Technische Universiteit
Eindhoven
University of Technology

Department of Mechanical Engineering
Dynamics & Control Research Group

Improvement of a Balancing Force and Posture Controller with Torque Minimization

DC 2015.006 - Internship Project Report

Talha Ali Arslan

Supervisors:
prof.dr. Henk Nijmeijer
dr. Alessandro Saccon
dr. Francesco Nori (IIT)
dr. Daniele Pucci (IIT)

Eindhoven, December 2014

Internship report

Talha Ali Arslan

December 12, 2014

Preface

This report is the result of an internship project that took place in the Robotics, Brain and Cognitive Sciences (RBCS) group in the Italian Institute of Tecnology in Genoa,Italy between September 1st, to November 14th in 2014 for 11 weeks, as part of the Mechanical Engineering MSc study program in TU/e (Einhoven University of Technology) in Eindhoven, The Netherlands.

On the IIT side, I would like to thank the RBCS group members, dr. Naveen Kuppuswamy, Silvio Traversaro, Jorhabib Eljaik, Francesco Romano and Luca Fiorio for assisting and helping me whenever I needed and for creating such a friendly atmosphere to work in. I would also like to thank dr. Daniele Pucci, for his supervision and guidance throughout this project and all the discussions we had which were greatly useful. Finally, I would like to thank dr. Francesco Nori, who is also the team leader, for his supervision and for hosting me in a group of such great spirited people.

On the TU/e side, I would like to thank dr. Alessandro Saccon for his support and encouragement and providing me with this opportunity. Finally, I would like to thank prof.dr. Henk Nijmeijer for supporting and providing me with opportunities since the beginning of my study.

Talha Ali Arslan
December, 2014

Abstract

To improve the balancing of the humanoid robot iCub, the minimization torques and conditions on contact wrenches for balancing stability are implemented to the force and posture control of iCub which is made possible thanks to its wholebody distributed force&torque sensors. The new formulation of the control problem optimizes the feet contact forces in order to minimize the joint torques which helps in reducing any unnecessary internal torques, and during the optimization, conditions on the feet contact forces that ensure stable balancing such as friction cones and zero moment point are imposed as linear inequalities onto the formulized quadratic program (QP). The new controller is also tested by simulations and experiments on the real robot.

Contents

Chapter 1

Introduction

1.1 Motivation

Robots today have more interaction with their environment and users than ever before and it seems that this trend will continue as they become more common in the industry and the daily life. To ensure the safety for all of the elements of the environment that are subject to possible damage in case of an interaction (including the robot itself), the robot must be able to anticipate and/or reciprocate any interactions in a safe manner.

Various control methods are used to ensure that the robots follow and complete their tasks as planned before or during their operation. However, the sensitivity of the robot to its environment has limits and to ensure safety, the question of how the robot will take action in the case of an unexpected change in the environment, must be answered profoundly. Especially in the field of humanoid robotics, safety of interaction is a major issue since they are designed and built to operate and interact with their environment in various ways such as care-giving for the elderly, servicing and assisting in different scenarios and working side by side with humans. Since the diversity of possible end-users of such robots is growing continuously, it is fair to state that the possible end-user is not always meant to be a person who has broad knowledge on how such a robotic platform operates and who can make sure of meeting the safety rules at all times. Hence, it is the engineers' duty to make sure that the robot can operate safely in the cases of unpredicted events in the robot's environment.

If a robot, which is controlled by the positions of its degrees of freedom that can be actuated to complete given tasks, is allowed to interact with its environment without being able to sense what is going on in its surroundings and update its tasks accordingly; any disturbance, unmodeled behavior or unexpected interaction can have a catastrophic result. Nevertheless, the safety of interaction with a position controlled robot can be improved in many ways. Any one or combination of the items below can contribute to safety of interactions:

- Addition of compliance to the system, which can highly reduce damage due to impacts, which may be achieved by using compliant mechanisms and/or control strategies to mimic compliant behavior.
- Incorporation of more sensory data (via additional sensors for various purposes) into the planning and updating of tasks during the operation for better estimation of states, and to anticipate or sense interactions with the environment.
- Addition of more restrictions on the environment to operate in.

Another method is the force control in which the torques on the joints, rather than their position or velocity, are controlled to generate desired forces at contact points. This can be

achieved by using force&torque sensors to estimate joint torques, contact forces and the dynamical model of the robot. At the cost of the estimation of forces and dynamical model using sensors, safety of interactions with the environment is achieved. In any case of interaction or unexpected impacts, compliant behavior is achieved at the joints for they are controlled to follow desired torques rather than position or velocity which otherwise would result in stiff behavior. However, force control strategy differs also in terms of given tasks and it is not as popular as position control, for now.

The humanoid robot iCub of the Italian Institue of Technology (IIT) in Genova, Italy, is a suitable platform for both position and force control with its various whole body distributed sensors such as force&torque and tactile skin sensors. In the Cognitive Humanoids Laboratory of the Robotics, Brain and Cognitive Sciences Department, a force and posture controller is used to keep the robot balanced by generating the desired contact forces at both feet. It is desired to use the possibilities of force control, to have a humanoid robot that can safely interact with its environment as humans do in the daily life. Walking is one of the necessary activities and it seems that the iCub is approaching that goal as it is already able to balance on its feet and move its center of mass with respect to given desired trajectories. This is the stage this internship project is aimed to contribute to, with the implementation of joint torque minimization while respecting the conditions on stable balancing, which will improve the balancing of the iCub to a point where it will hopefully decide to take one foot of the ground soon.

1.2 Objectives

The main objective of this project is to improve the balancing control of the humanoid robot iCub, by means of implementing new features such as the minimization of the joint torques and the consideration of the conditions related to stability during balancing. Objectives alongside the main one are the familiarization with the state of the art software and hardware that is used for designing, building and experimenting with humanoid robots capable of whole body force, torque and position control and to explore more of the theoretical and practical knowledge related to the dynamics and control of humanoid robots.

1.3 Outline

The outline of this report is as follows:

- In Chapter 2, a brief introduction to the dynamics and modeling of humanoid robots as floating-base systems, is given,
- In Chapter 3, the main actor of this and many other projects, the humanoid robot iCub of IIT[1] is introduced within the context of whole body control and this project.
- In Chapter 4, several concepts and conditions on the balancing task are explained and a balancing force and posture controller that is the starting point of this project is explained.
- In Chapter 5, the new controller is introduced along with the new formulation to achieve minimization of torques and to satisfy necessary conditions for balancing stability, followed by the results from simulations and experiments on the robot and discussions.
- In Conclusion, final remarks are made.

[1] Istituto Italiano di Tecnologia (Italian Institute of Technology), Genova, Italy

Chapter 2

Dynamics of Humanoid Robots

In this chapter, a brief introduction to the dynamics and equations of motion of humanoid robots is given by short descriptions of the classes of systems humanoids are related or belong to; to familiarize ourselves with the descriptions of dynamics of humanoid robots as constrained floating-base systems.

2.1 Fully Actuated Systems

Definition: Fully actuated systems have exactly the same number of control inputs and total number of degrees of freedom.

The equations of motion of a fully-actuated system can be defined as;

$$M(q)\ddot{q} + h(q, \dot{q}) = \tau\,, \tag{2.1}$$

where $q \in \mathbb{R}^n$ is the vector of generalized coordinates, $M(q) \in \mathbb{R}^{nxn}$ is the inertia matrix, $\tau \in \mathbb{R}^n$ is the vector of generalized input torques and $h(q, \dot{q}) \in \mathbb{R}^n$ is the vector of bias forces containing Coriolis, centrifugal and gravity forces.

With fixed-base and fully actuated systems at initial configuration q_0 and $\dot{q}(0) = 0$, a desired trajectory q^* which can be defined as;

$$q^* = q_0 + \iint_0^T \ddot{q}^* \,\mathrm{d}t \tag{2.2}$$

and $\ddot{q}^*$ can directly be used to calculate input torque trajectory that is necessary to follow q^* by;

$$\tau^* = M(q)\ddot{q}^* + h(q, \dot{q}). \tag{2.3}$$

which is not the case for systems that are not fully actuated.

2.2 Underactuated Systems

Definition: Underactuated systems are characterized by having fewer number of control inputs than the total number of degrees of freedom.

For the underactuated systems, the equations of motion can be written as;

$$M(q)\ddot{q} + h(q, \dot{q}) = S^T\tau. \tag{2.4}$$

with the only difference, from the equations of motion of fully actuated systems (2.1), being $\tau \in \mathbb{R}^a$ where a is the number of actuated degrees of freedom with $a < n$ and $S \in \mathbb{R}^{a\times n}$ being the selector matrix which selects the control inputs for the a active joints rather than $(n - a)$ passive joints.

Due to the difference between the total number of degrees of freedom (n) and the number of actuated degrees of freedom ($a < n$), the underactuated systems are not fully feedback linearizable and the problem of finding an acceleration trajectory $\ddot{q}*$ that brings the system from an initial configuration q_0 to a desired configuration q^* is not trivial at all, because the acceleration of the system that can be achieved at an instant of time is not available in all directions and it is also dependent on the states due to the fact that,

$$\ddot{q} = M^{-1}(q)\big(- h(q, \dot{q}) + S^T\tau^*\big) \tag{2.5}$$

defines $\ddot{q} \in \mathbb{R}^n$ at an instant of time with input torques $\tau^* \in \mathbb{R}^a$ where $a < n$ meaning that the n-dimensional vector $\ddot{q}$ lies on a manifol of dimension a, because of the underactuation. Although acceleration at an instance is only available in a subset of all directions, it is still possible that the system can be controllable, meaning that there can be found a trajectory for input torques τ^* that can achieve an acceleration trajectory $\ddot{q}^*$ that will bring the system from any initial configuration q_0 to a any final desired configuration q^* in a finite time.

2.3 Floating-base Systems

Floating-base systems, being a class of underactuated systems, have their equations of motion defined similar to (2.4) by adding a passive joint with 6 degrees of freedom from the base of the robot to the inertial frame resulting in a model in the same way as a fixed-base underactuated system;

$$M(q)\ddot{q} + h(q, \dot{q}) = S^T\tau, \tag{2.6}$$

with,

$$q = \begin{bmatrix} x_b \\ q_j \end{bmatrix} \in \mathbb{R}^{n+6}, \begin{matrix} x_b \in \mathbb{R}^6 \\ q_j \in \mathbb{R}^n, \end{matrix} \quad M(q) \in \mathbb{R}^{(n+6)x(n+6)}, h(q) \in \mathbb{R}^{n+6}, \quad S^T = \begin{bmatrix} 0_{6xn} \\ I_{nxn} \end{bmatrix} \tag{2.7}$$

where the vector of generalized coordinates $q \in \mathbb{R}^{n+6}$ is formed by combination of the 6 degrees of freedom (position and orientation) that come from the generalized coordinates of the base $x_b \in \mathbb{R}^6$ and generalized coordinates of the actuated joints $q_j \in \mathbb{R}^n$, which is typical for a robot with all of its joints actuated and free floating in 3-dimensional space.

2.4 Humanoid Robots as Constrained Floating-base Systems

As they are not fixed to any reference frame and can be found floating freely, humanoids are a class of floating base systems. However, humanoid robots usually have contacts -almost always with the ground-. Hence, they can be described as constrained floating-base systems.

Making contacts with environment results in constraints on the motion at the contact points which restricts movements in constrained directions where the constraint forces act, Assuming that the contacts are rigid and holonomic, the constraints on movements can be represented by imposing zero velocity at the contact points as;

$$J_c(q)\dot{q} = 0\,, \tag{2.8}$$

and also at the acceleration level as;

$$J_c(q)\ddot{q} + \dot{J}_c(q)\dot{q} = 0\,, \tag{2.9}$$

where $J_c(q) \in \mathbb{R}^{k\times(n+6)}$ is the jacobian of the contact points and k is the number of constrained directions of motion.

With the addition of constraints, forces at the rigid contact points act on the body and equation of motion of constrained floating-base systems becomes;

$$\begin{aligned} M(q)\ddot{q} + h(q,\dot{q}) - J_c^T(q) f_c &= S^T\tau, \\ \text{s.t.} \quad J_c(q)\ddot{q} + \dot{J}_c(q)\dot{q} &= 0. \end{aligned} \tag{2.10}$$

where $q \in \mathbb{R}^{(n+6)}$ is the vector of generalized coordinates containing the position and orientation of the base $\begin{bmatrix} x_b & w_b \end{bmatrix}^T \in \mathbb{R}^6$ and the joint positions $q_j \in \mathbb{R}^n$, $M(q) \in \mathbb{R}^{n\times n}$ is the inertia matrix, $\tau \in \mathbb{R}^n$ is the vector of generalized input torques, $f_c \in \mathbb{R}^k$ is the constraint forces that maintain the constraints, $J_c(q) \in \mathbb{R}^{k\times n}$ is the constraint jacobian and $h(q,\dot{q}) \in \mathbb{R}^n$ is the vector of bias forces containing Coriolis, centrifugal and gravity forces.

