Eindhoven University of Technology
Department of mechanical Engineering
Dynamics and control

Hybrid trajectory tracking for impact motions of humanoid robots

DC 2016.035

Eindhoven, June 28, 2016

E.B.C. de Mooij
0734166

Supervisor: prof.dr.ir. N. van de Wouw
Coaches: dr. A. Saccon
ir. S. Traversaro
ir. M.W.L.M. Rijnen

Abstract

Humanoid robots have long been one of humanities great ambitions. They have a couple key advantages over other robots, such as being able to use tools and infrastructure designed for humans. Although steady progress on humanoid robots is made, they are still of limited practical use. One of the main reasons for this limited use is the difficulty of controlling humanoid robots.

This report provides a proof of concept of a reference spreading based hybrid controller (RS-based controller) on a humanoid robot simulation. The RS-based controller is a recently developed control strategy for hybrid dynamical systems with hard impacts, based on an extended hybrid reference trajectory. In our proof of concept, we apply the RS-based hybrid controller on the simulation of a humanoid robot called the iCub.

In this report, we make a couple key steps in providing a proof of concept. Firstly, we formulate a dynamical model of the iCub that is suitable for simulating contact tasks. Secondly, we find a way to implement the RS-based controller on the iCub, using a task-level controller to generate the extended hybrid reference trajectories. Finally, we perform simulations of various, relevant case studies to serve as proof of concept.

In our case study simulations, we find that the RS-based controller can successfully stabilize a contact task for the iCub. The controller has favourable robustness properties in the nominal case, and has a reasonably good performance. However, the controller displayed poor performance and a lack of robustness for simulations with modular imperfections. While the RS-based control strategy shows promising results, it has to be combined with other control theories to be of practical use in the field of robotics.

List of symbols

I_n	$n \times n$ identity matrix
$0_{n \times m}$	$n \times m$ zero matrix
$1, 2, 3, ...$	Coordinate frames
$i[j]$	Mixed frame with the origin of i and the orientation of j
o_i	Origin of frame i
${}^{j}R_i$	Rotation matrix from i to j
${}^{j}H_i$	Homogeneous transformation matrix from i to j
${}^{j}\omega_{i,k}$	Rotational velocity of k w.r.t. i, expressed in j
${}^{j}v_{i,k}$	Linear velocity of k w.r.t. i, expressed in j
${}^{j}\mathrm{v}_{\mathrm{i,k}}$	Twist of k w.r.t. i, expressed in j
${}^{j}X_i$	Velocity transformation matrix from i to j
${}_{j}f$	Force vector expressed in j
${}_{j}\tau$	Torque vector expressed in j
${}_{j}\mathrm{f}$	Wrench, a combination of forces and torques expressed in j
${}_{j}X^{i}$	Force transformation matrix from i to j
${}^{j}J_{i,k/l}$	Geometric Jacobian of k relative to i, expressed in j with base velocity expressed in l
q	Generalized coordinate set
q_J	Joint displacement vector
H	Homogeneous transformation expressing the pose of the base link
ν	Set of generalized velocities
v	Base link velocity
χ	State of the iCub
M	Mass matrix
C	Coriolis matrix
G	Potential force vector
τ	Joint torques
S	Matrix with generalized force directions related to the joing torques
h	Vector of generalized bias forces
f_{foot}	Contact wrench acting on the left foot frame 6
f_{foot}	Contact force vector acting on the left foot frame 6
τ_{foot}	Contact torque vector acting on the left foot frame 6
p	Zero tipping moment point
C_{foot}	Constraint matrix
x_{wall}	x position of the wall
f_{hand}	Contact force vector acting on the left hand frame 4
$\mathrm{g}^{j \leftarrow i}$	Guard function from mode i to mode j
c_i	Constraint function of the i-th constraint
$\Gamma^{j \leftarrow i}$	Velocity reset map from mode i to mode j

J_{foot}	Jacobian corresponding with the left foot frame 6
J_{hand}	Jacobian corresponding with the linear velocity part of the left handframe 6
k_p, k_d	Constants used for Baumgarte stabilization
Λ_{foot}	6 dimensional force/torque impulse acting on the left foot
λ_{hand}	3 dimensional force impulse acting on the left hand
t_i	Impact time
t_i^*	Expected impact time
α	Reference trajectory
$\bar{alpha}$	Extended reference trajectory
μ	Feed forward
e_s	Traditional tracking error
e	Hybrid tracking error
β	Task
P	Centroidal momentum
k_s	Safety margin constant
$k_{p,CoM}, k_{d,CoM}$	Feedback constants for the CoM tracking task
$k_{p,hand}, k_{d,hand}$	Feedback constants for the hand tracking task
$k_{p,pos}, k_{d,pos}$	Feedback constants for the postural task
K_p, K_d	Hybrid controller feedback matrices

Contents

1 Introduction **1**
- 1.1 Advantages of humanoid robots . . . 1
- 1.2 Current humanoid robots . . . 2
  - 1.2.1 iCub . . . 3
- 1.3 Brief review of literature . . . 4
- 1.4 Research objective . . . 4
- 1.5 Report structure . . . 5

2 Multibody dynamics with unilateral constraints **6**
- 2.1 Multibody system notation . . . 6
  - 2.1.1 Time derivatives and differentials . . . 6
  - 2.1.2 Positions, velocities and forces . . . 7
  - 2.1.3 Jacobian . . . 10
- 2.2 iCub humanoid robot . . . 11
- 2.3 Contact dynamics . . . 12
  - 2.3.1 Left foot contact . . . 13
  - 2.3.2 Left hand contact . . . 15
  - 2.3.3 Hybrid system . . . 15
  - 2.3.4 Constrained dynamical model . . . 17
  - 2.3.5 Velocity reset map . . . 19

3 Hybrid trajectory tracking applied to a humanoid robot model **21**
- 3.1 Reference spreading control for hybrid systems . . . 21
- 3.2 Optimal constrained controller . . . 25
  - 3.2.1 Optimal constrained controller: free motion mode . . . 26
  - 3.2.2 Optimal constrained controller: contact mode . . . 28
  - 3.2.3 Minimizing joint motion . . . 29
- 3.3 Generating the state and input reference trajectory . . . 31
  - 3.3.1 Reaching for the wall . . . 31
  - 3.3.2 Pushing away from the wall . . . 32
  - 3.3.3 Moving to initial position . . . 35
  - 3.3.4 The complete desired motion with extensions . . . 37
- 3.4 The iCub hybrid control law . . . 38

4 Simulation study: performance of the RS hybrid controller on the iCub **40**
- 4.1 Perturbations of initial conditions . . . 41
  - 4.1.1 Random perturbation of the initial conditions . . . 41
  - 4.1.2 Selectively perturbed initial conditions: large impact time mismatch . . . 46
- 4.2 Model imperfections . . . 52
  - 4.2.1 Encoder bias . . . 53
  - 4.2.2 Imperfect actuators . . . 55

4.2.3 Changing the wall position 58
4.3 Discussion 64

5 Conclusions and recommendations **65**
5.1 Conclusions 65
5.2 Recommendations 66

Chapter 1

Introduction

For centuries, humanity has had ambitious goals and fantasies for future technology. While in the past people envisioned personal flying machines and miraculous cures, today humanoid robots are one of the technologies expected to become widely adopted in the future. The potential applications are endless, and range from robotic nurses and laborers to companionship robots. While significant progress is consistently being made in the field of humanoid robotics, the current practical applications are still limited. The main reasons are the costs of the physical robot and the difficulty of programming and controlling them in an effective and robust manner.

