mathematical introduction to robotic manipulation. CRC press, 1994.

[Nakamura 87] **Yoshihiko Nakamura**, **Hideo Hanafusa et Tsuneo Yoshikawa**. *Task-priority based redundancy control of robot manipulators*. The International Journal of Robotics Research, vol. 6, n° 2, pages 3–15, 1987.

[Nise 07] **Norman S Nise**. Control systems engineering. John Wiley & Sons, 2007.

[Nocedal 06] **Jorge Nocedal et Stephen J Wright**. Numerical optimization, second edition. Springer New York, 2006.

[Padois 07] **Vincent Padois**, **Jean-Yves Fourquet**, **Pascale Chiron et others**. *Kinematic and dynamic model-based control of wheeled mobile manipulators: A unified framework for reactive approaches*. Robotica, vol. 25, n° 2, page 157, 2007.

[Pagilla 01] **P.R. Pagilla et Biao Yu Biao Yu**. *A stable transition controller for constrained robots*. IEEE/ASME Transactions on Mechatronics, vol. 6, n° 1, pages 65–74, 2001.

[Pang 96] **Jong-Shi Pang et Jeffrey C Trinkle**. *Complementarity formulations and existence of solutions of dynamic multi-rigid-body contact problems with Coulomb friction*. Mathematical programming, vol. 73, n° 2, pages 199–226, 1996.

[Park 98] **Ki Cheol Park**, **Pyung Hun Chang et Seung Ho Kim**. *The enhanced compact QP method for redundant manipulators using practical inequality constraints*. In Proceedings of the International Conference on Robotics and Automation, volume 1, pages 107–114, 1998.

[Petrič 13] **Tadej Petrič et Leon Žlajpah**. *Smooth continuous transition between tasks on a kinematic control level: Obstacle avoidance as a control problem*. Robotics and Autonomous Systems, vol. 61, n° 9, pages 948–959, 2013.

[Piazzi 00] **Aurelio Piazzi et Antonio Visioli**. *Global minimum-jerk trajectory planning of robot manipulators*. IEEE Transactions on Industrial Electronics, vol. 47, n° 1, pages 140–149, 2000.

[Raibert 81] **Marc H Raibert et John J Craig**. *Hybrid position/force control of manipulators*. Journal of Dynamic Systems, Measurement, and Control, vol. 103, n° 2, pages 126–133, 1981.

[Raunhardt 07] **Daniel Raunhardt et Ronan Boulic**. *Progressive clamping*. In Proceedings of the International Conference on Robotics and Automation, pages 4414–4419, 2007.

[Rubrecht 10] **Sébastien Rubrecht**, **Vincent Padois**, **Philippe Bidaud et Michel De Broissia**. *Constraints Compliant Control: constraints compatibility and the displaced configuration approach*. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 677–684, 2010.

[Saab 13] **Layale Saab**, **Oscar E. Ramos**, **Francois Keith**, **Nicolas Mansard**, **Philippe Soueres et Jean-Yves Fourquet**. *Dynamic Whole-Body Motion Generation Under Rigid Contacts and Other Unilateral Constraints*. IEEE Transactions on Robotics, vol. 29, n° 2, pages 346–362, 2013.

[Sadeghian 13] **Hamid Sadeghian**, **Luigi Villani**, **Mehdi Keshmiri et Bruno Siciliano**. *Dynamic multi-priority control in redundant robotic systems*. Robotica, vol. 31, n° 07, pages 1155–1167, 2013.

[Salini 11] **J. Salini**, **V. Padois et P. Bidaud**. *Synthesis of complex humanoid whole-body behavior: A focus on sequencing and tasks transitions*. In Proceedings of the International Conference on Robotics and Automation, pages 1283 –1290, 2011.

[Salini 12a] **J. Salini**. *Dynamic control for the task/posture coordination of humanoids: toward synthesis of complex activities*. Ph.d. thesis, Université Pierre et Marie Curie, Paris, France, June 2012.