Chapter 3

The Humanoid iCub

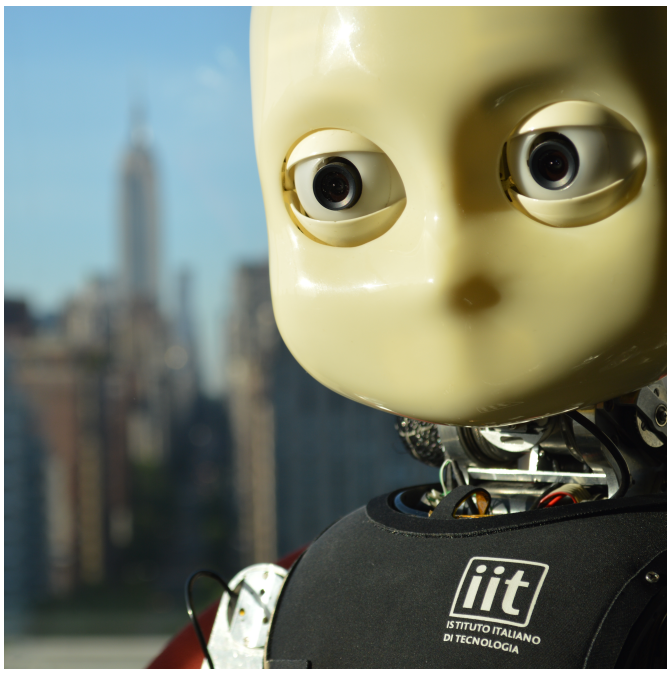

Figure 3.1: iCub[1]

The humanoid robot iCub is an open-source platform for research in humanoid robotics, brain and cognitive sciences. Being a platform that is used in such broad fields of research, it has various sensors and mechanical features that allow testing for various tasks. At the size of a human child, iCub has four limbs consisting of two legs and two arms including hands with five fingers and various sensors such as force&torque, tactile skin, gyroscopes, accelerometers and cameras distributed over its body to be able to sense its environment. Basically, it is a humanoid that is built with the intention to allow it to interact with people and the environment. In this chapter, we introduce the humanoid iCub in the context of whole body force and posture control for balancing; by describing the mechanical characteristics, whole body distributed sensors and software to be used for the purposes of this project. It should be noted that these are only a subset of all the features of the humanoid iCub and more details on the platform can be found in [1].

[1]Photo Credit: Istituto Italiano di Tecnologia - http://www.iit.it/en/social/photo-gallery.html

3.1 Mechanics

iCub has 53 degrees of freedom that are actuated, distributed as; 6 for the head (3 for the neck and 3 for the eyes), 3 for the torso, 16 for each arm (9 of which are for each hand) and 6 for each leg. Except for the hands and the head where brushed DC motors are used, all of the remaining joints are actuated by brushless DC motors with harmonic drive gears.

The material used for the body parts is mostly Ergal (an aluminum alloy). Steel and plastic is also used and the total mass of the robot is around 25 kg. It does not have a certain mass since it is a research platform that is subject to be customized by researchers by removal, addition and testing of new sensors and parts.

Figure 3.2: iCub Mechanics[2]

For the purpose of the project this report is on, 25 degrees of freedom from the arms (3 from each shoulder, 2 from each elbow), the legs (3 from each hip, 1 from each knee and 2 from each ankle) and the torso are exploited and others such as the wrists, the hands and the head are kept at an initial configuration with position control.

iCub is not fully autonomous, yet, because power is supplied to the robot via a cord which also communicates with a cluster machine where the higher-level control takes place. For the

[2]Image Credit: Istituto Italiano di Tecnologia - http://www.iit.it/en/products/catalog.html

low-level motor control, there are microcontroller boards on the robot that can be used. During the course of this project, for joint torque control, high and low level control were both carried out at 100 Hz by the off-board cluster machine. To implement the estimation of joint torques to on-board means of computation is an ongoing process which can allow low-level joint torque control at a higher frequency with less delays.

3.2 Sensors

iCub is equipped with sensors such as; two digital cameras in the head, microphones, force&torque sensors, tactile skin, gyroscopes and accelerometers. In addition, there are absolute position encoders in every actuated joint.

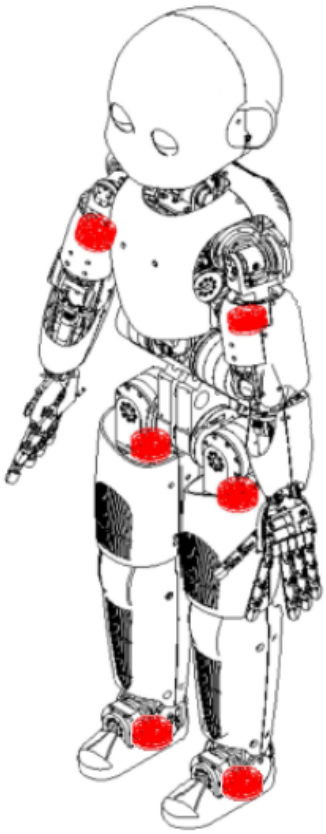

Figure 3.3: Force/torque sensors on iCub

Whole-body distributed force&torque sensors play an important role in the estimation of joint torques and the control the torques at each joint. There are six force&torque(F/T) sensors; one between the shoulder and the elbow on each arm, one between the hip and the knee and one between the ankle and the foot on each leg, as depicted in Figure 3.3. By using these sensors, internal forces can be estimated at the joints which in turn can be used to estimate internal dynamis and external forces.

The framework proposed by [2] for the estimation of internal/external forces on the floating-base system consists of seperating the whole body kinematic tree of the floating-base into subtrees by the location of the F/T sensors. In return, independent kinematic subtrees are obtained which are also governed by Newton-Euler dynamic equations. Each sensor gives a direct mea-

surement of the external wrench acting on that body consisting of the subtree. Then by using these external wrenches in a Newton-Euler force propagation recursion, the joint torques can be estimated. Hence, a torque controller with joint-torque friction compensation is able to provide desired torques by using the estimation from the measurement of internal forces and torques. Tactile skin sensors can also be incorporated into this process to estimate external wrenches with their accurate locations and this is an ongoing work too.

Specifications of the humanoid iCub can also be summarized as in the table below[3].

Height	104cm
Weight	23-25 kg
Sensors	Stereo Cameras, microphones, encoders, force/torque sensors, tactile sensors (capacitive) fingertips and skin on upper body and arms, gyroscopes, accelerometers
Actuators	Large joints as e.g. the shoulder (brushless motors, 150W), small joints as e.g. hands (DC motors). 54 motors in total
Power	220/110V AC, tethered via 48-12V power supply
Computing (On-board)	20 microcontroller boards for movement, 16 boards for sensors and a Pentium duo for data acquisition and synchronization.
Computing (Off-board)	a cluster with 30-40 cores and GPU processing and more if needed
Software (On-board)	Debian Linux.
Software (Off-board)	any of Windows, Linux, MacOs in any combination depending on the configuration/user needs.
	Software middleware controlling the cluster and the robot called YARP
Degrees of Freedom	53 motors controlling 76 joints
Structure and Materials	Mostly Ergal (aluminum alloy), steel and plastic
Cost	Latest configuration about 250.000 Euros
Year	Started 2004, first release 2008
Location	Genoa, Rome, Lyon, Paris, Barcelona, Munich, Bielefeld, London, Plymouth, Aberystwyth, Lisbon, Urbana-Champaign and Ankara

3.3 iCub Software for Simulations

Being an open-source research platform that is used in more than 20 different locations around the world, iCub project is maintained by a large research community. This way, it is easy to access and even contribute to any materials on research related to iCub.

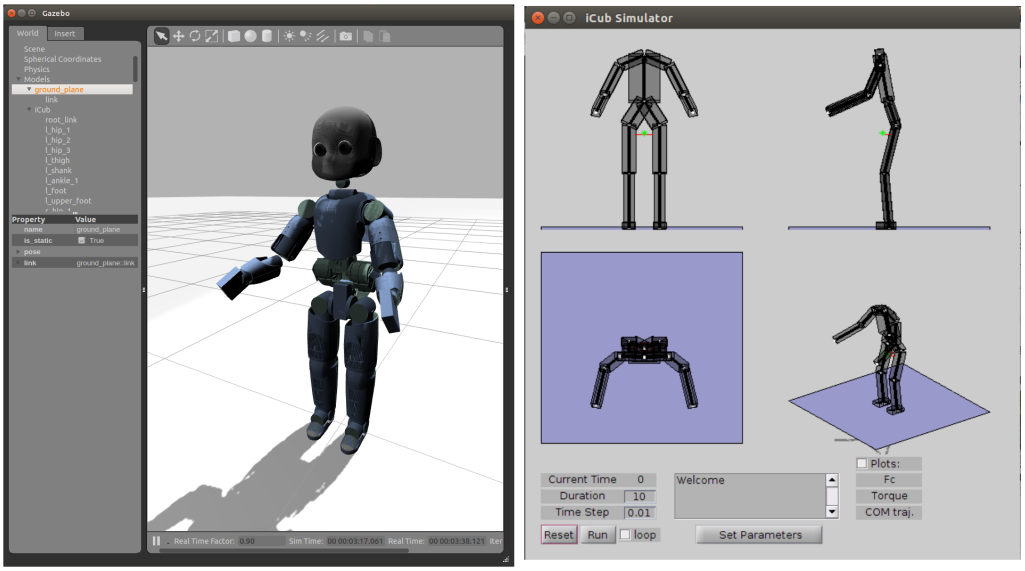

Figure 3.4: Simulating iCub in Gazebo (left) & Matlab (right)

Besides the real experimental platform, the humanoid iCub can also be simulated and switching between simulations and the real platform does not require a lot of effort. Simulations could previously be done by using Gazebo. However because of the complexity of the robot model and a lack of freedom on customization of the Gazebo simulator, force&torque control simulations could not be done properly, even if they were very simple tasks and they were also previously validated on the real robot. For this reason, a software package using Matlab is developed, to which this internship project also contributed. It is still under development but it was extensively used for prototyping purposes and it will be explained in Section 5.3: Simulation Environment.

To be able work on iCub either through simulations or on one of the real platforms, certain packages of software needs to be installed and this can be done by following the instructions in Wiki page for the iCub [4] and packages' related repositories on GitHub[3]. Any one of the Linux, Mac OS X or Windows operating systems can be used. During this project, Linux-based Ubuntu 14.04 LTS was used.

One of the core component of the related software is the software library iDynTree which is used to obtain the estimated joint torques and external wrenches in an efficient, generic and easy way. The library uses the kinematic information and the data from force&torque sensors to do the estimations with methods from [5, 6]. Another core software is YARP (Yet Another Robot Platform)[Metta et.al.2006] which is a middleware used for decoupling the all the software modules and devices cleanly. By using different connection types, it allows creating a collection of programs that communicate in a peer-to-peer way. It is also free and open-source. All the other related software necessary to be able to work with iCub is available with instructions to

[3]https://github.com/

install at the iCub Wiki[4].

Currently, the balancing controller for the iCub is tested on the real robot using Matlab Simulink. The WBI Toolbox in Simulink that comes along with the iCub software consists of Simulink blocks that provide the robot dynamics, kinematics, measurments etc. to be able to compute desired joint torques within a controller. Being able to design the controller in Simulink makes it easier for the researchers to understand, debug and modify them. However, it should be noted that using Simulink creates extra computational effort and this process can be done a lot faster using C/C++ implementations of the controllers. Considering that the current control loop in Matlab is processed at 10 ms (i.e. at a frequency of 100 Hz), which suggests that the speed of the control loop can not be counted on for highly dynamical tasks. However, for the balancing task, this frequency proves to be sufficient for now. Also there is an ongoing work on the implementaion of the controllers into C/C++.

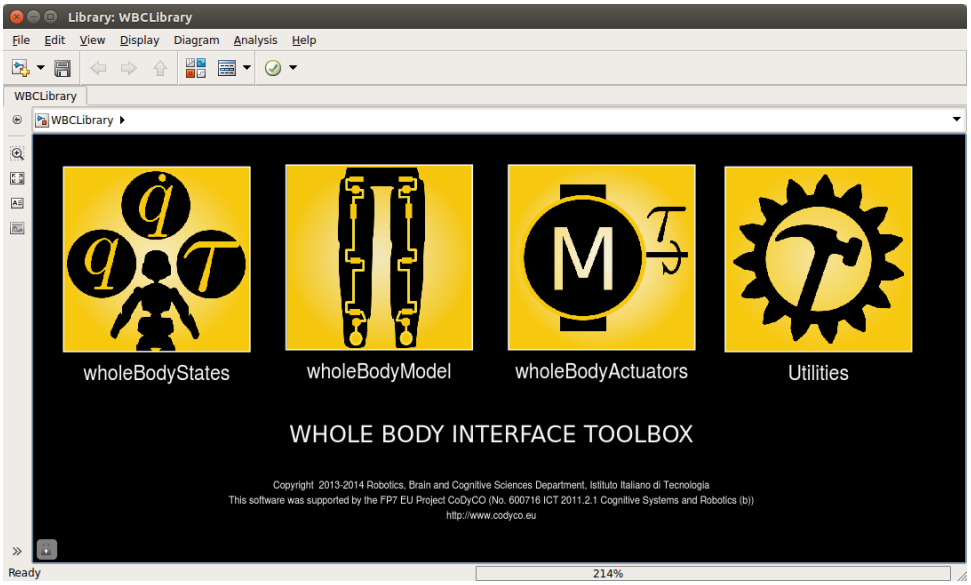

Figure 3.5: WBI Toolbox in Simulink

Chapter 4

Balancing with Force and Posture Control

In this chapter, the balancing task for a biped humanoid will be explained by related concepts and conditions. Then, a method which is based on the TSID (Task Space Inverse Dynamics) framework by [7] to balance a biped humanoid robot is explained. This method also forms the basis for the new controller with torque minimization which is the result of this project and will be explained in the next chapter.