1.1 Advantages of humanoid robots

When mentioning humanoid robots in this manuscript, we consider bipedal robots that strongly resemble the human anatomy. Although these kind of robots are difficult to design, produce and control, they have a couple key advantages over other types of robots. First and foremost, most of our infrastructure is optimized for humans. From stairs and sidewalks to tools and vehicles, most objects encountered in both daily life and the industry are specifically designed for human use. A humanoid robot can make effective use of all these objects. It can move around buildings using the stairs, perform repairs with conventional tools, drive in cars and open doors. It could possibly be able to perform any action that an actual human can, and could therefore potentially replace humans in a wide variety of simple tasks and labor.

For industrial applications, humanoid robots do not share many of the disadvantages of human employees. Robots can be sent to work in highly hazardous environments, as they can be designed to be immune to, e.g., extreme temperatures or large amounts of toxic chemicals and radiation. Furthermore, they can work for extended periods of time without rest, allowing labor to continue 24 hours a day. Unlike other kinds of robots, humanoids do not require significant changes to the infrastructure to function, giving them a unique set of advantages over both human employees and traditional industrial robots.

Another key advantage is that humanoid robots can be made to look like humans. This can significantly increase the acceptance of humans towards robots. Studies have shown that, for a wide variety of tasks, people prefer robots that are similar in appearance to humans [1]. This is mainly the case for tasks in which compassion and trust are important, such as those performed by nurses, teachers and shop assistants. When these tasks are performed by human-looking robots, the customers have shown to be significantly more cooperative with robots [1].

1.2 Current humanoid robots

To give an insight of the current state of development for humanoid robots, we provide a few examples of current humanoid robots. These examples are just a few of the many innovative robots that currently exist, and should not be seen as a complete representation of the current state of the field of robotics. Thereafter, we give a brief introduction of the iCub robot, a humanoid robot with open-source software and hardware that will be used throughout this manuscript.

The ASIMO is an advanced humanoid robot, first presented by Honda in 2000[1]. ASIMO stands for **A**dvanced **S**tep in **I**nnovative **MO**bility and, according to Honda, is the world's most advanced humanoid robot [1]. The robot is displayed in Figure 1.1a. The most recent version can walk, run (up to 9km/hour), jump and climb stairs. Furthermore, it can perform actions such as shaking hands, pouring a drink in a cup, using sign language, and recognize a persons face [1].

(a) The ASIMO humanoid robot. Image courtesy Honda [1].

(b) The TORO humanoid robot. Image courtesy DLR[2].

(c) The HRP-2 humanoid robot. Image courtesy kawada[3].

Other advanced humanoid robots are the TORO and HRP-2, displayed in Figure 1.1b and Figure 1.1c, respectively. These robots are focused on balance and locomotion in particular. For example, the HRP-2 can use obstacles like tables to support itself during motion tasks, and the TORO can stand on uneven surfaces. These advanced humanoid robots are too expensive for most practical applications. However, they serves as a proof of principle for the possible applications of humanoid robots.

Humanoid robots have already been built successfully, although they are still very expensive. The current main challenges lie in the field of, among others, automated intelligence, perception

[1] Asimo website at http://asimo.honda.com/
[2] TORO website at http://www.dlr.de/rmc/rm/en/desktopdefault.aspx/tabid-6838/
[3] HRP-2 website at http://global.kawada.jp/mechatronics/hrp2.html

of the environment, interaction with humans, and motion control [5]. A typical challenge for humanoid robots is to keep their balance, both during walking and actual tasks. Humanoid robots are complex dynamical systems with many internal degrees of freedom. Since they have no direct link with their environment, humanoid robots are underactuated systems, relying on contact with the environment to achieve controllability [6]. This is one of the reasons that the perception of the environment is such a vital challenge for humanoid robots. If the environment is misinterpreted, an unexpected contact force, or the lack of an expected one, can disturb the robots balance. This manuscript focuses on the control of robots during contact tasks with impacts at relatively high speed.

1.2.1 iCub

The iCub is an advanced humanoid robot with open-source hardware and software. It has the body of a three and a half year old child [2]. It has 53 internal degrees of freedom, and has been designed to allow manipulation and locomotion studies [2]. Because the iCub and its software are open source, this robot is very suitable for research purposes, with more than 30 robots available in Europe. Among the functions developed for the iCub are a balancing controller [3] and an articulated talking face [4].

Figure 1.2: The iCub humanoid robot. Image courtesy IIT Genova [20].

The research in this report is applied on the iCub humanoid robot. The availability of open-source software and models makes it an attractive research object. Since we do not have access to a physical iCub robot, we make use of simulations of a dynamical model of the iCub. In this report, we aim to create and research a hybrid controller for contact tasks of the iCub. We base this controller on the reference spreading based hybrid controller (RS hybrid controller), described in [7] and [8] and the iCub optimal constrained balancing controller (OC-controller) described in [3].

There are a couple of key challenges that are taken on in this report. First, we need to formulate a dynamical model of the iCub that is suitable for simulating contact tasks. As stated before, humanoid robots such as the iCub have complex dynamical models with many degrees of freedom. Since we want to simulate a contact task, it is required to incorporate suitable contact dynamics in our model. Secondly, we have to find a way to design and implement the RS hybrid controller for/on the iCub. Finally, if we are to study the performance of such novel iCub controller, we need to perform simulations in various, relevant conditions, and define a performance criteria for the controller.

1.3 Brief review of literature

The control of humanoid robots is an important and thoroughly investigated field of research, but still has open challenges [9]. Several different control strategies have been developed. A particularly popular approach is a task based control strategy, where a predefined task sequence is used to generate a reference online [10]. Once a subtask is completed, i.e., an impact occurs, the reference for the next subtask is generated. Many walking controllers make use of this strategy, dividing the walking motion in several subtasks, like moving the foot towards the ground until it makes contact. An example of such a walking controller can be found in [11], where the authors presents a controller that uses 5, subsequently performed, subtasks for walking. This kind of controllers is suitable when the order of tasks is important, but fail to effectively track a time-dependent reference function with chaning transitions between subtasks [10].

A strictly time based control theory is often capable of effectively tracking a time-dependent reference function. However, such controllers are mainly implemented successfully in noncontact tasks, and are often unable to cope with impacts. A controller that combines a time-based and task-based control theory for humanoid robots does not yet exist. In [7], the authors propose a reference spreading based (RS) hybrid control theory that can follow a predefined time varying trajectory. They make use of a hybrid reference that has a noncontact and a contact reference around the expected times of impact. The controller then uses the reference that corresponds with the current mode of the system. The hybrid reference is obtained by extending parts of a conventional reference forward and backwards in time, around the expected impact times. However, this controller has only been researched on simple dynamical systems [7].

1.4 Research objective

The research objective of this report is to **provide a proof of concept for the RS hybrid control strategy on a simulation of a humanoid robot performing a relatively high speed contact task**. As a side objective, we aim to perform a performance analysis on the proof of concept controller.

We can formulate a couple key steps in obtaining our proof of principle.

We begin our research by **formulating a dynamical model of the iCub, capable of performing a contact task**. We model the iCub as a hybrid dynamical system where we focus on relatively simple contact dynamics, in particular, an arm initiating a point contact with a wall. In this system, impacts are considered discrete events in which a velocity reset map is used to calculate the post event state.