[Salini 12b] **J. Salini**, **S. Barthélemy et A. Micaelli**. *Arboris-Python*, 2012.

[Samson 91] **Claude Samson**, **Bernard Espiau et Michel Le Borgne**. Robot control: the task function approach. Oxford University Press, 1991.

[Schatzman 02] **Michelle Schatzman et Michelle Schatzman**. Numerical analysis: a mathematical introduction. Oxford University Press, 2002.

[Sentis 04] **L. Sentis et O. Khatib**. *Task-Oriented Control of Humanoid Robots Through Prioritization*. In Proceedings of the IEEE/RAS International Conference on Humanoid Robots, 2004.

[Sentis 10] **L. Sentis**, **Jaeheung Park et O. Khatib**. *Compliant Control of Multi-Contact and Center of Mass Behaviors in Humanoid Robots*. IEEE Transactions on Robotics, vol. 26, n° 3, pages 483–501, 2010.

[Siciliano 90] **Bruno Siciliano**. *Kinematic control of redundant robot manipulators: A tutorial*. Journal of Intelligent and Robotic Systems, vol. 3, n° 3, pages 201–212, 1990.

[Siciliano 91] **Bruno Siciliano et Jean-Jacques E Slotine**. *A general framework for managing multiple tasks in highly redundant robotic systems*. In Proceedings of the International Conference on Advanced Robotics, pages 1211–1216, 1991.

[Siciliano 08] **Bruno Siciliano et Oussama Khatib**. Springer handbook of robotics, chapter 27. Contact Modeling and Manipulation. Springer Science & Business Media, 2008.

[Simon 93] **Dan Simon**. *The application of neural networks to optimal robot trajectory planning*. Robotics and Autonomous Systems, vol. 11, n° 1, pages 23–34, 1993.

[Stasse 08] **Olivier Stasse**, **Adrien Escande**, **Nicolas Mansard**, **Sylvain Miossec**, **Paul Evrard et Abderrahmane Kheddar**. *Real-time (self)-collision avoidance task on a hrp-2 humanoid robot*. In Proceedings of the International Conference on Robotics and Automation, pages 3200–3205, 2008.

[Sugiura 10] **Hisashi Sugiura**, **Michael Gienger**, **HERBERT JANßEN et Christian Goerick**. *Reactive self collision avoidance with dynamic task prioritization for humanoid robots*. International Journal of Humanoid Robotics, vol. 7, n° 01, pages 31–54, 2010.

[Tarn 96] **Tzyh-jong Tarn**, **Yunying Wu**, **Ning Xi et Alberto Isidori**. *Force regulation and contact transition control*. Control System, IEEE, vol. 16, n° 1, pages 32–40, 1996.

[Volpe 93] **Richard Volpe et Pradeep Khosla**. *A theoretical and experimental investigation of impact control for manipulators*. The International Journal of Robotics Research, vol. 12, n° 4, pages 351–365, 1993.

[Vukobratović 04] **Miomir Vukobratović et Branislav Borovac**. *Zero-moment point-thirty five years of its life*. International Journal of Humanoid Robotics, vol. 1, n° 01, pages 157–173, 2004.

[Walker 90] **Ian D Walker**. *The use of kinematic redundancy in reducing impact and contact effects in manipulation*. In Proceedings of the International Conference on Robotics and Automation, pages 434–439, 1990.

[Wensing 13] **Patrick M Wensing et David Orin**. *Generation of dynamic humanoid behaviors through task-space control with conic optimization*. In Proceedings of the International Conference on Robotics and Automation, pages 3103–3109, 2013.

[Whitney 77] **Daniel E Whitney**. *Force feedback control of manipulator fine motions*. Journal of Dynamic Systems, Measurement, and Control, vol. 99, n° 2, pages 91–97, 1977.

[Whitney 87] **Daniel E Whitney**. *Historical perspective and state of the art in robot force control*. The International Journal of Robotics Research, vol. 6, n° 1, pages 3–14, 1987.