4.1 Concepts and Conditions Related to Balancing Stability

To ensure stability during the balancing task, there are some conditions related to the feet contact forces (contact wrenches) that can be imposed as necessary while computing desired feet contact forces.

4.1.1 Friction Cone

Due to the friction coefficient between the surfaces of a planar contact, there is limit tangential force that can be supported by the contact until slippin occurs. If this happens with a humanoid robot, it would mean that the contact position changes and while the contact is not constrained in the directions it was previously, the state of the controller must change in order to operate successfully. For the balancing task, we want to keep the feet at their initial configurations at all times and since we will control the contact wrenches at the feet (from now on 'feet contact forces'), we must choose the desired feet contact forces in accordance with conditions of none slipping.

Friction cones can be used for this purpose, where the net force acting on the contact is desired to be kept inside a cone of friction, the cone angle of which is defined by the friction coefficient.

4.1.2 Support Polygon

The support polygon(SP) of a robot describes the minimal convex hull that can be formed around and thus including contact points of the robot.

4.1.3 Zero Moment Point (ZMP)

Zero moment point (ZMP) can be described as the tipping point where a planar contact's resultant wrench (force and moment) can be described with a force and moment couple where the net tangential moment is zero. If this point is outside the support polygon, meaning that the resultant contact wrench can be replaced by the same force and zero moment at a location

where it is outside the edges of a contact or the area supported in the case of multiple contacts, then there would be tipping over the edge closest to the ZMP. If zero moment point is on the edge of the support polygon, then

4.2 Formulation of a Force and Posture Controller

In this section, the force and posture controller based on the TSID framework proposed by [7] for balancing of the iCub is explained. It will be referred to as the previous controller in the next chapters.

4.2.1 Centroidal Momentum Derivative

When a floating-base system has external wrenches due to gravitation or contacts, according to D'Alembert's principle, the rate of change in angular and linear momenta of the system's center of mass are equal to the sum of external moments and forces, respectively. The rate in linear and angular momenta can be written as;

$$\begin{aligned} \dot{l}_{com} &= \vec{F_c} + m\vec{g} \\ \dot{k}_{com} &= (\vec{p_c} - \vec{p}_{com}) \times \vec{F_c} + \vec{M_c} \end{aligned} \tag{4.1}$$

where m is the total mass of the system, g is the gravitational acceleration, $\dot{l}_{com}$ and $\dot{k}_{com}$ denote the linear and angular momenta of the center mass respectively, $\vec{F_c}$ and $\vec{M_c}$ denote the force and moment components of the external wrenches at the contact points respectively, $\vec{p_c}$ is the position of the contact and $\vec{p}_{com}$ is the position of the center of mass.

From $\tau = \vec{r} \times \vec{f} = S(\vec{r})\vec{f}$, the equation 4.1 can written in a more compact form as;

$$\begin{bmatrix} \dot{l}_{com} \\ \dot{k}_{com} \end{bmatrix} = \begin{bmatrix} I & 0 \\ S(\vec{r_c}) & I \end{bmatrix} \begin{bmatrix} \vec{F_c} \\ \vec{M_c} \end{bmatrix} + \begin{bmatrix} m\vec{g} \\ 0 \end{bmatrix} \tag{4.2}$$

where $S(\cdot)$ denotes the skew symmetric matrix operator and $\vec{r_c} = \vec{p_c} - \vec{p}_{com}$ denote the relative position of the contact point with respect to the center of mass.

For the two contacts case, similarly;

$$\begin{bmatrix} \dot{l}_{com} \\ \dot{k}_{com} \end{bmatrix} = \begin{bmatrix} I & 0 & I & 0 \\ S(\vec{r_{c_1}}) & I & S(\vec{r_{c_2}}) & I \end{bmatrix} \begin{bmatrix} \vec{F_{c_1}} \\ \vec{M_{c_1}} \\ \vec{F_{c_2}} \\ \vec{M_{c_2}} \end{bmatrix} + \begin{bmatrix} m\vec{g} \\ 0 \end{bmatrix} \tag{4.3}$$

Then the rate of change in the spatial centroidal momentum of the system can be written as;

$$\dot{H}_{com} = Af_c + f_{grav} \tag{4.4}$$

where we defined,

- The vector containing the external wrenches on multiple contact points: $\begin{bmatrix} \vec{F_{c_1}} \\ \vec{M_{c_1}} \\ \vec{F_{c_2}} \\ \vec{M_{c_2}} \end{bmatrix} \in \mathbb{R}^{12}$ as *the spatial feet contact forces* $f_c = \begin{bmatrix} f_{left} \\ f_{right} \end{bmatrix}$ (we omit the vector notation for simplicity),

- The summation matrix for the forces and moments: $\begin{bmatrix} I & 0 & I & 0 \\ S(\vec{r_{c_1}}) & I & S(\vec{r_{c_2}}) & I \end{bmatrix} \in \mathbb{R}^{6\times12}$ as A matrix,
- The spatial force acting the system due to gravity: $\begin{bmatrix} m\vec{g} \\ 0 \end{bmatrix} \in \mathbb{R}^6$ as f_{grav}.

We know that for the center of mass, the linear momentum derivative can also be written as,

$$\dot{l}_{com} = m\ddot{p}_{com} \tag{4.5}$$

Hence if we can control the rate of change in the linear momentum of the center of mass, we can control the acceleration and therefore velocity and position of the center of mass. Additionally, since this is a humanoid, not a free-floating contactless robot, we are interested in minimum angular momentum. Hence we can define the desired rate of change in the spatial centroidal momentum to achieve desired accelerations of the center of mass and minimize the angular part as,

$$\dot{H}^* = \begin{bmatrix} \dot{l}^*_{com} \\ \dot{k}^*_{com} \end{bmatrix} = \begin{bmatrix} m\ddot{p}^*_{com} \\ -K_h k_{com} \end{bmatrix} \tag{4.6}$$

To follow a reference trajectory $p^d_{com}(t)$ with continuous first and second derivatives $\dot{p}^d_{com}(t)$, $\ddot{p}^d_{com}(t)$, the desired acceleration can be chosen as,

$$\ddot{p}^*_{com} = \ddot{p}^d_{com} - K_p(p_{com} - p^d_{com}) - K_d(\dot{p}_{com} - \dot{p}^d_{com}) \tag{4.7}$$

where p_{com} and $\dot{p}_{com}$ denote the position and the linear velocity of the center of mass in Cartesian space respectively, K_p and K_d are matrices with positive terms on the diagonal which denote the proportional and derivative gains on the position and velocity errors in three directions. The position of the center of mass is available via forward kinematics and the linear velocity of the center of mass can be acquired by, $\dot{p}^d_{com} = \begin{bmatrix} I_{3\times3} & 0_{3\times3} \end{bmatrix} J_{com}(q)^T v$, where J_{com} is the jacobian for the center of mass and v is the derivative of the generalized coordinates.

Finally, we can define the $\dot{H}^*$ which will help in achieving desired center of mass trajectory and minimizing angular momentum, from Equations 4.6 and 4.7 as,

$$\dot{H}^* = \begin{bmatrix} m\left(\ddot{p}^d_{com} - K_p(p_{com} - p^d_{com}) - K_d(\dot{p}_{com} - \dot{p}^d_{com})\right) \\ -K_h k_{com} \end{bmatrix} \tag{4.8}$$

4.2.2 Desired Contact Forces

Recall Equation 4.4 from the previous subsection defining the rate of change in spatial momentum of the center of mass of the robot,

$$\dot{H} = Af_c + f_{grav} \tag{4.9}$$

where f_c denotes the vector containing the contact forces f_{left} and f_{right} on left and right feet respectively; f_{grav} denotes the gravitational forces acting on the robot due to its mass; A denotes the matrix that relates the spatial feet contact forces to the rate of change in spatial centroidal momentum which is denoted as $\dot{H}$.

A desired trajectory of the center of mass can be achieved by setting a desired time derivative of the centroidal momentum and computing the desired contact forces accordingly. Hence, 4.4 needs to be solved with respect to f_c.

$$Af_c^* = \dot{H}^* - f_{grav} \tag{4.10}$$

For the case of both feet in contact with the ground, A matrix is a rectangular matrix ($m = 6$ rows by $n = 12$ columns with $n < m$) which does not have a full column rank and the system is indeterminate, meaning that there are infinite number of solutions to the problem above (4.10). We can formulate it as an optimization problem,

$$f_c^* = \arg\min_f ||\dot{H}^* - \dot{H}||^2 \tag{4.11}$$

which is subject to Equation 4.9 and the solution of this optimization is given by,

$$f_c^* = A^+(\dot{H}^* - f_{grav}) + (I - A^+A)f_0 \tag{4.12}$$

where f_0 is an arbitrary vector that is projected into the nullspace of f_c^* and A^+ is the Moore-Penrose pseudo inverse of A. This solution gives a set of desired feet contact forces f_c^* which would result in the desired rate of change in the centroidal momentum $\dot{H}^*$. If the arbitrary vector f_0 is taken as a zero vector, we get,

$$f_c^* = A^+(\dot{H}^* - f_{grav}) \tag{4.13}$$

which gives the minimum norm solution for f_c.

4.2.3 Desired Joint Torques

Recall the equations of motion of the humanoid robot from Section 2.4 as,

$$\begin{aligned} M(q)\dot{v} + h(q,v) &= S^T\tau + J_c^T(q)f_c \\ J_c(q)\dot{v} + \dot{J}_c v &= 0 \end{aligned} \tag{4.14}$$

where $M(q)$ is the inertial mass matrix, $h(q,v)$ is the vector containing the bias forces, S is the selector matrix $\begin{bmatrix}0, & I_{25x25}\end{bmatrix}$, J_c is the jacobian matrix for the contact points and f_c is the vector containing contact forces.

While omitting the dependencies on q and v for simplicity, $\dot{v}$ from Equation 4.14 can be obtained as,

$$\dot{v} = M^{-1}(S^T\tau + J_c^T f_c - h) \tag{4.15}$$

then substituting above expression into the constraint part of Equation 4.14 and leaving τ on one side,

$$J_c M^{-1} S^T \tau = J_c M^{-1}(h - J_c^T f_c) - \dot{J}_c v \tag{4.16}$$

Due to the fact that $J_c M^{-1} S^T$ does not have full column rank, the above expression does not have a unique but a set of solutions for τ , similar to Equation 4.10. Hence, it can be solved for τ as,

$$\begin{aligned} \tau^* &= \Lambda^+ \left(J_c M^{-1}(h - J_c^T f_c^*) - \dot{J}_c v \right) + N_\tau \tau_0 \\ \text{with} \quad \Lambda &= J_c M^{-1} S^T \ , \ N_\tau = (I - \Lambda^+\Lambda) \end{aligned} \tag{4.17}$$

where Λ^+ is the Moore-Penrose pseudo inverse of Λ, N_τ is the null-space projector matrix, τ_0 is an arbitrary vector.

In the TSID framework by [7], the task of controlling the forces at the contacts is done with the highest priority and any secondary tasks such as motion and posture can be done by exploiting the null-space of the primary control task. The above expression for the desired joint torques is derived such that the forces at the contacts match closely to the desired values and the null-space of this solution can be exploited bu setting the arbitrary vector τ_0 for any secondary task, which in this case will the postural task.

With the task of controlling the contact forces at the feet, we ensure the controlled movement and position of the center of mass of the robot. However, this task can be achieved by infinitely many joint configurations. Hence, to keep joint position at acceptable regions, we have to impose a postural task to make each joint withstand the external forces due to the contacts and gravity and stay close to a desired position. At the joint level, this can be done by considering the generalized forces and forces due to the contacts acting on the joints along with a feedback on position to impose impedance for the joints as they move away from their desired values which form the desired posture for the robot. Hence the arbitrary vector τ_0, that will be projected onto the null-space of the force control task, can be defined as;

$$\tau_0 = S\left(g(q) - J_c(q)^T f_c^*\right) - K_{imp}(q_j - q_0) \tag{4.18}$$

where S is transpose of the selector matrix as in Equation 4.14 to choose only joint related terms, $g(q)$ is the vector of generalized forces,$J_c(q)$ is the contact Jacobian, f_c is the feet contact forces, K_{imp} is a diagonal matrix containing the gains for the impedance behavior for each joint and q_0 is the desired joint configuration which is fixed as the initial configuration for now.