Once we have a suitable dynamical model for the iCub, **we generate a reference trajectory for the RS hybrid controller**. We use an adjusted version of the OC-controller [3] to obtain

a motion signal of the iCub performing a contact task without losing balance. This motion signal serves as the basic reference for the RS hybrid controller. By extending the motion signal around the event times, we obtain a hybrid trajectory suitable for the RS hybrid controller. **We finalize the hybrid controller by designing and tuning the control law.**

The proof of concept of the RS hybrid controller can now be created by **performing simulations of the controller on the iCub model**. We use several disturbances on the model, based on physical phenomena such as encoder bias and actuator imperfections. We perform a concise performance analysis on the different simulations.

1.5 Report structure

In Chapter 2, we provide a multibody notation for the iCub model, and derive a hybrid dynamical model. We pay special attention to the contact dynamics in the model. Chapter 3 presents a RS hybrid controller for the iCub. We first explain the underlying theory of RS hybrid controller, after which we present an optimal constrained controller that can generate a contact motion of the iCub. We finish this chapter by presenting the hybrid reference and feedback law used in the RS hybrid controller. Chapter 4 contains several simulations of the RS hybrid controller on the iCub model. We conclude this manuscript with conclusions and recommendations.

Chapter 2

Multibody dynamics with unilateral constraints

In this chapter, the dynamical model for the iCub robot is presented. The iCub model is a complex robotic system with many rigid bodies and degrees of freedom. To clearly describe this dynamical model, we make use of a recently proposed [13] rigid body notation to express homogeneous transformation matrices, linear and rotational velocities, and forces and torques. We start this chapter by presenting this notation. Thereafter, we give a brief description of the iCub and introduce its free floating dynamical model. The model is completed by describing the constraint forces that have to be included when the robot makes contact with the environment. The resulting hybrid model will be used throughout this manuscript. The last section of this chapter describes the addition of these constraints to the model.

2.1 Multibody system notation

In the fields of robotics and dynamics, several different notations are used. Examples of notations can be found in Chapter 2.2 of "Springer Handbook of Robotics" [12], which gives a detailed explanation of the Featherstone notation typically used in the field of robotics, and "Multibody Dynamics Notation" [13]. The latter is a recent attempt to unify the Featherstone notation with Lie groups, often used in differential geometry for mechanics, and is used as basis for our notation. The following notation is used in this report.

2.1.1 Time derivatives and differentials

Consider a time varying matrix $A(t)$ of dimension $n \times m$, with t denoting time. We write the infinitesimal perturbation of $A(t)$ as

$$\delta A(t). \tag{2.1}$$

The time derivative of $A(t)$, denoted as $\dot{A}(t)$, is defined as

$$\dot{A}(t) := \frac{\mathrm{d}A(t)}{\mathrm{d}t}. \tag{2.2}$$

When considering a scalar function, e.g., the function $a(b, c) \in \mathbb{R}$ with arguments $b, c \in \mathbb{R}$. The partial derivative of $a(b, c)$ with respect to b is denoted with

$$\frac{\partial a}{\partial b}. \tag{2.3}$$

In a matrix function, a different notation is used. Consider, for example, a 2×2 matrix-valued function of two arguments $A(b, c)$, with $b, c \in \mathbb{R}$. The partial derivative of A with respect to b is denoted as

$$\mathrm{D}_1 A(b,c) = \begin{bmatrix} \frac{\partial A_{11}}{\partial b} & \frac{\partial A_{12}}{\partial b} \\ \frac{\partial A_{21}}{\partial b} & \frac{\partial A_{22}}{\partial b} \end{bmatrix}. \tag{2.4}$$

Similarly, the partial derivative of A with respect to c is denoted as

$$\mathrm{D}_2 A(b,c) = \begin{bmatrix} \frac{\partial A_{11}}{\partial c} & \frac{\partial A_{12}}{\partial c} \\ \frac{\partial A_{21}}{\partial c} & \frac{\partial A_{22}}{\partial c} \end{bmatrix}. \tag{2.5}$$

2.1.2 Positions, velocities and forces

Coordinate vector

Consider a point p and a frame with origin o_1 and orientation (set of three unit vectors) denoted as [1]. This frame, collectively written as $\{1\} = (o_1, [1])$, will be denoted simply as 1. The coordinates of p with respect to 1, i.e., the vector from o_1 to p, expressed in 1, are denoted

$${}^1p = ({}^1p_x, {}^1p_y, {}^1p_z)^T. \tag{2.6}$$

Similarly, the origin of frame 2 is denoted as o_2. The coordinates of o_2 with respect to frame 1 are written as

$${}^1o_2. \tag{2.7}$$

Rotation matrix

Given two frames 1 and 2, the orientation of 2 w.r.t. 1 is given by the rotation matrix

$${}^1R_2. \tag{2.8}$$

If the origins of two frames coincide, the rotation matrix can be used as a coordinate transformation, e.g.

$${}^1p = {}^1R_2 {}^2p \quad \text{when} \quad {}^1o_1 = {}^1o_2. \tag{2.9}$$

Homogeneous transformation matrix

Given two frames 1 and 2, the position and orientation of 2 with respect to 1 can be written as a homogeneous transformation matrix

$${}^1H_2 := \begin{bmatrix} {}^1R_2 & {}^1o_2 \\ 0_{1\times 3} & 1 \end{bmatrix}. \tag{2.10}$$

Considering a point p, the homogeneous representation of p, expressed in 2, is defined as

$${}^2\bar{p} := [{}^2p; 1]. \tag{2.11}$$

The homogeneous representation of p expressed in 1 can be calculated with

$${}^1\bar{p} = {}^1H_2 {}^2\bar{p}. \tag{2.12}$$

Hat and vee operations

Given the vector $w = (x, y, z)^T \in \mathbb{R}^3$, the 3×3 skew-symmetric matrix $w^\wedge$ (read w *hat*) is defined as

$$w^\wedge := \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}. \tag{2.13}$$

Similarly, given a vector $\mathrm{v} = (v_x, v_y, v_z, \omega_x, \omega_y, \omega_z)^T \in \mathbb{R}^6$, the matrix $\mathrm{v}^\wedge$ is defined as

$$\mathrm{v}^\wedge := \begin{bmatrix} 0 & -\omega_z & \omega_y & v_x \\ \omega_z & 0 & -\omega_x & v_y \\ -\omega_y & \omega_x & 0 & v_z \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{2.14}$$

The *vee* operation is the inverse of the *hat* operation. Considering a 3×3 skew symmetric matrix W, the vector $W^\vee$ is given as

$$W^\vee = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}^\vee := \begin{bmatrix} x \\ y \\ z \end{bmatrix}. \tag{2.15}$$

Similarly, considering the 4×4 matrix N, the vector $N^\vee$ is given as

$$N^\vee = \begin{bmatrix} 0 & -\omega_z & \omega_y & v_x \\ \omega_z & 0 & -\omega_x & v_y \\ -\omega_y & \omega_x & 0 & v_z \\ 0 & 0 & 0 & 0 \end{bmatrix}^\vee := \begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}. \tag{2.16}$$

Body twist

Given two frames 1 and 2, the relative velocity of 2 w.r.t. 1, expressed in 2, can be expressed as a combination of the linear and angular velocity, called twist

$${}^2\mathrm{v}_{1,2} := \begin{bmatrix} {}^2v_{1,2} \\ {}^2\omega_{1,2} \end{bmatrix} \in \mathbb{R}^6, \tag{2.17}$$

with ${}^2v_{1,2} \in \mathbb{R}^3$ the linear velocity vector and ${}^2\omega_{1,2} \in \mathbb{R}^3$ the angular velocity vector of 2 w.r.t. 1, expressed in 2. The twist ${}^2\mathrm{v}_{1,2}$ can be written as a homogeneous twist matrix

$${}^2\mathrm{v}_{1,2}^\wedge = \begin{bmatrix} {}^2\omega_{1,2}^\wedge & {}^2v_{1,2} \\ 0_{1\times 3} & 0 \end{bmatrix}. \tag{2.18}$$

As explained in Chapter 4 of [13], this matrix can be calculated as

$${}^2\mathrm{v}_{1,2}^\wedge = {}^1H_2^{-1}\, {}^1\dot{H}_2, \tag{2.19}$$

with ${}^1H_2^{-1}$ denoting the inverse of ${}^1H_2$ and ${}^1\dot{H}_2$ the time derivative of ${}^1H_2$. The twist as calculated above will be called body twist, and is often referred to in other literature as left trivialized velocity [13].