[Wieber 06] **Pierre-Brice Wieber**. *Trajectory free linear model predictive control for stable walking in the presence of strong perturbations*. In Proceedings of the IEEE/RAS International Conference on Humanoid Robots, pages 137–142. IEEE, 2006.

[Wu 96] **Yunying Wu**, **Tzyh-Jong Tarn**, **Ning Xi et Alberto Isidori**. *On robust impact control via positive acceleration feedback for robot manipulators*. In Proceedings of the International Conference on Robotics and Automation, volume 2, pages 1891–1896. IEEE, 1996.

[Youcef-Toumi 89] **K Youcef-Toumi et DA Gutz**. *Impact and force control*. In Proceedings of the International Conference on Robotics and Automation, pages 410–416. IEEE, 1989.

[Zhang 04a] **Yunong Zhang**, **Shuzhi Sam Ge et Tong Heng Lee**. *A unified quadratic-programming-based dynamical system approach to joint torque optimization of physically constrained redundant manipulators*. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 34, n° 5, pages 2126–2132, 2004.

[Zhang 04b] **Yunong Zhang et Jun Wang**. *Obstacle avoidance for kinematically redundant manipulators using a dual neural network*. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 34, n° 1, pages 752–759, 2004.

Appendix A

Proof of the maintenance of strict hierarchies represented by standard lexicographic orders subject to constraints

This section proves that the proposed GHC approach (3.9) can maintain strict task hierarchies represented by standard lexicographic orders while accounting for linear constraints.

Suppose there are n_t tasks that should be organized in a way such that each task i has a strict lower priority than task $i-1$ with i = 2, ..., n_t. In this case, the generalized projector $\mathbf{P}_i$ for a task i is in fact a null-space projector, which projects a task Jacobian into the null-space of all the previous $i-1$ tasks, and each $\mathbf{A}_i$ is an identity matrix. Let each task objective function be $\boldsymbol{f}_i = \mathbf{J}_i \boldsymbol{x}'_i - \boldsymbol{x}^d_i$, with $\boldsymbol{x}'_i$ being a joint space task variable, such as $\dot{\boldsymbol{q}}'_i$, $\ddot{\boldsymbol{q}}'_i$, or $\boldsymbol{\tau}'_i$, etc. Moreover, the global variable $\boldsymbol{x} = \sum_i \mathbf{P}_i \boldsymbol{x}'_i$ should satisfy linear equality or inequality constraints $\mathbf{G}\boldsymbol{x} \leq \boldsymbol{h}$.

At the first stage, the regulation term is neglected, and the optimization problem can be written as follows

$$\begin{aligned} &\underset{\boldsymbol{x}'_{(n_t)}}{\arg\min} \sum_{i=1}^{n_t} \left\| \mathbf{J}_i \boldsymbol{x}'_i - \boldsymbol{x}^d_i \right\|^2 \\ &\text{subject to } \mathbf{G} \sum_{i=1}^{n_t} \mathbf{P}_i \boldsymbol{x}'_i \leq \boldsymbol{h} \end{aligned}, \tag{A.1}$$

where $\boldsymbol{x}'_{(n_t)} = \left\{ \boldsymbol{x}'_1, \boldsymbol{x}'_2, \ldots \boldsymbol{x}'_{n_t} \right\}$, and the solution to (A.1) is denoted as $\boldsymbol{x}^*_{(n_t)} = \left\{ \boldsymbol{x}^*_1, \boldsymbol{x}^*_2, \ldots \boldsymbol{x}^*_{n_t} \right\}$.

When $n_t = 1$, the optimization problem can be written as

$$\begin{aligned} &\underset{\boldsymbol{x}'_{(1)}}{\arg\min} \left\| \mathbf{J}_1 \boldsymbol{x}'_{(1)} - \boldsymbol{x}_1^d \right\|^2 \\ &\text{subject to } \mathbf{G}\boldsymbol{x}'_{(1)} \leq \boldsymbol{h} \end{aligned} . \tag{A.2}$$

The solution to this problem $\boldsymbol{x}^*_{(1)}$ is the same as the one to the problem formulated by HQP.