4.2.4 The Control Law

From the derivation in the last three subsections, the force and posture control law can be concisely written as (From Equations 4.8, 4.13, 4.17, 4.18),

$$\tau^* = \Lambda^+\left(J_c M^{-1}(h - J_c^T f_c^*) - \dot{J}_c v\right) + N_\tau \tau_0$$

with,

$$\begin{aligned}
\Lambda &= J_c M^{-1} S^T \\
N_\tau &= (I - \Lambda^+\Lambda) \\
f_c^* &= A^+(\dot{H}^* - f_{grav}) \\
\dot{H}^* &= \begin{bmatrix} m\left(\ddot{p}_{com}^d - K_p(p_{com} - p_{com}^d) - K_d(\dot{p}_{com} - \dot{p}_{com}^d)\right) \\ -K_h k_{com} \end{bmatrix} \\
\tau_0 &= S\left(g(q) - J_c(q)^T f_c^*\right) - K_{imp}(q_j - q_0)
\end{aligned} \tag{4.19}$$

This controller computes the joint torques necessary in achieving the desired contact forces in order to control the position of the center of mass for a given trajectory while as a secondary task, it keeps the posture of the robot close to a desired configuration. However, it is not yet complete in the sense that it does not include conditions of the contact forces to ensure stability of the balancing task. This will be addressed in the formulation for a controller which is the outcome of this project. From now on, the controller described in 4.19 will be referred to as the previous controller. Simulations and experiments done with this controller will also be presented in the next chapter to help the analysis and comparison with the new controller which comes next.

Chapter 5

Balancing Controller with Torque Minimization

In this chapter, we present the main outcome of this project which is a balancing force and posture controller with torque minimization achieved by a new formulation as an optimization problem where additional constraints can be imposed. Firstly, we start with the problem definition in Section 5.1 where we describe the tasks we want to address with this controller. Following that is the derivation for the new controller in Section 5.2, which has the controller described in the previous chapter as its starting point (from now on 'the previous controller'). After the formulation of the new controller, we describe the implementation of the controller to simulations and the real robot in Section 5.3. Finally, we present and discuss the results from simulations and experiments in Section 5.4 and Section 5.5, respectively.

5.1 Problem Definition

The previous balancing controller exploits the feet contact forces to change the centroidal momentum of the robot which results in a desired acceleration of the center of mass as it is presented in the previous chapter in Section 4.2. However, due to the fact that the problem of choosing the feet contact forces that result in the desired centroidal momentum derivative does not have a unique but a set of solutions. Exploiting it can allow us to take into account any desired constraints at the force and torque level and it can help towards the minimization of the desired joint torques.

The objective of this project is to improve upon the previous state of the balancing controller by implementing optimization to minimize joint torques and along the process adding constraints on contact forces, joint torques and other additional means if necessary, to ensure that the given motion tasks on the center of mass (CoM) of the robot can be achieved. In the following subsections, these tasks are defined and how they are going to be addressed in the derivation of the new formulation for the controller will be discussed briefly. Afterwards, in Section 5.2, the new controller will be explained in detail.

5.1.1 Balancing Task

The balancing task for the robot will be for the case of both feet touching the ground while the contacts are assumed to be rigid. In order to keep its balance, the robot feet where the contacts are present should not slip, lift or rotate in any direction. The conditions that ensure such stability for the balancing task can be included as constraints on the feet contacts forces staying within a friction cone which ensure no slipping, and constraints on the zero moment

point (ZMP)[Vukobratovic and Joricic 1969] resulting from the feet contact forces being inside the support polygon of the robot which ensure no tipping over. In order to ensure that the balancing is achieved these constraints must be included in the computation for the desired contact forces.

5.1.2 Postural Task

Keeping the center of mass of the robot (CoM) at a desired position can be achieved in an infinite number of configurations with a floating-base system which is a subclass of underactuated systems. Hence to ensure internal stability of the joints and keeping a meaningful and useful whole body configuration at all times, a postural task needs to be implemented. This is implemented in the new controller in the same way as it is done in the previous version of the controller (Equation 4.19), by exploiting the nullspace of the task of generating the desired feet contact forces as described in Section 4.2.3.

5.1.3 Torque Minimization and constraints

Since there is no unique solution for the feet contact forces that result in achieving the balancing and postural tasks, this free choice can be exploited to formulate the desired joint torques as the objective function of an optimization problem. The solution to such a problem will result in desired feet contact forces that minimize the desired joint torques. Additionally, the constraints on the feet contact forces and the joint torques will be added to the optimization problem to ensure a desired execution for the balancing task.

5.2 Controller Design

5.2.1 Desired Contact Forces

Recall the derivation of the desired contact forces in 4.2.2 for the previous controller (4.19), where the set of solutions to the problem of computing the contact forces to achieve the desired centroidal momentum derivative,

$$\dot{H} = Af_c + f_{grav} \tag{5.1}$$

is given as,

$$f_c^* = A^+(\dot{H}^* - f_{grav}) + (I - A^+A)f_0 \ . \tag{5.2}$$

where f_c^* denote the vector containing the external wrenches at the contact points (*feet contact forces*), $\dot{H}^*$ denotes the spatial centroidal momentum derivative containing the linear and angular momenta derivative of the center of mass, A is the matrix relating the forces and moments at the contact points to the momenta derivative, A^+ is the Moore-Penrose pseudo inverse of A, f_{grav} is the gravitation force acting on the center of mass, $(I - A^+A)$ is the nullspace projector matrix and f_0 is an arbitrary vector.

The above equation defines the set of possible choices for the contact forces that solves to problem of achieving a desired rate of change in the spatial centroidal momentum to control the center of mass for a given trajectory. Although there seems to be many solutions, there are conditions, as described in Section 4.1, that must be considered in the process, to ensure balancing stability and for this, we can exploit the choice of the the arbitrary vector f_0 in Equation 5.2 to make sure that the contact forces are defined with respect to necessary conditions.

5.2.2 Desired Joint Torques

Recalling the derivation of the desired joint torques in 4.2.3 for the previous controller (4.19), where the set of solutions to the problem of computing the joint torques that results in achieving desired contact forces,

$$J_c M^{-1} S^T \tau = J_c M^{-1}(h - J_c^T f_c) - \dot{J}_c v \tag{5.3}$$

is given as,

$$\begin{aligned} \tau^* &= \Lambda^+ \left(J_c M^{-1}(h - J_c^T f_c^*) - \dot{J}_c v \right) + N_\tau \tau_0 \\ \text{with} \quad & \Lambda = J_c M^{-1} S^T \ , \ N_\tau = (I - \Lambda^+ \Lambda) \end{aligned} \tag{5.4}$$

where Λ^+ is the Moore-Penrose pseudo inverse of Λ, N_τ is the null-space projector matrix, τ_0 is an arbitrary vector.

We will exploit the null-space projection to impose the secondary task of keeping the robot posture close to a desired configuration, with the same choice of the arbitrary vector as τ_0 as in the previous controller (4.19),

$$\tau_0 = S\left(g(q) - J_c(q)^T f_c^*\right) - K_{imp}(q_j - q_0) \tag{5.5}$$

For simplicity in the derivation which will follow, we group the terms in τ_0 as,

$$\tau_0 = m_{\tau_0} f_c^* + n_{\tau_0} \tag{5.6}$$

where,

$$m_{\tau_0} = -S J_c(q)^T \quad \text{and} \quad n_{\tau_0} = S^T g(q) - K_{imp}(q_j - q_0)$$

Substituting the expression for the desired feet contact forces from Equation 5.2 into the Equation 5.4 with Equation 5.6,

$$\begin{aligned} \tau^* = \Lambda^+ & \left(J_c M^{-1} \left(h - J_c^T \left(A^+(\dot{H}^* - f_{grav}) + (I - A^+ A) f_0 \right) \right) - \dot{J}_c v \right) \\ & + N_\tau \left(m_{\tau_0} \left(A^+(\dot{H}^* - f_{grav}) + (I - A^+ A) f_0 \right) + n_{\tau_0} \right) \end{aligned} \tag{5.7}$$

By grouping the multipliers of f_0 in τ^*,

$$\begin{aligned} \tau^* &= \left[\left(N_\tau m_{\tau_0} - \Lambda^+ J_c M^{-1} J_c^T \right) (I - A^+ A) \right] f_0 \\ &\quad + \Big[\left(N_\tau m_{\tau_0} - \Lambda^+ J_c M^{-1} J_c^T \right) \left(A^+ \left(\dot{H}^* - f_{grav} \right) \right) \\ &\qquad + N_\tau n_{\tau_0} + \Lambda^+ \left(J_c M^{-1} h - \dot{J}_c v \right) \Big] \\ \tau^* &= X_\tau f_0 + Y_\tau \end{aligned} \tag{5.8}$$

with,

$$X_\tau = \left(N_\tau m_{\tau_0} - \Lambda^+ J_c M^{-1} J_c^T\right)\left(I - A^+ A\right)$$
$$Y_\tau = \left(N_\tau m_{\tau_0} - \Lambda^+ J_c M^{-1} J_c^T\right)\left(A^+\left(\dot{H}^* - f_{grav}\right)\right) + N_\tau n_{\tau_0} + \Lambda^+\left(J_c M^{-1} h - \dot{J}_c v\right)$$

Let V be a function related to the norm of the joint torques that is to be minimized with respect to the arbitrary vector f_0,

$$V = \frac{1}{2}||\tau||^2 \tag{5.9}$$

Substituting 5.8 into 5.9,

$$V = \frac{1}{2}||X_\tau f_0 + Y_\tau||^2$$

$$= \frac{1}{2}(f_0^T X_\tau^T + Y_\tau^T)(X_\tau f_0 + Y_\tau)$$

$$V = \frac{1}{2}\left(f_0^T X_\tau^T X_\tau f_0 + 2Y_\tau^T X_\tau f_0 + Y_\tau^T Y_\tau\right) \tag{5.10}$$

The square of the joint torque norm is to be minimized with respect to f_0, so by extracting the terms from the expression 5.10 which are related to the arbitrary vector f_0,

$$V_{f_0} = \frac{1}{2} f_0^T \left(X_\tau^T X_\tau\right) f_0 + \left(X_\tau^T Y_\tau\right)^T f_0 \tag{5.11}$$

can be recognized as a candidate objective function of an optimization problem where we minimize the joint torques, with respect to the arbitrary vector f_0, which will be used to compute the desired contact forces and related joint torques. Additionally, we want to impose conditions on the desired contact forces such that the balancing stability is ensured. Since these conditions are in terms of the feet contact forces, they can be imposed on the optimization problem as linear inequality constraints for the solution of f_0. Hence the optimization problem can be formulized as a quadratic program the quadratic and linear terms of which are $X_\tau^T X_\tau$ and the $X_\tau^T Y_\tau$, respectively, as follows,

$$\begin{aligned} \min_{f_0} \quad & V(f_0) \;= \frac{1}{2} f_0^T \left(X_\tau^T X_\tau\right) f_0 + \left(X_\tau^T Y_\tau\right)^T f_0 \\ \text{subject to} \quad & A_{ineq} f_0 \leq b_{ineq} \end{aligned} \tag{5.12}$$

where the inequality constraints,

$$A_{ineq} = \begin{bmatrix} A_{fcone} \\ A_{zmp} \end{bmatrix}$$

will be formulated next.

Inequality Constraints of Feet Contact Forces

Now the conditions of feet contact forces, discussed in Section 4.1 are going to be defined in terms of f_0 to be included in the quadratic program as described by (5.12).

Friction cone can be considered in the optimization as linear inequalities obtained by using a numerical approximation of the cone via a set of segments and using the angular offsets and coefficients we can get the linear inequalities on the forces as;

$$A_{ineq_f} f \leq b_{ineq_f},$$

where,

$$\begin{aligned} A_{ineq_f} &= \begin{bmatrix} -\alpha_i & 1 & -o_i c_f & 0 & 0 & 0 \\ \dots & \dots & \dots & 0 & 0 & 0 \\ \alpha_i & -1 & o_i c_f & 0 & 0 & 0 \\ \dots & \dots & \dots & 0 & 0 & 0 \end{bmatrix} \\ b_{ineq_f} &= 0 \end{aligned} \tag{5.13}$$

where i denote the subscript for each approximated point, α_i denotes the angular coefficient of the line between ith and $(i+1)$th approximation points and c_f denotes the friction coefficient. A_{ineq_f} is at each contact where and for two contacts,

$$A_{ineq_{f_c}} = \begin{bmatrix} A_{ineq_f} & 0 \\ 0 & A_{ineq_f} \end{bmatrix} \qquad b_{ineq_{f_c}} = \begin{bmatrix} b_{ineq_f} \\ b_{ineq_f} \end{bmatrix} \tag{5.14}$$

can be used.

For the ZMP conditions we use a simple set of constraints related to the rectangular approximation of the feet and tipping momments necessary for those dimension.

5.3 Simulation Environment

Matlab[1] is used to simulate the balancing task. For this, we used mex-wholebodymodel project[8] which is a mex-C/C++ interface to the WBI (Whole Body Interface) components and which allows obtaining forward dynamics from Matlab. The forward dynamics is integrated over the duration of the simulation by the Matlab function 'ode15s'. The progress of the states and the contact forces are recorded and these results are then used in the visualizer part to animate the hole body motion. In the following two subsections, this process is explained in more detail.