Spatial twist

The velocity of 2 w.r.t. 1 can also be expressed in 1. For this, the term "spatial twist" is introduced. The spatial twist of 2 w.r.t. 1 is denoted ${}^{1}\mathrm{v}_{1,2}$. It describes the twist of the point fixed to 2 that happens to coincide with o_1 at the current instant, expressed with respect to the position and orientation of 1 [12]. The spatial twist ${}^{1}\mathrm{v}_{1,2}$ can be written as a twist matrix

$$ {}^{1}\mathrm{v}_{1,2}^{\wedge} = \begin{bmatrix} {}^{1}\omega_{1,2}^{\wedge} & {}^{1}v_{1,2} \\ 0_{1\times 3} & 0 \end{bmatrix}. \tag{2.20} $$

As explained in [13], this matrix can be calculated as

$$ {}^{1}\mathrm{v}_{1,2}^{\wedge} = {}^{1}\dot{H}_2 \, {}^{1}H_2^{-1}. \tag{2.21} $$

Note that [13] use the term "right-trivialized velocity". The body twist can be related to the spatial twist using the velocity transformation matrix ${}^{1}X_2$, i.e.,

$$ {}^{1}\mathrm{v}_{1,2} = {}^{1}X_2 \, {}^{2}\mathrm{v}_{1,2}, \tag{2.22} $$

where the velocity transformation matrix ${}^{1}X_2$ is given by

$$ {}^{1}X_2 = \begin{bmatrix} {}^{1}R_2 & {}^{1}o_2^{\wedge} \, {}^{1}R_2 \\ 0_{3\times 3} & {}^{1}R_2 \end{bmatrix}. \tag{2.23} $$

Mixed twist

The velocity of 2 w.r.t. 1 could alternatively be expressed in a so-called mixed frame with origin o_2 and orientation [1], denoted as $(o_2, [1])$. This velocity representation will be called "mixed twist" and is denoted ${}^{2[1]}\mathrm{v}_{1,2} \in \mathbb{R}^6$. The mixed velocity can be calculated from the body twist with

$$ {}^{2[1]}\mathrm{v}_{1,2} = {}^{2[1]}X_2 \, {}^{2}\mathrm{v}_{1,2}. \tag{2.24} $$

The velocity transformation ${}^{2[1]}X_2$ is given by

$$ {}^{2[1]}X_2 = \begin{bmatrix} {}^{1}R_2 & 0_{3\times 3} \\ 0_{3\times 3} & {}^{1}R_2 \end{bmatrix}, \tag{2.25} $$

as derived in [13].

Wrench

Consider a frame 2, rigidly attached to a rigid body L, and a force ${}_{2}f_L$ and torque ${}_{2}\tau_L$, both expressed in 2, applied on L at point o_2. The combination of this force and torque is called a wrench. The wrench applied to body L, expressed in 2 becomes

$$ {}_{2}\mathrm{f}_L := \begin{bmatrix} {}_{2}f_L \\ {}_{2}\tau_L \end{bmatrix}. \tag{2.26} $$

To express the wrench in frame 1, the wrench transformation matrix ${}_{1}X^{2}$ is introduced such that

$$ {}_{1}\mathrm{f}_L = {}_{1}X^{2} \, {}_{2}\mathrm{f}_L. \tag{2.27} $$

The wrench transformation matrix can calculated from the fact that the power related to a force is independent of the frame in which it is expressed. It follows that

$$ {}_{2}\mathrm{f}_L^{T} \, {}^{2}\mathrm{v}_{1,2} = {}_{1}\mathrm{f}_L^{T} \, {}^{1}\mathrm{v}_{1,2}, \tag{2.28} $$

which can be rewritten to

$$ {}_2\mathrm{f}_L^T \, {}^2\mathrm{v}_{1,2} = ({}_1X^2 \, {}_2\mathrm{f}_L)^T \, {}^1X_2 \, {}^2\mathrm{v}_{1,2}, \tag{2.29} $$

and subsequently to

$$ {}_2\mathrm{f}_L^T \, {}^2\mathrm{v}_{1,2} = {}_2\mathrm{f}_L^T \, {}_1X^{2^T} \, {}^1X_2 \, {}^2\mathrm{v}_{1,2}, \tag{2.30} $$

which results in the relation

$$ {}_1X^2 = {}^1X_2^{-T} = {}^2X_1^T. \tag{2.31} $$

2.1.3 Jacobian

Consider a free-floating, multibody system with a base frame 2 and an end-effector frame 3. Its configuration is parametrized using $q = (H, q_J) \in SE(3) \times \mathbb{R}^{n_J}$, with $H = {}^1H_2 \in SE(3)$ the position and orientation of the base frame w.r.t. an inertial frame 1, and $q_J \in \mathbb{R}^{n_J}$ the joint configuration of n_J joints. The infinitesimal perturbations of ${}^1H_3$ can be written as

$$ {}^1\delta H_3 = \mathrm{D}_1 {}^1H_3 \cdot {}^1\delta H_2 + \mathrm{D}_2 {}^1H_3 \cdot \delta q_J, \tag{2.32} $$

with

$$ {}^1\delta H_2 := \delta {}^1H_2, \tag{2.33} $$

$$ {}^1\delta H_3 := \delta {}^1H_3. \tag{2.34} $$

We define the left trivialized infinitesimal perturbations ${}^2\Delta_{1,2}$ and ${}^3\Delta_{1,3}$ such that

$$ {}^2\Delta_{1,2}^\wedge = {}^1H_2^{-1} \, {}^1\delta H_2 \tag{2.35} $$

and

$$ {}^3\Delta_{1,3}^\wedge = {}^1H_3^{-1} \, {}^1\delta H_3. \tag{2.36} $$

As stated in [13], one can find a linear relation between ${}^3\Delta_{1,3}^\wedge$, ${}^2\Delta_{1,2}^\wedge$ and q_J from (2.32). This relation can be written as

$$ {}^3\Delta_{1,3} = {}^3J_{1,3/2} \begin{bmatrix} {}^2\Delta_{1,2} \\ \delta q_J \end{bmatrix}, \tag{2.37} $$

in which we call ${}^3J_{1,3/2}$ the Jacobian. As explained in [13], the Jacobian can be expressed in other frames, using