When $n_t = k$, the optimization problem is formulated as

$$\begin{aligned} &\underset{\boldsymbol{x}'_{(k)}}{\arg\min} \sum_{i=1}^{k} \left\| \mathbf{J}_i \boldsymbol{x}'_i - \boldsymbol{x}_i^d \right\|^2 \\ &\text{subject to } \mathbf{G} \sum_{i=1}^{k} \mathbf{P}_i \boldsymbol{x}'_i \leq \boldsymbol{h} \end{aligned} . \tag{A.3}$$

Suppose the solution $\boldsymbol{x}^*_{(k)}$ can maintain the strict task hierarchy: if a task $k+1$ is inserted with lowest priority with respect to the set of k tasks, then the optimization problem with the $k+1$ tasks can be written as

$$\begin{aligned} &\underset{\boldsymbol{x}'_{(k+1)}}{\arg\min} \sum_{i=1}^{k} \left\| \mathbf{J}_i \boldsymbol{x}'_i - \boldsymbol{x}_i^d \right\|^2 + \left\| \mathbf{J}_{k+1} \boldsymbol{x}'_{k+1} - \boldsymbol{x}_{k+1}^d \right\|^2 \\ &\text{subject to } \mathbf{G} \left(\sum_{i=1}^{k} \mathbf{P}_i \boldsymbol{x}'_i + \mathbf{P}_{k+1} \boldsymbol{x}'_{k+1} \right) \leq \boldsymbol{h} \end{aligned} . \tag{A.4}$$

As $\mathbf{P}_k \mathbf{P}_{k+1} = \mathbf{P}_{k+1}$, the term $\sum_{i=1}^{k} \mathbf{P}_i \boldsymbol{x}'_i + \mathbf{P}_{k+1} \boldsymbol{x}'_{k+1}$ in the constraint in (A.4) is equivalent to $\sum_{i=1}^{k-1} \mathbf{P}_i \boldsymbol{x}'_i + \mathbf{P}_k \varsigma_k$, with

$$\varsigma_k = \boldsymbol{x}'_k + \mathbf{P}_{k+1} \boldsymbol{x}'_{k+1} . \tag{A.5}$$

Then problem (A.4) can be written as

$$\begin{aligned} \underset{\boldsymbol{x}'_{(k)},\varsigma_k,\boldsymbol{x}_{k+1}}{\arg\min} \sum_{i=1}^{k-1} \left\| \mathbf{J}_i \boldsymbol{x}'_i - \boldsymbol{x}^d_i \right\|^2 + \left\| \mathbf{J}_k \varsigma_k - \boldsymbol{x}^d_k \right\|^2 + \\ \left\| \mathbf{J}_{k+1} \boldsymbol{x}'_{k+1} - \boldsymbol{x}^d_{k+1} \right\|^2 \\ \text{subject to } \mathbf{G} \left(\sum_{i=1}^{k-1} \mathbf{P}_i \boldsymbol{x}'_i + \mathbf{P}_k \varsigma_k \right) \leq \boldsymbol{h} \\ \varsigma_k = \boldsymbol{x}'_k + \mathbf{P}_{k+1} \boldsymbol{x}'_{k+1} \end{aligned} . \tag{A.6}$$

$\boldsymbol{x}'_k$ in (A.6) is a free variable, and this problem can be separated into two sub-problems. The first sub-problem is

$$\begin{aligned} \underset{\boldsymbol{x}'_{(k-1)},\varsigma_k}{\arg\min} \sum_{i=1}^{k-1} \left\| \mathbf{J}_i \boldsymbol{x}'_i - \boldsymbol{x}^d_i \right\|^2 + \left\| \mathbf{J}_k \varsigma_k - \boldsymbol{x}^d_k \right\|^2 \\ \text{subject to } \mathbf{G} \left(\sum_{i=1}^{k-1} \mathbf{P}_i \boldsymbol{x}'_i + \mathbf{P}_k \varsigma_k \right) \leq \boldsymbol{h} \end{aligned} . \tag{A.7}$$