5.3.1 Forward Dynamics

From Chapter 2, recalling the equations of motion of a constrained floating base robot (2.10) with an additional term,

$$\begin{aligned} &M(q,v)\dot{v} + h(q,v) - J_c(q)^T f_c - Q_f = S^T \tau^* \\ &s.t. \quad J_c(q)\dot{v} + \dot{J}_c(q,v)v = 0 \end{aligned} \tag{5.15}$$

where Q_f is added to add frictional effects, disturbances etc. at the joint space to evaluate the performance of the controller more realistically, if desired.

Taking the inverse of the mass matrix in the first part of 5.15 to obtain $\dot{v}$,

$$\dot{v} = M^{-1}\left(-h + J_c^T f_c + S^T \tau^* + Q_f\right) \tag{5.16}$$

[1]Matlab R2014a with student licence under TU/e

And substituting the above expression for $\dot{v}$ into the constraint part of 5.15,

$$J_c M^{-1}\left(-h + J_c^T f_c + S^T \tau^* + Q_f\right) + \dot{J}_c v = 0 \tag{5.17}$$

Then we obtain the contact forces due to the contacts, which are assumed to be rigid, as,

$$f_c = \left(J_c M^{-1} J_c^T\right)^{-1}\left(J_c M^{-1}(h - S^T\tau^* + Q_f) - \dot{J}_c v\right) \tag{5.18}$$

For the integration of the forward dynamics, we form the vector,

$$x = \begin{bmatrix} q \\ v \end{bmatrix} \tag{5.19}$$

with,

$$q = \begin{bmatrix} p_{base} \\ Q_{base} \\ q_j \end{bmatrix} \in \mathbb{R}^{3+4+25} \qquad v = \begin{bmatrix} \dot{p}_{base} \\ w_{base} \\ \dot{q}_j \end{bmatrix} \in \mathbb{R}^{3+3+25}$$

where p_{base} is the position of the floating base, Q_{base} is the orientation of the floating base in terms of quaternions, q_j is the joint angular positions, $\dot{q}_j$ is the joint angular velocities and $\dot{p}_{base}$ and w_{base} are the linear and the angular velocities of the floating-base respectively.

By forming the derivative of x as below, note that $\dot{q} \neq v$,

$$\dot{x} = \begin{bmatrix} \dot{q} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} \dot{p}_{base} \\ \dot{Q}_{base} \\ \dot{q}_j \\ \dot{v} \end{bmatrix} \tag{5.20}$$

where $\dot{Q}_{base}$ is obtained by calculating the quaternion derivative from w_{base} and Q_{base} as,

$$\dot{Q}_{base} = \frac{1}{2}\begin{bmatrix} 0 & -w_{base}^T \\ w & -S(w_{base}) \end{bmatrix} Q_{base} + (1 - |Q_{base}|)Q_{base}$$

where also a numerical trick is used to make the norm of the Q_{base} equal to 1.

Substituting Equation 5.18 and Equation 5.16 into Equation 5.20,

$$\dot{x} = \begin{bmatrix} \dot{p}_{base} \\ \dot{Q}_{base} \\ \dot{q}_j \\ M^{-1}\left(-h + J_c^T\left(J_c M^{-1} J_c^T\right)^{-1}\left(J_c M^{-1}(h - S^T\tau^* + Q_f) - \dot{J}_c v\right) + S^T\tau^* + Q_f\right) \end{bmatrix} \tag{5.21}$$

To create the forward dynamics, all the other necessary terms such as M, h, J_c and $\dot{J}_c v$ in the forward dynamics part and g, H, J_{com}, p_{com}, p_{right} required specially for the controller to compute desired joint torques, are obtained by using mex-wholebodymodel interface.

We can then integrate $\dot{x}$ over time to evaluate the states of the floating base robot with rigid contacts as described in 5.15 and desired torques obtained from the control law described by 5.8 where the arbitrary vector f_0 is obtained by the optimization problem 5.12. The integration

function 'ode15s', which solves stiff differential equations and differential algebraic equations with variable order method, seems to be the most efficient one to use compared to fixed order and non-stiff solvers, for our purpose.

After the integration, the states at each time step is stored and the movement of the robot along with other desired results can be visualized and plotted.

5.3.2 Matlab Visualizer

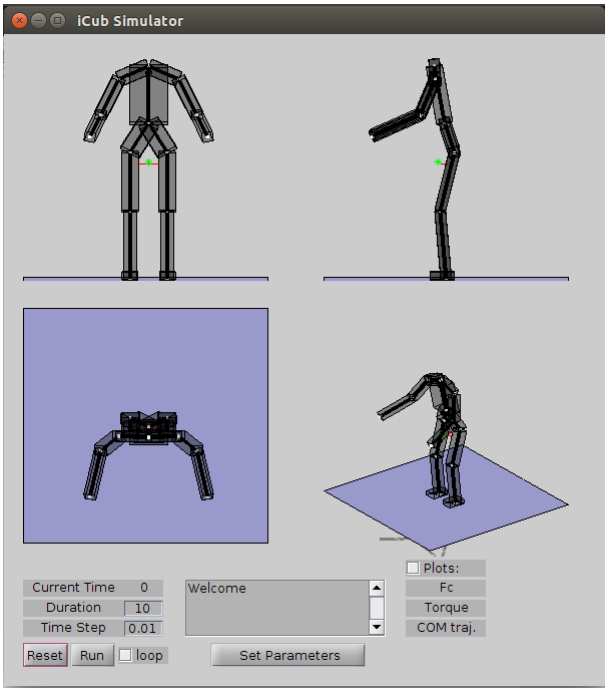

Figure 5.1: Matlab Visualizer for simulating iCub

The visualizer made in Matlab consists of a graphical user interface (GUI) which can be used to,

- Run simulations,
- Visualize the results on a robot figure,
- Change simulation parameters such as duration, time step etc. ,
- Edit the m-file where other parameters such as trajectory, controller gains, initial configuration etc. are defined and can be edited.

The visualizer along with the forward dynamics and controller implementation are all included in mex-wholebodymodel project[2] which is open-source.

[2]https://github.com/robotology-playground/mex-wholebodymodel

5.4 Results

In this section, several results from simulations and experiments will be presented. Completion of the primary task of moving the center of mass of the robot with a certain trajectory will be investigated through the error between desired and achieved trajectories. Task of minimization of the torques will be investigated by the norm of the joint torques. In addition, to see the influence of torque minimization on the feet contact forces, they will be investigated and the effect on internal torques will be discussed.

5.4.1 Simulations

To observe the minimization of joint torques with the new formulation, the results of three simulations with the same tasks but different controllers will be presented in this section.

The task for the center of mass of the robot to follow is a sinusodial trajectory with an amplitude of 5 cm to the sides (y-direction) and a frequency of 1 rad/s (duration of approximately 6.28 seconds per cycle). Firstly, the previous controller without minimization using f_0; secondly, the current controller with no constraints on optimization and lastly the current controller with constraints on the contact forces is tested and results are presented as below.

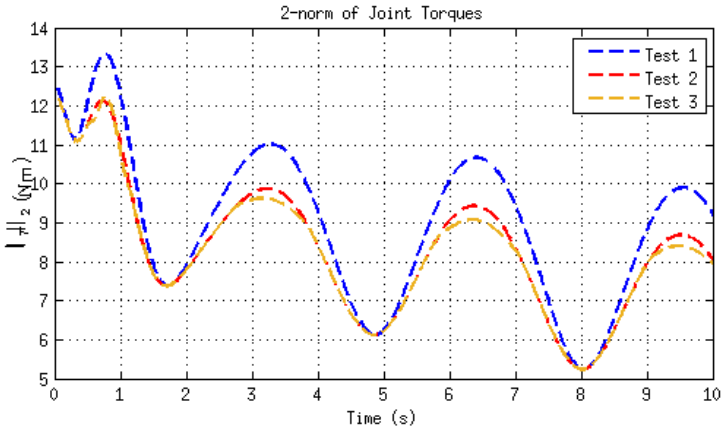

Figure 5.2

Simulation with the previous controller (without optimization)

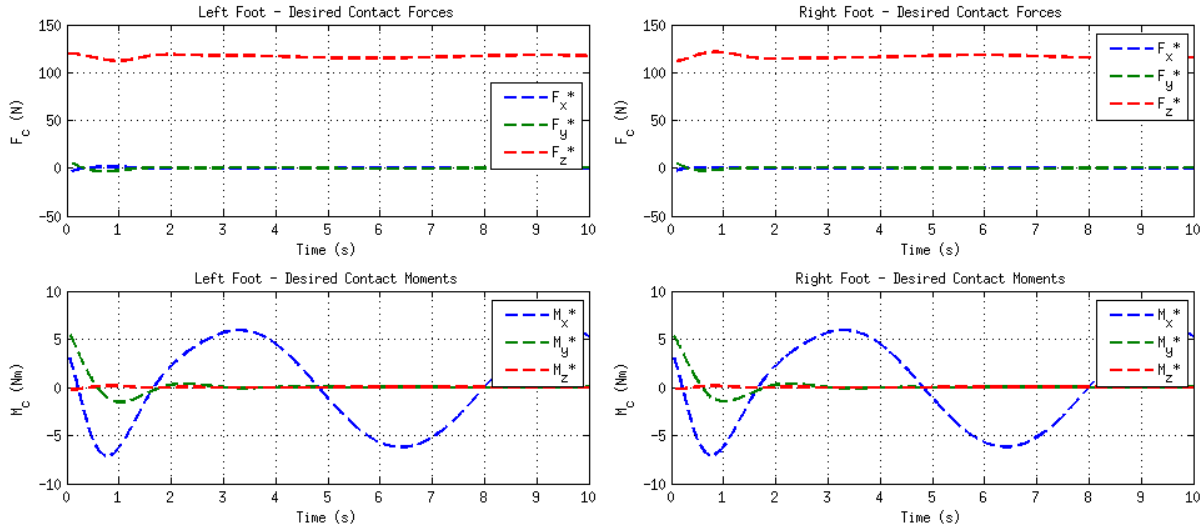

Figure 5.3: **Simulation with no QP** - Feet Contact Forces - left foot forces(upper left) & left foot moments(lower left) & right foot forces(upper right) & right foot forces(lower right)

From Figure 5.3, it can be seen that during the motion of the center of mass which is periodic from side to side, there is no significant difference between left and right foot contact forces. Although the vertical projection of the center of mass moves towards the feet while inside the support polygon, the desired contact forces are such that similar vertical forces are generated and it suggest that the controller results in extra internal joint torques to generate such contact forces even if it is close to being able to support itself on one foot. This can clearly be seen in the right part of Figure 5.4, where the robot is seen leaning to the sides as a result of the given task, but the desired resultant linear forces on the feet do not differ based on the projection of CoM. Additionally, it should be noted that the robot is able to perform the task of CoM trajectory with minor errors as seen in the lower left part of Figure 5.4 and the impedance task was also achieved which kept the posture of the robot close to its initial configuration. The norm of the joint torques of this test is presented in Figure 5.2 by the blue dashed line.

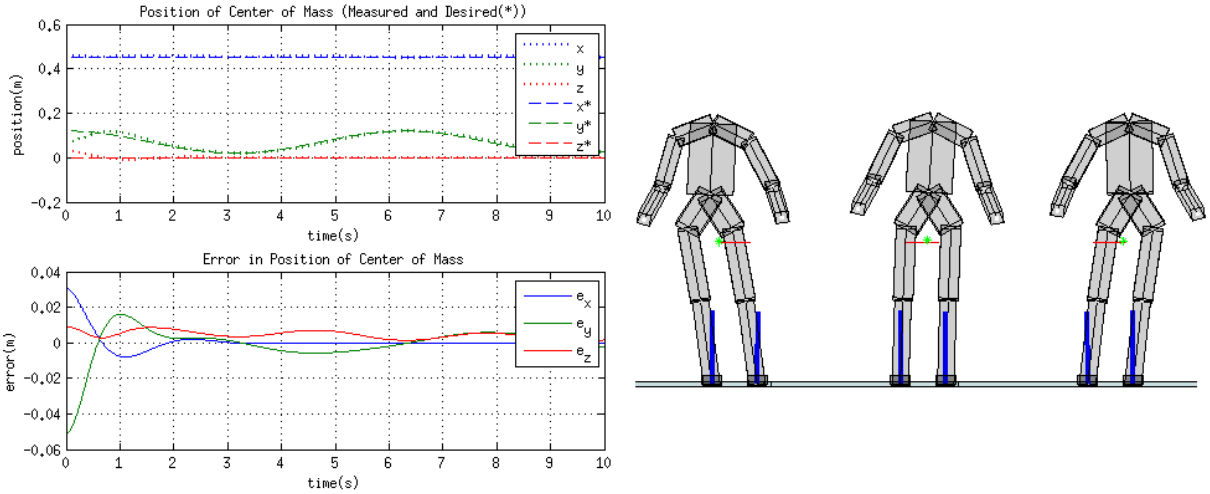

Figure 5.4: **Simulation with no QP** - Desired (dashed) and Measured (dotted) Center of Mass trajectories(upper left) & Error in CoM trajectory(lower left) & Snapshots of the robot from the test with contact forces as blue arrows(right)