$$ {}^5J_{1,3/4} = {}^5X_3 \, {}^3J_{1,3/2} \, {}^2\bar{X}_4, \tag{2.38} $$

with

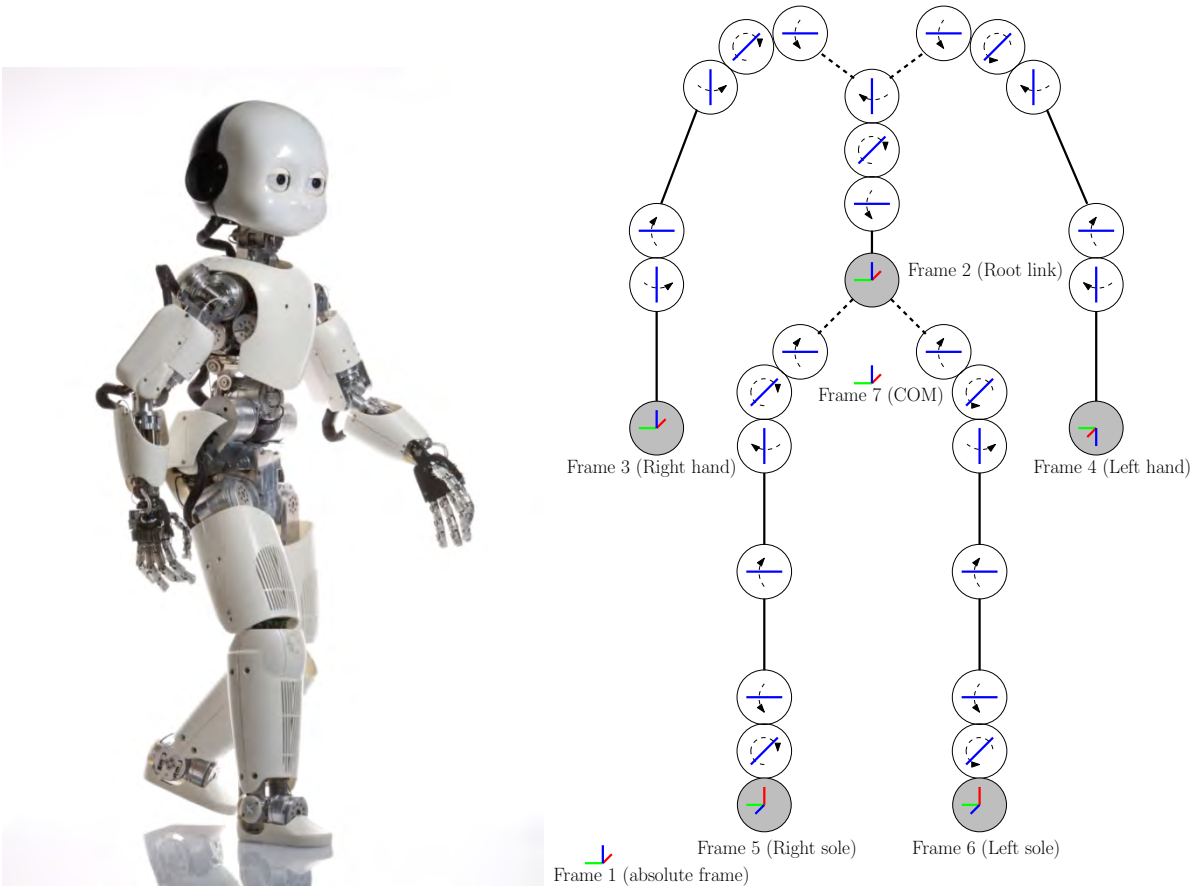
$$ {}^2\bar{X}_4 = \begin{bmatrix} {}^2X_4 & 0_{6 \times n_J} \\ 0_{n_J \times 6} & I_{n_J \times n_J} \end{bmatrix}. \tag{2.39} $$

Now that we presented the notation used in this report, we can start to formulate a dynamical model of the iCub.

2.2 iCub humanoid robot

The iCub is a humanoid robot and a classical example of a simply supported free floating robot, i.e., a robot that is not rigidly attached to the ground, allowing for locomotion. Other examples of free floating robots are unmanned air vehicles, autonomous underwater vehicles, and mobile robots. A picture of the iCub is given in Figure 2.1a.

We can model the iCub as a free floating body with a floating (underactuated) base, alternatively called root link, and several internal (actuated) joints. The joints and root link of the iCub can be seen in Figure 2.1b, which depicts a schematic overview of the iCub. The figure depicts, in particular, the reference frames that are used in this report: the absolute frame 1, the root link (frame 2), right and left hand (frame 3 and 4), right and left sole (frame 5 and 6), and the center of mass (frame 7). Frame 7 is not rigidly attached to any part of the iCub, but instead, its origin o_7 corresponds to the center of mass of the iCub, and its orientation [7] is the same as that of frame 1, i.e., [7] = [1].



(a) The iCub humanoid robot.

(b) A schematic overview of the iCub.

Figure 2.1: (a) The iCub humanoid robot, image courtesy of of IIT Genova, and (b) a schematic overview of the iCub joints and key reference frames used in this work: the absolute frame (frame 1), the root link (frame 2), right and left hand (frame 3 and 4), right and left sole (frame 5 and 6), and the center of mass (frame 7).

We can describe the state q of the iCub with a homogeneous transformation matrix ${}^1H_2$, from now on simply denoted as H, describing the position and orientation of the root link frame 2 w.r.t. the inertial frame 1, and n_J joint variables q_J. We define the generalized coordinates q,

therefore, as

$$q := (H, q_J) \in SE(3) \times \mathbb{R}^{n_J}. \tag{2.40}$$

The generalized velocity of the iCub can be expressed with the root link twist $\mathrm{v} := {}^{2[1]}\mathrm{v}_{1,2}$, describing the body velocity of the iCubs base w.r.t. the inertial frame 1, and the time derivative of q_J, combined into the generalized velocity ν defined as

$$\nu := (\mathrm{v}, \dot{q}_J). \tag{2.41}$$

The configuration and velocity variables define the state χ of the system, namely

$$\chi := [q^T, \nu^T]^T. \tag{2.42}$$

Being a free-floating robot, the iCub is an under-actuated system. While all internal degrees of freedom (the joints) are actuated, the external degrees of freedom (position and orientation of the root link) are not directly actuated. The iCub can use contacts with the environment to constrain some degrees of freedom. When 6 or more degrees of freedom are constrained, it is possible for the system to become fully actuated or even over actuated. As for any free-floating robotic system with contacts described by multi-body dynamics, the equations of motion have the following form [12]:

$$M(q)\dot{\nu} + C(q,\nu)\nu + G(q) = S\,\tau + \sum_{i \in \mathcal{I}_N} J_i^T(q)\mathrm{f}_i, \tag{2.43}$$

with, $M(q)$ is the mass matrix, $C(q,\nu)$ the Coriolis matrix, $G(q)$ the potential force vector, S the matrix with generalized force directions related to the joint torques, τ the joint torques, $\mathcal{I}_N$ the set of closed contacts, f_i the i-th contact wrench, and $J_i(q)$ the Jacobian associated with the i-th contact point. For the iCub, we use

$$S = \begin{bmatrix} 0_{6\times(n_J)} \\ I_{n_J} \end{bmatrix}. \tag{2.44}$$

The first six rows refer to the position and orientation of the root link, while the other rows refer to the joint angles. To keep the notation compact, we will sometimes make use of the vector of generalized bias forces $h(q,\nu)$, simply defined as

$$h(q,\nu) := C(q,\nu)\nu + G(q). \tag{2.45}$$

The matrices and vectors appearing in (2.43) are computed making use of iDynTree [6], accessible in MATLAB via the mex-wholebodymodel[1] interface. This opensource software has been used in other projects, e.g., to create a balancing controller for the iCub [3].