The optimal solution $\sum_{i=1}^{k-1} \mathbf{P}_i \boldsymbol{x}^{*,\prime}_i + \mathbf{P}_k \varsigma^*_k$ to this problem is equivalent to the one of (A.3). Indeed, these two solutions have the same effect on task k

$$\mathbf{J}_k \sum_{i=1}^{k} \mathbf{P}_i \boldsymbol{x}^{*,\prime}_i = \mathbf{J}_k \left(\sum_{i=1}^{k-1} \mathbf{P}_i \boldsymbol{x}^{*,\prime}_i + \mathbf{P}_k \varsigma^*_k \right) . \tag{A.8}$$

To prove (A.8), one needs to notice that $\mathbf{J}_i \mathbf{P}_j = \mathbf{0}$ with $j \geq i$. The second sub-problem is given by

$$\underset{\boldsymbol{x}_{k+1}}{\arg\min} \left\| \mathbf{J}_{k+1} \boldsymbol{x}'_{k+1} - \boldsymbol{x}^d_{k+1} \right\|^2 . \tag{A.9}$$

Therefore, the insertion of a lower priority task $k+1$ does not change the optima of the k previous task objectives. In other words, the strict task hierarchy of an arbitrary number of tasks subject to linear constraints can be maintained.

We have proved that each lower priority task will not increase the obtained optima of all the previous tasks. The rest of this proof explains the roles of the regulation term. As mentioned in Section 3.2, the use of a regulation term, which minimizes the norm of

each task variable, helps to ensure the uniqueness of the solution. As each task objective i is assigned with the weight $\omega_i = 1$, which is much greater than the weight of the regulation term ($\omega_r << 1$), the task variables are optimized to mainly satisfy task objectives. Moreover, in GHC, this regulation term also helps to improve the performance of lower priority tasks. Consider $k+1$ levels of tasks to handle, as $\mathbf{J}_i\mathbf{P}_j = \mathbf{0}$ with $j \geq i$, the final solution is $\sum_{i=1}^{k} \mathbf{P}_i \boldsymbol{x}_i^* + \mathbf{P}_{k+1}\boldsymbol{x}_{k+1}^*$. Denoting the elements required by task i as $\boldsymbol{x}_i^{i,*}$ and the rest elements that are are not effectively handled by task objective i as $\boldsymbol{x}_i^{f,*}$, the final solution can be rewritten as $S = \sum_{i=1}^{k} \mathbf{P}_i^i \boldsymbol{x}_i^{i,*} + \sum_{i=1}^{k} \mathbf{P}_i^f \boldsymbol{x}_i^{f,*} + \mathbf{P}_{k+1}\boldsymbol{x}_{k+1}^*$, with $\mathbf{P}_i^i$ and $\mathbf{P}_i^f$ the columns in $\mathbf{P}_i$ that correspond to $\boldsymbol{x}_i^{i,*}$ and $\boldsymbol{x}_i^{f,*}$ respectively. The term $\sum_{i=1}^{k} \mathbf{P}_i^f \boldsymbol{x}_i^{f,*}$ that is not required by the k previous tasks may contribute to task $k+1$ and affect its task performance. The minimization of the norm of $\boldsymbol{x}_i^f$ in the regulation term improves the performance of task $k+1$ by making S closer to $\sum_{i=1}^{k} \mathbf{P}_i^i \boldsymbol{x}_i^{i,*} + \mathbf{P}_{k+1}\boldsymbol{x}_{k+1}^*$, where $\mathbf{P}_i^i \boldsymbol{x}_i^{i,*}$ are used to perform the k previous tasks and $\mathbf{P}_{k+1}\boldsymbol{x}_{k+1}^*$ is used to perform the $(k+1)$-th task in the null-space of all the higher priority tasks.