Simulation with the new controller (with optimization)

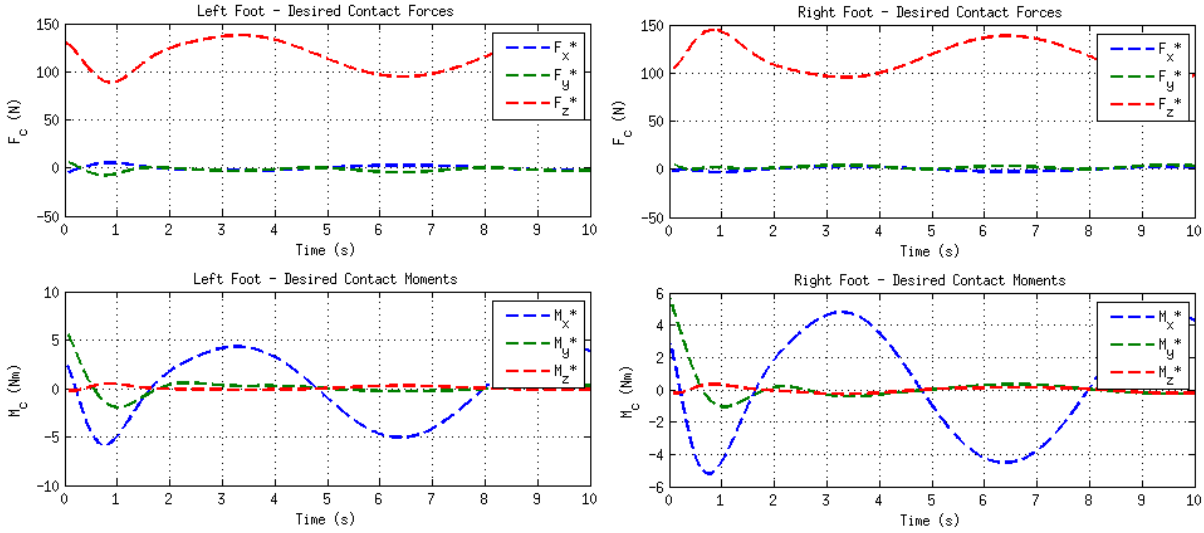

Figure 5.5: **Simulation with QP** - Feet Contact Forces - left foot forces(upper left) & left foot moments(lower left) & right foot forces(upper right) & right foot forces(lower right)

With the new controller, as explained in Section 5.2, desired feet contact forces are based on the minimization of joint torques. As a result of the new formulation, difference between right and left feet contact forces is observed as seen in Figure 5.5, as opposed to the previous test's results in Figure 5.3. Comparing also with the trajectory plot in the Figure 5.6, the desired vertical forces on both feet are oppositely harmonious with the periodic motion of CoM. In other words, as the robot leans to one side, the new controller results in a desired vertical force on that corresponding foot greater than the other one, as it can also be seen by the blue arrows depicting net feet contact forces in Figure 5.6's right side. In Figure 5.2, the joint torque norm of this test can be found (dashed red) and it can clearly be seen that the minimization of the joint torques is achieved. It is also interesting to note that the instances when both norms get close to each other are when the robot's CoM position is crossing the center of both feet and the minimization factor is maximum when the CoM is at the extremes.

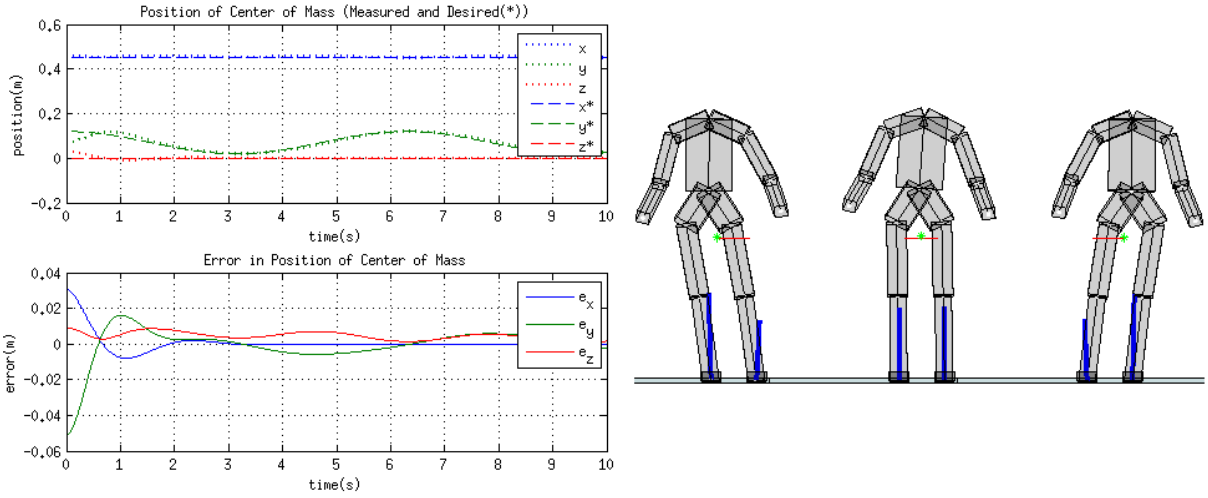

Figure 5.6: **Simulation with QP** - Desired (dashed) and Measured (dotted) Center of Mass trajectories(upper left) & Error in CoM trajectory(lower left) & Snapshots of the robot from the test with contact forces as blue arrows(right)

Simulation with the new (with optimization and a strict constraint)

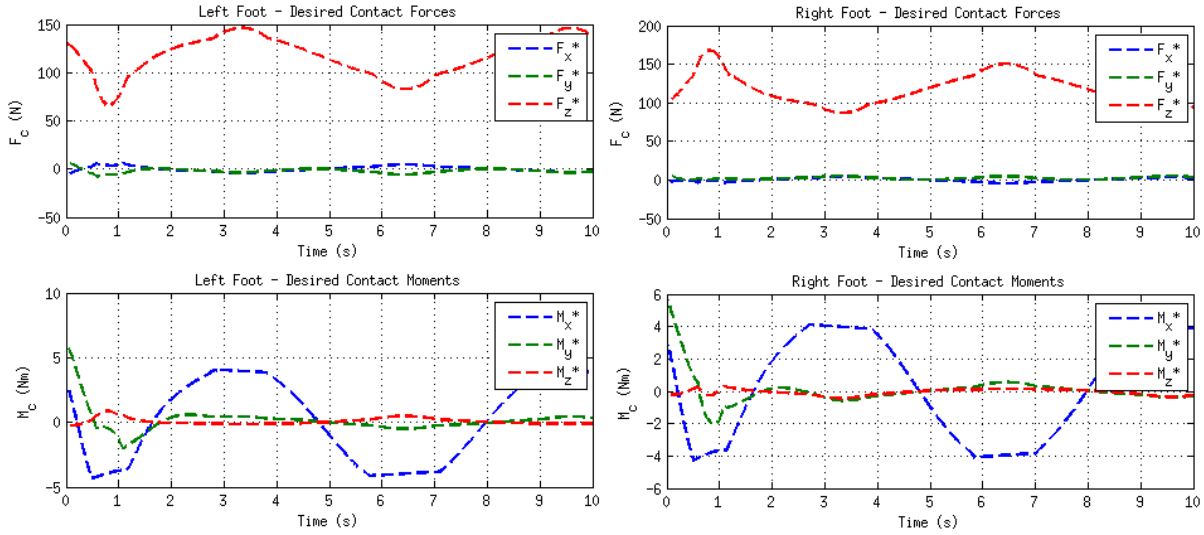

Figure 5.7: **Simulation with QP and a strict constraint** - Feet Contact Forces - left foot forces(upper left) & left foot moments(lower left) & right foot forces(upper right) & right foot forces(lower right)

To clearly observe how imposing a constraint will affect the completion of the task when it becomes active, a strict constraint on the desired contact moment in x-direction (M_x*) along with other constraints formulated in Section 5.2.2 is introduced. With the use of quadratic programming, minimal joint torques and feet contact forces that satisfy these constraints can be found if the constraints allow such solution. As seen in Figure 5.7, a limit of 4 Nm on $|M_x^*|$ is achieved and the motion task along with impedance task (Figure 5.8) is achieved in a similar way to the previous tests. However, relying on QP for a solution under such a strict constraint is not desirable for it increases the computational load in an unpredictable way for each control loop. Still this test is important, because regarding the side stepping and walking tasks in the future, an attempt to find a solution to achieve the task and which is also satisfied by the constraints may allow to anticipate when a recovery action such as a side step will be necessary. Additionally, in Figure 5.2, a strict constraint becoming active seems to minimize the torques although it is actually due to the change in achieved CoM trajectory which is slightly different.

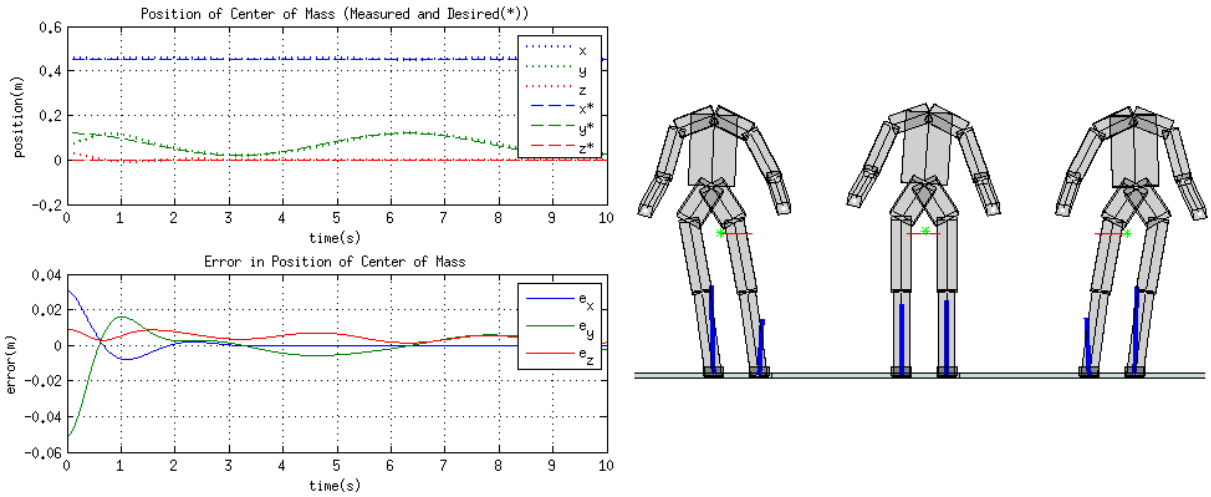

Figure 5.8: **Simulation with QP and a strict constraint** - Desired (dashed) and Measured (dotted) Center of Mass trajectories(upper left) & Error in CoM trajectory(lower left) & Snapshots of the robot from the test with contact forces as blue arrows(right)

5.4.2 Experiments on the Robot (iCubGenova01)

Experiments are done on the robot iCubGenova01 (*'black iCub'*) which is currently in IIT, Genova, Italy. Here, the results from tests with two kinds of trajectory tasks for the CoM carreid out with the previous and the new controller are presented.

The first task is keeping the CoM at the initial configuration (constant trajectory). This task is tested with the previous and the new controller. Feet contact forces and CoM trajectories are presented seperately whereas the the joint torque norms are presented together. It should be noted that for about 10 seconds initially, the robot is not disturbed and after about 10 seconds from the start, external disturbances are introduced in the order of a downward push on the left arm, a downward push on the right arm, a downward push on both arms and a backwards push on the torso. The measured forces and torques are included in the results to compare them with the desired values eventhough there is no feedback law relating them in the controller. This is done to test the robustness of such a force and posture controller which does not use the error between desired and measured forces and torques and this also allows to evaluate motor joint torque control and the whole body estimation of joint torques and contact forces.

Figure 5.9: The Black iCub - *iCubGenova01*

The second task is a periodic trajectory given to the CoM, but the parameters defining the task for the new controller is slightly different. During tests, it was seen that with the previous controller, the robot is not able to achieve a movement of 3 cm for its CoM to the sides, even if it is a minimum-jerk trajectory generated using the results of [Pattacini 2010]. Hence we were able to test periodic motions with amplitudes only smaller than 3 cm and we tested the previous controller with 2.5 cm of amplitude and with a frequency of 0.3 Hz (duration of approximately 3.3 seconds per cycle). Then for the new controller we present the results of a task with an amplitude of 4 cm with a frequency of 0.25 Hz.