2.3 Contact dynamics

In this work, we consider the iCub in an environment composed by a flat ground (the floor) and a wall, as displayed in Figure 2.2. We aim to model a contact task for the iCub in a realistic, yet relatively simple manner. The environment and contact task are choosen with this aim in mind. In the contact task, the iCub will use the wall and floor to support itself by standing on one leg, using its hand to periodically establish a contact with the wall. This contact task is relativly simple to model, yet challenging to control. The goal of this manuscript is to stabilize

[1]Matlab MEX interface to the iWholeBodyModel interface *https://github.com/robotology/mex-wholebodymodel*

the (hybrid) trajectory of this motion.

The interactions between the robot and the environment can best be described as unilateral holonomic contact constraints (position constraints acting in one direction) with impacts. For simplicities sake, it is assumed that all collisions are inelastic and all contacts are non-slipping. We consider the floor at $z_{floor} = 0$ and a wall at $x_{wall} = 0.4$, both expressed in the absolute frame 1. To limit the number of contact points, the iCub will only use its left hand and leg for contacts.

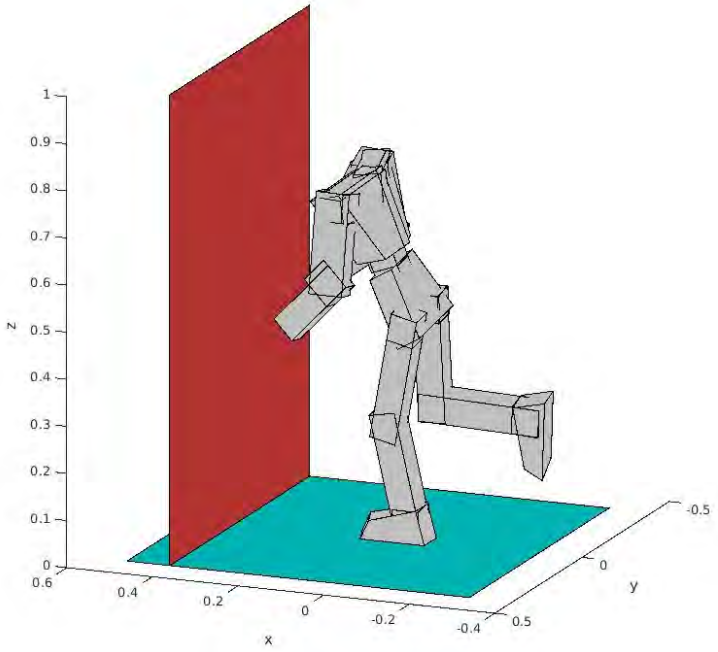


Figure 2.2: The iCub in an environment with a floor and a wall.

2.3.1 Left foot contact

The contact between the left foot and the floor is modeled as a planar contact imposing no sliding conditions. To limit the complexity of modeling complete foot detachment and adhesion to the ground, we only consider situations where the left foot remains in contact with the floor. This allows us to model the normal and friction forces as a net contact wrench at the origin of frame 6, represented by $\mathrm{f}_{foot} = {}_6\mathrm{f}_{foot}$.

Clearly, our assumption that the foot always remains in contact with the floor is only valid if it can be maintained by a physically valid contact wrench. We write the set of valid contact wrenches as $(\mathrm{f}_{foot})_{val}$. The initial position of the foot is such that frame 6 coincides with frame 1, allowing us to write the position constraint and the feasibility condition on the contact wrench as

$$ {}^1H_6 = I_4, \tag{2.46} $$

$$ \text{s.t. } \mathrm{f}_{foot} \in (\mathrm{f}_{foot})_{val}. \tag{2.47} $$

Should condition (2.47) be violated, the dynamic simulation is terminated. As will be explained later, the chosen desired motion has been selected for this not to happen, although we monitor this condition to ensure that the simulation is physically valid in terms of the unilateral nature of the foot constraint. Hereto, the area of the iCub sole can be approximated by a trapezoid, as depicted in Figure 2.3, and is denoted by A_{foot}.

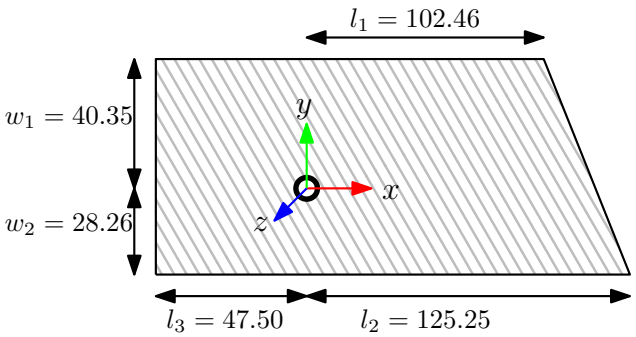


Figure 2.3: The area of the iCub sole can be approximated by a trapezoid. Frame 6 is the application point of the net constraint wrench. The lengths in the figure are expressed in millimeters.

We can write the contact wrench as

$$\mathrm{f}_{foot} = \begin{bmatrix} f_{foot} \\ \omega_{foot} \end{bmatrix}, \tag{2.48}$$

with

$$f_{foot} = \begin{bmatrix} (f_{foot})_x \\ (f_{foot})_y \\ (f_{foot})_z \end{bmatrix} \tag{2.49}$$

and

$$\omega_{foot} = \begin{bmatrix} (\omega_{foot})_x \\ (\omega_{foot})_y \\ (\omega_{foot})_z \end{bmatrix}. \tag{2.50}$$

The set $(\mathrm{f}_{foot})_{val}$ consists of all contact wrenches that can be generated by a force distribution on A_{foot}, with all force components pointing away from the floor (i.e., no adhesion is allowed). This implies that

$$(f_{foot})_z \geq 0. \tag{2.51}$$

Furthermore, the center of pressure (COP) has to lie within the convex hull of the sole area. We assume that this convex hull coincides with A_{foot}. The COP coincides with the zero tipping moment point (ZMP) [14]. The ZMP is defined as the point on the ground where the tipping moment acting on the biped, due to gravity and inertia forces, equals zero, the tipping moment being defined as the component of the moment that is tangential to the supporting surface [14]. Let $p = {}^{6[1]}p$ denote the zero moment point, with

$$p = \begin{bmatrix} p_x \\ p_y \\ 0 \end{bmatrix}. \tag{2.52}$$

We can relate the tipping moment of the net contact wrench to the ZMP, using

$$\begin{bmatrix} (\tau_{foot})_x \\ (\tau_{foot})_y \\ 0 \end{bmatrix} = p^{\wedge} f_z e_3 = \begin{bmatrix} p_y \\ -p_x \\ 0 \end{bmatrix} (f_{foot})_z, \tag{2.53}$$

with e_3 the unit vector in z direction. We can rewrite (2.53) to

$$p = \begin{bmatrix} -(\tau_{foot})_y \\ (\tau_{foot})_x \\ 0 \end{bmatrix} (f_{foot})_z^{-1}. \tag{2.54}$$

Using Figure 2.3 and the condition that the ZMP has to lie in A_{foot}, we can now define the set of valid contact wrenches

$$(\mathrm{f}_{foot})_{val} = \left\{ \mathrm{f}_{foot} \in \mathbb{R}^6 \mid C_{foot}\mathrm{f}_{foot} \leq 0_5 \right\}, \tag{2.55}$$

with

$$C_{foot} = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & -w_2 & -1 & 0 & 0 \\ 0 & 0 & -w_1 & 1 & 0 & 0 \\ 0 & 0 & -l_3 & 0 & 1 & 0 \\ 0 & 0 & -\frac{l_2 w_1 + l_1 w_2}{w_1 + w_2} & \frac{l_2 - l_1}{w_1 + w_2} & -1 & 0 \end{bmatrix}. \tag{2.56}$$

Note that, since we assume a non-slipping situation, all in plane forces and torques are always physically valid, provided that condition (2.51) is fulfilled.

2.3.2 Left hand contact

The contact between the hand and the wall will be modeled as a point contact, as if the iCub is making contact with the wrist. We opt to model the hand contact this way, because the theory for hybrid systems has been designed for simple contacts/impacts, where only one constraint becomes active at a time. The wrist contact point is located at the origin of frame 4. Let $(o_4)_x := ({}^1o_4)_x$ denote the x coordinate of the origin of frame 4. The unilateral constraint for the hand can now be formulated as

$$(o_4)_x \leq x_{wall}. \tag{2.57}$$

If the constraint is closed, a constraint force $f_{hand} := {}_{4[1]}f_{hand}$ is active on the hand. This force contains the normal and tangential forces applied on the left hand. Since the hand is considered a point contact, no contact torques can be applied on the hand. Because of the nature of unilateral contacts, the contact force f_{hand} is only physically valid if the normal component points out from the wall. This results in the set of valid contraint forces

$$(f_{hand})_{val} = \{(f_{hand}) \in \mathbb{R}^3 \mid (f_{hand})_x \leq 0\}. \tag{2.58}$$

If the contact force required to keep the contact closed does not lie in this set, the contact is opened, and no contact forces can be applied until it is closed again. Since we consider a non-slipping situation, any value for the friction forces is physically valid, as long as (2.58) is satisfied.

2.3.3 Hybrid system

For the purpose of applying the hybrid controller, detailed in Chapter 4, the dynamics of the iCub is modeled as a hybrid dynamical system [15]. We introduce the hybrid time (t, j), with t the continuous time and j an event counter (discrete time). We consider two possible modes, one where the left hand is free to move (free motion mode, obeying $\dot{\chi} = F_1(\chi, \tau, t, j)$, $c_1(\chi) = 0_4$, $g^{2\leftarrow 1}(\chi) \geq 0$), and one where the hand is in contact with the wall (contact mode, obeying $\dot{\chi} = F_2(\chi, \tau, t, j)$, $c_1(\chi) = 0_4$, $c_2(\chi) = 0_3$, $g^{2\leftarrow 1}(\chi) \geq 0$). Recall that $\chi = [q, \nu]$ denotes the state and τ denotes the joint torques, which serve as the input of the system. In Figure 2.4, the hybrid system is displayed schematically.

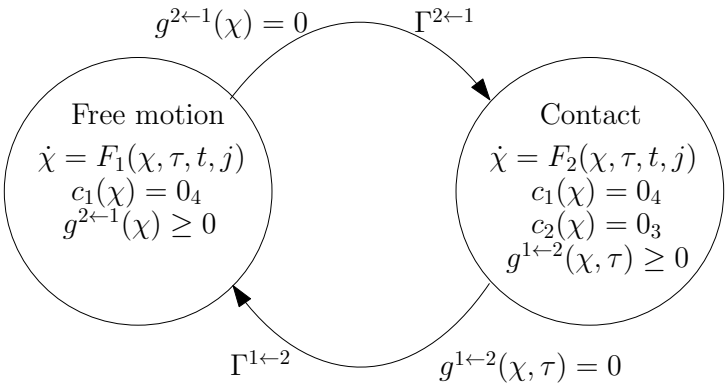


Figure 2.4: A scematic representation of the hybrid dynamical model, using a combination of bilateral constrains, guard function regulated switching, and a velocity reset map to model the iCub's interactions with the environment.

The switching between modes is regulated by guard functions. The system starts in free motion mode under the bilateral contraint $c_1(\chi) = 0_4$ with $c_1(\chi)$ defined as

$$c_1(\chi) := {}^1H_6(\chi) - I_4, \tag{2.59}$$

which fully constrains the left foot. The system switches to contact mode if the hand touches the wall. The guard function for this transition is

$$g^{2\leftarrow1}(\chi) := x_{wall} - (o_4)_x, \tag{2.60}$$

with switching condition

$$g^{2\leftarrow1}(\chi) = 0. \tag{2.61}$$

The unilateral contact constraint is therefore

$$g^{2\leftarrow1}(\chi) \geq 0. \tag{2.62}$$

When switching from free motion mode to contact mode, the system undergoes a state jump as a result of the impulsive forces generated during impact of the hand with the wall. We can model this state jump through a velocity reset map $\Gamma^{2\leftarrow1}$, which we detail in Section 2.3.5.

Once in contact mode, the system obeys $c_1(\chi) = 0_4$ and $c_2(\chi) = 0_3$, with $c_2(\chi)$ defined as

$$c_2(\chi) := {}^1o_4 - [x_{wall}, 0, 0]^T, \tag{2.63}$$

which constraints the position of the left hand. Note that the constraint $c_2(\chi) = 0_3$ implicitly means that $g^{2\leftarrow1}(\chi) = 0$. The system switches back to free motion mode when the contact force needed to maintain the hand constraint is physically invalid. For this, we write our second guard function

$$g^{1\leftarrow2}(\chi, \tau) := -(f_{hand})_x, \tag{2.64}$$

with switching condition

$$g^{1\leftarrow2}(\chi, \tau) = 0. \tag{2.65}$$

The unilateral contact constraint is therefore

$$g^{1\leftarrow2}(\chi, \tau) \geq 0. \tag{2.66}$$

When switching from contact mode to free motion mode, we use a reset map $\Gamma^{1\leftarrow 2}$, with

$$\Gamma^{1\leftarrow 2}(\chi) := \chi. \tag{2.67}$$

No velocity reset is needed in this case, since no impact occurs and, therefore, no impulsive force is generated. In both modes, the system obeys an equation of motion with active bilateral constraints. Together with the guard functions and the velocity reset map, these bilateral constrains behave like unilateral ones. The simulation will be terminated if the left foot contact wrench is physically invalid, regardless of the current mode. The different components of the hybrid dynamical model are explained below.

2.3.4 Constrained dynamical model

In this section, we explain how we maintain bilateral constraints used in the hybrid dynamical model.