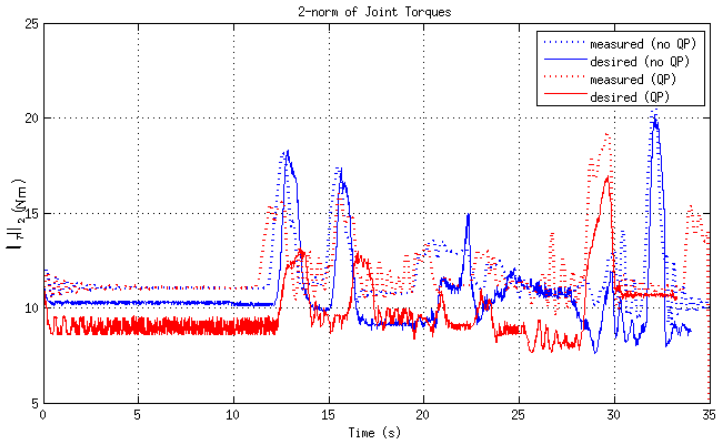

Figure 5.10

Experiment with the previous controller (no optimization) for constant trajectory

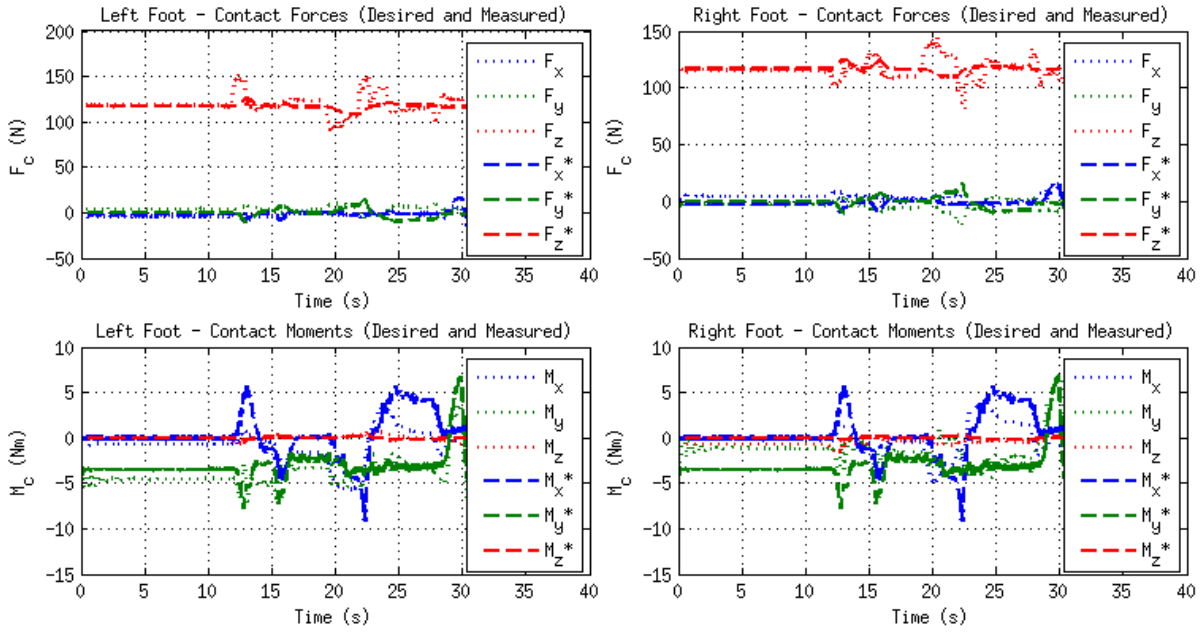

Figure 5.11: **Experiment with no QP** - Desired (dashed) and Measured (dotted) Feet Contact Forces - left foot forces(upper left) & left foot moments(lower left) & right foot forces(upper right) & right foot forces(lower right)

First 10 seconds in Figures 5.11 and 5.12, the robot keeps its CoM at its initial position with minor errors, with the previous controller. To test if it is also robust to external disturbances, after 10 seconds, the robot is disturbed from its arms and torso. After each disturbance, the measured contact forces are observed to converge to their desired values and the error CoM trajectory is minimized after the external disturbances. Eventhough the controller does not have any information about these newly introduced contacts, it reacts in a safe manner just to achieve desired contact forces and joint torques rather than positions, which is an advantage of using such force and postural control framework. Additionally in Figure 5.15, the norm of the desired and the estimated joint torques from this test is depicted with the blue line and the blue dotted line, respectively. Mismatch is seen between the desired and estimated torques and currently the estimation process is still under development.

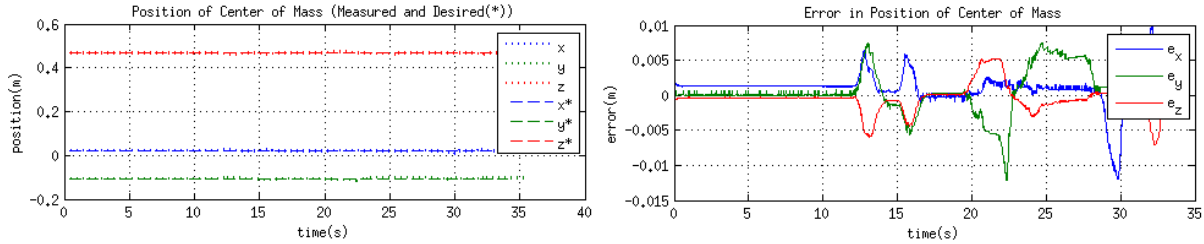

Figure 5.12: **Experiment with no QP** - Desired (dashed) and Measured (dotted) Center of Mass trajectories(upper left) & Error in CoM trajectory(lower left) & Snapshots of the robot from the test with contact forces as blue arrows(right)

Experiment with the new controller (with optimization) for constant trajectory

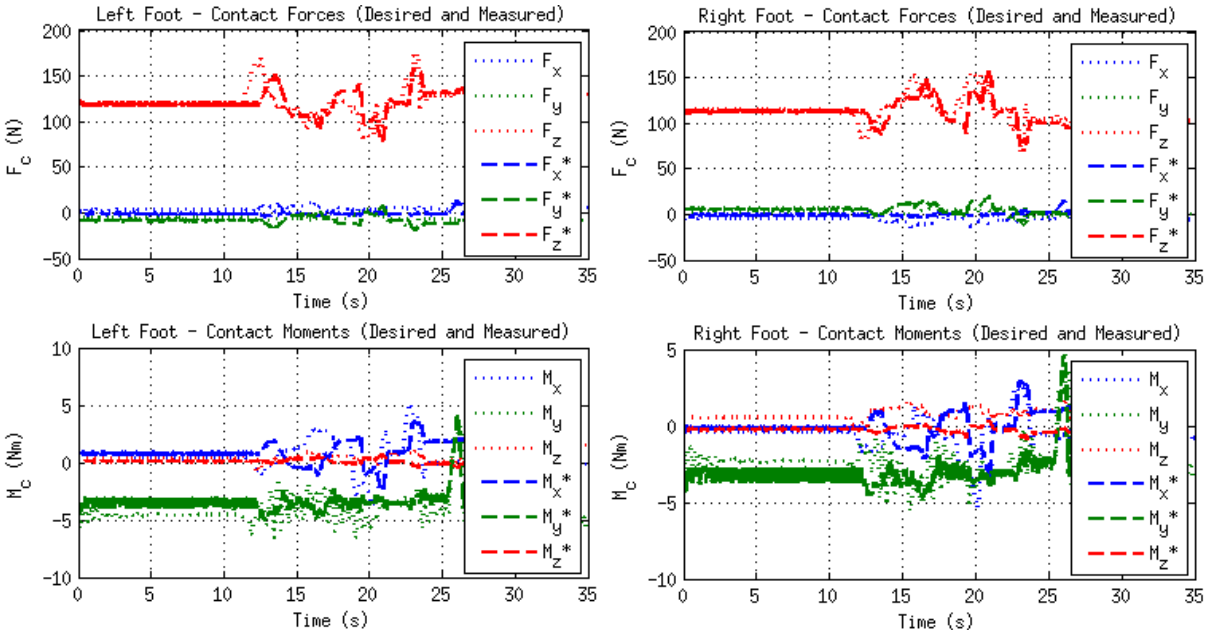

Figure 5.13: **Experiment with QP** - Desired (dashed) and Measured (dotted) Feet Contact Forces - left foot forces(upper left) & left foot moments(lower left) & right foot forces(upper right) & right foot forces(lower right)

In this test with the new controller with torque minimization, the task is the same as the previous test (constant trajectory of CoM in the center). The aim is to observe the motion and contact forces in the cases of without and with external disturbances. The feet contact forces in Figure 5.13 and the CoM trajectory and its error in Figure 5.14 show that during the no disturbance phase in the initial 10 seconds, the robot does not move and it keeps its CoM at the desired position with a minor error. Also when the robot is disturbed, as the robot posture changes, desired contact forces changes and the deflection of the CoM can also be seen in the error plot. However, errors are not kept constant and as the contacts are released, error CoM position gets smaller again. As in the previous test, we see the advantage of using a force and posture controller as it allows the robot have safe interactions with its environment even if it is not aware of these interactions as external disturbances. Also the minimization of torques can be seen in Figure 5.10, where the norm of desired joint torques from this test is depicted in red which is below the norm of the torques from the previous test most of the time even under external disturbances.

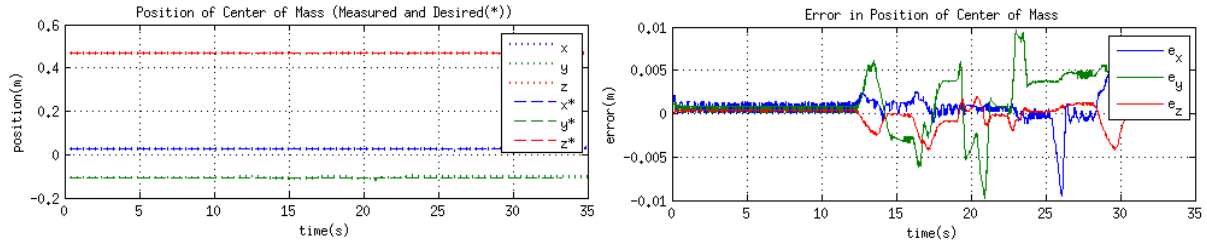

Figure 5.14: **Experiment with QP** - Desired (dashed) and Measured (dotted) Center of Mass trajectories(upper left) & Error in CoM trajectory(lower left) & Snapshots of the robot from the test with contact forces as blue arrows(right)

Experiment with the previous controller (no optimization) for periodic trajectory

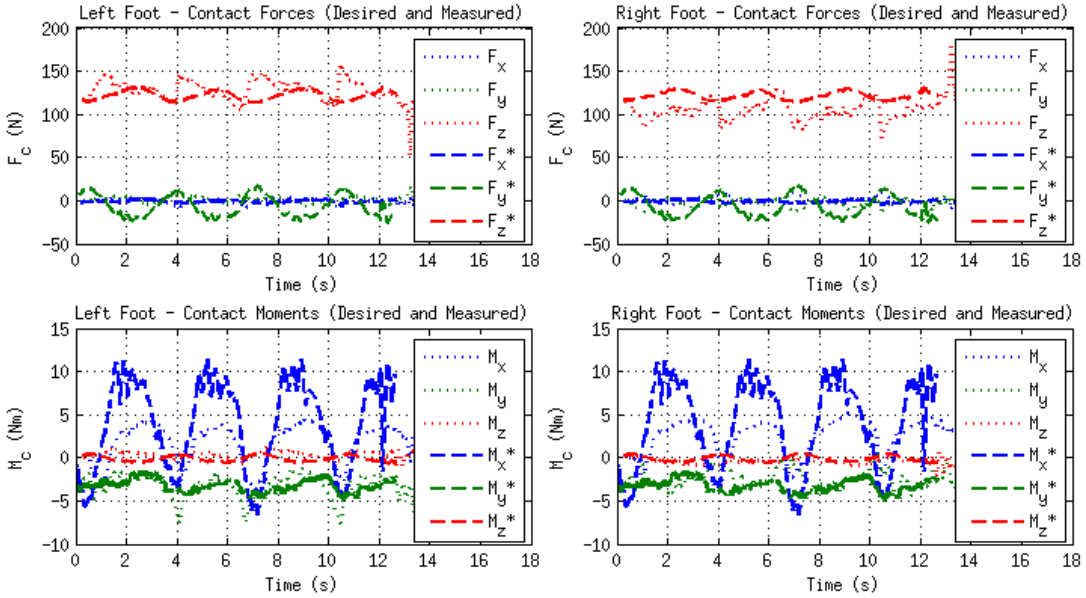

Figure 5.15: **Experiment with no QP** - Desired (dashed) and Measured (dotted) Feet Contact Forces - left foot forces(upper left) & left foot moments(lower left) & right foot forces(upper right) & right foot forces(lower right)

In this test, a task of periodic motion of CoM with an amlitude of 2.5 cm and 0.3 Hz is given to the robot using the old controller. Resulting desired feet contact forces in Figure 5.15 are in a harmony with the task and with each other too, meaning that regardless of the direction causing a symmetry in the posture of the robot, moving CoM sideways results on desired feet contact forces that are similar to eachother. However measured feet contact forces show trends that are opposite to each other in the sense that when vertical force on one foot increases as the CoM moves to that side, the vertical force on the other foot decreases. Computation of desired joint torques from such desired contact forces that require more effort and that may or may not be achievable is suspected to result in unnecessary internal torques. We also suspected that the reason the robot is not able to move its CoM 3 cm and more to the sides with the previous control is the generation of internal torques to achieve such contact forces.

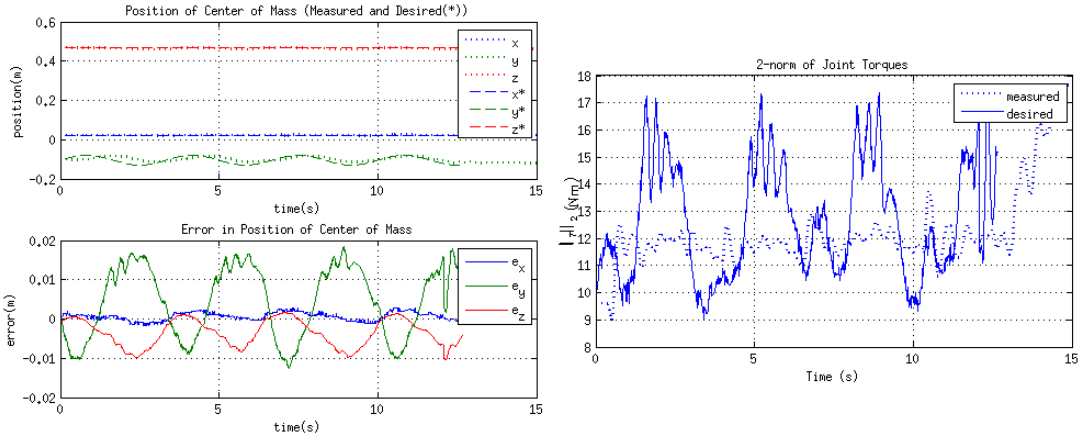

Figure 5.16: **Experiment with no QP** - Desired (dashed) and Measured (dotted) Center of Mass trajectories(upper left) & Error in CoM trajectory(lower left) & Snapshots of the robot from the test with contact forces as blue arrows(right)

Experiment with the new controller (with optimization)

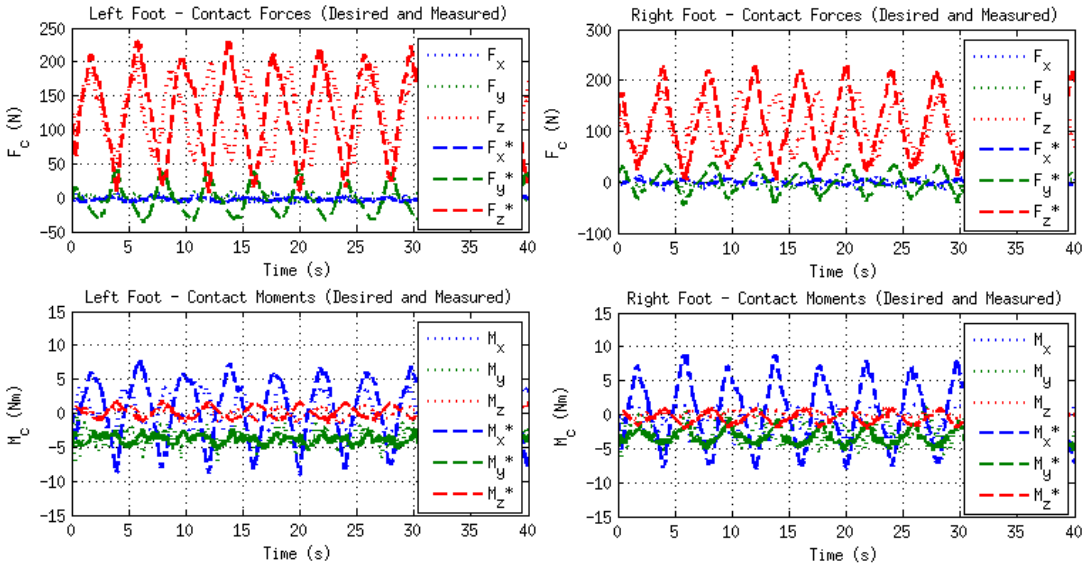

Figure 5.17: **Experiment with QP** - Desired (dashed) and Measured (dotted) Feet Contact Forces - left foot forces(upper left) & left foot moments(lower left) & right foot forces(upper right) & right foot forces(lower right)

In this test[3], a task for the motion of CoM with an amplitude of 4 cm and a frequency of 0.25 Hz is given to the robot with the new controller implemented. With the new controller, the robot is seen to be able to achieve motions that are highly dynamic relative to the previous tasks. As seen in the Figure 5.17, the desired contact forces are now related to the CoM position and the asymmetry of the robot during the motion. In fact the desired vertical forces seem to oscillate as the CoM motion progresses but with a different phase from each other. Also from Figure 5.18's right part, the norm of joint torques seem to oscillate between similar maximum and minimum values with the previous test (Figure 5.16) eventhough the amplitude of motion is increased from 2.5 to 4 cm.

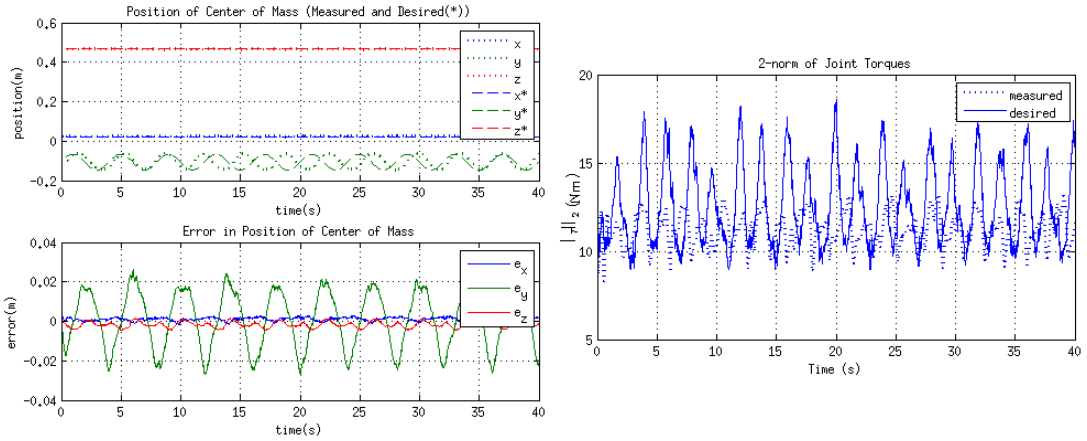

Figure 5.18: **Experiment with QP** - Desired (dashed) and Measured (dotted) Center of Mass trajectories(upper left) & Error in CoM trajectory(lower left) & Snapshots of the robot from the test with contact forces as blue arrows(right)

[3]The measured data (dotted) should be ignored for this experiment. It is due to non-real time operation of two computers collecting data and it will be explained in the next section.

5.5 Discussion

Experimental results showed that with the new formulation of the desired contact forces that relate to the minimization of desired joint torques allows the robot to complete CoM tasks which it was not able to. As we examined for both controllers the desired feet contact forces and resultant joint torque norms, we saw that with the new formulation, the desired contact forces are computed differently which results in a better joint torque distribution among the robot, helping in minimizing the torques as well as moving in a more relaxed and flexible way.

Although the new controller helped in exposing the mechanical capabilities of the robot, there are some issues about the implementation on the experimental setup. Firstly we have noted that neither in the old nor the new controller, there is no feedback from measured contact forces inside the controller. In a way, it relies on the fact that the modeling of the robot and the estimation of the internal wrenches and joint torques are accurate and thus it is possible to accurately compute joint torques that will result in desired contact forces. During the experiments and analysis, we mostly refer to the desired values given by the controller, because it is observed in several cases that the desired and the estimated joint torques differ depending on the current states of the robot. This is an ongoing work carried out by the group and for now it proves to be sufficient for balancing task. Since there is no feedback law for the contact forces, differences between measured and desired contact forces only helps in our analysis to show that with such a framework, despite the uncertainties in the estimation of forces and torques, balancing on both feet and motion tasks for CoM can be achieved even under external disturbances.

The frequency of the high and low level control loops is an important issue that can have a significant effect on the whole body force and postural control. The frequency that is used during testing is 100 Hz (10 ms per loop). With the implementation of the optimization under constraints, unless it is given a challenging task that require a faster motion and activate the constraints, the robot could be controlled at the same frequency of 100 Hz with the optimization too. However this is still slow for controlling a humanoid robot. During tests, we have seen that when some of the constraints on feet contact forces became active (e.g. when the robot is about to slip its foot), the quadratic solver required more time for the solution and if the motion task is challenging, the controller was not able to keep up with the setup in real-time. Additionally, the joint torque control is done at a high-level where it also has a frequency of 100 Hz. There is an ongoing work too, to implement this joint torque control along with estimation of joint torques back into the firmware on the robot which can result in a low-level control with a frequency higher than that of the high-level control. Also currently the controllers are tested on the robot by implementing them into Matlab Simulink models. This makes the process easier but also it creates additional delays in communication and computational effort. That is why during the tests that are presented in this report, two computers were used to collect data to ease the load on one. However due to operating in non-real time and different computational loads, there were differences in the sampling frequencies for both computers. A promising alternative such as using C/C++ for programming controllers for the robot in an efficient way is an ongoing work and moving the control part into a more efficient framework can improve in completing more challenging tasks.

Due to the reasons above, we have gone as far as being able to minimize joint torques based on the new formulation with quadratic programming, but challenging tasks which would result in activating some constraints could not be thoroughly tested, because in such cases, the control loop takes longer than 10 ms, the robot generates jerky motions and it becomes risky as the robot or the things it can interact can get damaged. For less challenging and slower tasks, this was no issue. Despite the fact that the challenging tasks were pushing the computational

effort to the limits, in those cases, the quadratic solver indicates (and therefore the controller knows) when the constraints are becoming active and the motion is becoming more challenging (maybe impossible) and decisions on the control side can be made to take an action to recover from that situation such as a step. Hence, instead of relying on the solution of the optimization problem that finds the feet contact forces while respecting inequality conditions and minimizing the torques, it can be used as an indicator to change the control strategy on the go, which can be useful for the walking task.

Conclusion

With the implementation of torque minimization via a new formulation as a quadratic optimization problem for finding the desired contact wrenches and joint torques, it is seen that the movements of the robot has become more relaxed and easy, such that it can achieve tasks on moving its center of mass with extents and speeds it was not able to achieve before. Along with torque minimiation, we implemented in the optimization problem the conditions on feet contact forces. It was observed that for tasks that we can call less challenging, the controller was able to follow given tasks that required the quadratic program solver to find proper set of solutions when the constraints became active. However, for faster motions, which were more challenging in terms of computational load too, it was not observed. Nevertheless, it is fair to state that with certain conditions on the operation of the optimization problem, the state of the control can be monitored and the cases of instabilities due to contact states changing can be anticipated beforehand and necessary control action can be taken.

Bibliography

[1] G. Metta, L. Natale, F. Nori, G. Sandini, D. Vernon, L. Fadiga, C. von Hofsten, K. Rosander, M. Lopes, J. Santos-Victor, A. Bernardino, and L. Montesano, "The icub humanoid robot: an open-systems platform for research in cognitive development," *Neural Networks*, 2010.

[2] M. Fumagalli, S. Ivaldi, M. Randazzo, L. Natale, G. Metta, and G. Sandini, "Force feedback exploiting tactile and proximal force/torque sensing. theory and implementation on the humanoid robot icub," *Autonomous Robots*, 2010.

[3] iCub Specifications, "http://www.icub.org/images/brochures/icubflyer.pdf,"

[4] the iCub Wiki, "http://wiki.icub.org/wiki/icub_software_installation,"

[5] A. D. Prete, L. Natale, F. Nori, and G. Metta, "Contact force estimations using tactile sensors and force / torque sensors," *Human Robot Interaction*, 2012.

[6] S. Ivaldi, M. Fumagalli, M. Randazzo, F. Nori, G. Metta, and G. Sandini, "Computing robot internal/external wrenches by means of inertial, tactile and f/t sensors: theory and implementation on the icub," *Proc. of the 11th IEEE-RAS International Conference on Humanoid Robots, Bled, Slovenia*, 2011.

[7] A. Del Prete, *Control of Contact Forces using Whole-Body Force and Tactile Sensors: Theory and Implementation on the iCub Humanoid Robot.* PhD thesis, Universita di Genova, Istituto Italiano di Tecnologia, Genoa, February 2013.

[8] mex-wholebodymodel project GitHub Repository, "https://github.com/robotology-playground/mex-wholebodymodel," *Robotology Playground.*

[9] D. E. Orin, A. Goswami, and S.-H. Lee, "Centroidal dynamics of a humanoid robot," *Autonomous Robots*, 2013.

[10] M. W. Spong, "The control of underactuated mechanical systems," *Plenary Address at the First International Conference on Mechatronics*, 1994.

[11] A. Del Prete, N. Mansard, F. Nori, G. Metta, and L. Natale, "Partial force control of constrained floating-base robots," *Intelligent Robots and Systems (IROS)*, 2014.

[12] J. Nocedal and S. J. Wright, *Numerical Optimization.* Springer, 1999.