Left foot constraint

Consider a 6-dimensional holonomic constraint, acting on the left foot (frame 6) of the iCub, represented by the homogeneous constraint

$${}^{1}H_6(\chi) = \begin{bmatrix} I_3 & 0_{3\times 1} \\ 0_{1\times 3} & 1 \end{bmatrix}. \tag{2.68}$$

We will drop the function arguments for ease of notation. Taking the time derivative of (2.68) results in

$$\frac{\mathrm{d}}{\mathrm{d}t}{}^{1}H_6 = 0_{4\times 4}. \tag{2.69}$$

Recalling that

$${}^{6[1]}\mathrm{v}_{1,6} = {}^{6[1]}X_6\,{}^{6}\mathrm{v}_{1,6} = {}^{6[1]}X_6 \left({}^{1}H_6^{-1}\tfrac{\mathrm{d}}{\mathrm{d}t}{}^{1}H_6\right)^{\vee}, \tag{2.70}$$

from (2.68) and (2.70) we get

$${}^{6[1]}J_{1,6/2}\begin{bmatrix} {}^{2}\mathrm{v}_{1,2} \\ \dot{q}_J \end{bmatrix} = {}^{6[1]}\mathrm{v}_{1,6} = {}^{6[1]}X_6 \left({}^{6}H_1^{-1}\tfrac{\mathrm{d}}{\mathrm{d}t}{}^{1}H_6\right)^{\vee} = 0_6. \tag{2.71}$$

For the sake of symplicity, we will indicate the Jacobian ${}^{6[1]}J_{1,6/2}$ simply as J_{foot} and the velocity vector $[{}^{2}\mathrm{v}_{1,2}^T, \dot{q}_J^T]^T$ as ν. By taking the time derivative of (2.71), that in the newly introduced notation reads $J_{foot}\nu = 0$, one obtains

$$\dot{J}_{foot}\nu + J_{foot}\dot{\nu} = 0_6. \tag{2.72}$$

If this equation is satisfied, and the initial conditions satisfy the constraint (2.68), the constraint will be maintained. The constrained is maintained via a constraint wrench f_{foot} acting on the left foot. From (2.43) and (2.45), we can write the equation of motion for the iCub standing on its left leg as

$$M\dot{\nu} + h = S\tau + J_{foot}^T \mathrm{f}_{foot}. \tag{2.73}$$

Combining (2.72) with (2.73), we obtain

$$\begin{bmatrix} M & J_{foot}^T \\ J_{foot} & 0 \end{bmatrix}\begin{bmatrix} \dot{\nu} \\ -\mathrm{f}_{foot} \end{bmatrix} + \begin{bmatrix} h - S\tau \\ \dot{J}_{foot}\nu \end{bmatrix} = 0. \tag{2.74}$$

By solving (2.74), only the acceleration of the constraint is limited. Therefore, in simulations, numerical errors can cause a drift in the constraint. We can solve this by applying a "spring" and "damper" term as dictated by Baumgarte's stabilization method [16], resulting in the equation

$$\dot{J}_{foot}\nu + J_{foot}\dot{\nu} = -k_p \begin{bmatrix} {}^1o_6 \\ {}^1\Theta_6 \end{bmatrix} - k_d J_{foot}\nu, \tag{2.75}$$

used instead of (2.72). In (2.75), k_p and k_d are positive constants, represent the "spring" and "damper" constants, respectively, and ${}^1\Theta_6$ is the set of Cardan angles (x-y-z) of the left foot. Note that we can use the Cardan angles for Baumgarte's stabilization as a linearization around ${}^1\Theta_6 = 0_3$. In general, one should use the matrix logarithm for mapping the rotational matrix to the corresponding axis-angle representation. Combining (2.75) with the equation of motion results in

$$\begin{bmatrix} M & J_{foot}^T \\ J_{foot} & 0 \end{bmatrix} \begin{bmatrix} \dot{\nu} \\ -\mathrm{f}_{foot} \end{bmatrix} + \begin{bmatrix} h - S\tau \\ \dot{J}_{foot}\nu + k_p \begin{bmatrix} {}^1o_6 \\ {}^1\Theta_6 \end{bmatrix} + k_d J_{foot}\nu \end{bmatrix} = 0. \tag{2.76}$$

as the constrained equation of motion for free motion mode of the hybrid dynamical model: the iCub standing on its left foot. Note that the Baumgarte's stabilization method is only needed for simulations, and that (2.74) is theoretically correct.

Left hand constraint

The left hand constraint can be modeled similar to the left foot constraint. Consider a 3 dimensional bilateral holonomic constraint (point contact), acting on the left hand (frame 4) of the iCub,

$${}^1o_4(t) \equiv {}^1o_4(t_i), \tag{2.77}$$

with t_i denoting the time of impact (activation of the constraint). For the sake of brevity, we will simply write o_4 for ${}^1o_4(t)$ and $o_4(t_i)$ for ${}^1o_4(t_i)$. The first order time derivative of (2.77) is

$$\dot{o}_4 = J_{hand}\nu = 0_3, \tag{2.78}$$

with $J_{hand} := [I_3, 0_{3\times3}]{}^{4[1]}J_{1,4/2}$ the linear velocity part of the Jacobian for the left hand, and ν the generalized velocities of the iCub. Taking the time derivative of (2.78), we obtain

$$\dot{J}_{hand}\nu + J_{hand}\dot{\nu} = 0_3. \tag{2.79}$$

The contact of the hand with the wall generates a contact force $f_{hand} = {}_{4[1]}f$. Note that f_{hand} represents a contact force, not a wrench. The equation of motion for the iCub supporting itself with its left hand and foot can be written as

$$M\dot{\nu} + h = S\tau + J_{foot}\mathrm{f}_{foot} + J_{hand}f_{hand}. \tag{2.80}$$

Combining (2.80) with (2.79) and (2.72) results in the linear system

$$\begin{bmatrix} M & J_{foot}^T & J_{hand}^T \\ J_{foot} & 0 & 0 \\ J_{hand} & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\nu} \\ -\mathrm{f}_{foot} \\ -f_{hand} \end{bmatrix} + \begin{bmatrix} h - S\tau \\ \dot{J}_{foot}\nu \\ \dot{J}_{hand}\nu \end{bmatrix} = 0. \tag{2.81}$$

We can once again ensure numerical stability by using Baungarte's stabilization, resulting in

$$\begin{bmatrix} M & J_{foot}^T & J_{hand}^T \\ J_{foot} & 0 & 0 \\ J_{hand} & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\nu} \\ -\mathrm{f}_{foot} \\ -f_{hand} \end{bmatrix} + \begin{bmatrix} h - S\tau \\ \dot{J}_{foot}\nu + k_p \begin{bmatrix} {}^1o_6 \\ {}^1\Theta_6 \end{bmatrix} + k_d J_{foot}\nu \\ \dot{J}_{hand}\nu + k_p[o_4 - o_4(t_i)] + k_d J_{hand}\nu \end{bmatrix} = 0. \tag{2.82}$$

We use (2.82) to simulate the constrained motion in contact mode of the hybrid dynamical system: the iCub supporting itself on its left foot and left hand. For this application, simply choosing k_p and k_d the same for the foot and the hand works fine.

2.3.5 Velocity reset map

In this section, we explain how we model the dynamics during the collision between the hand and the wall. We assume that the impact duration is infinitesimal. This allows us to model the impact event as an impulsive force applied to the mechanical system, that results in a jump in velocity, quantified by a velocity reset map. The assumption of an infinitesimal impact duration is typical for nonsmooth mechanics [17]. We denote the pre- and post-impact velocity as ν^- and ν^+, respectively. We can write the change in momentum during impact as the sum of all impulsive forces. For the iCub, we have

$$M(\nu^+ - \nu^-) = J_{foot}^T \Lambda_{foot} + J_{hand}^T \lambda_{hand}, \tag{2.83}$$

with $\Lambda_{foot} = {}_{6[1]}\Lambda$ denoting the 6 dimensional force/torque impulse acting on the left foot, and $\lambda_{hand} = {}_{4[1]}\lambda$ denoting the 3 dimensional force impulse acting on the left hand. We can write the pre- and post-impact linear velocity of the iCubs left hand as

$$v_{hand}^- = J_{hand}\nu^- \tag{2.84}$$
$$v_{hand}^+ = J_{hand}\nu^+, \tag{2.85}$$

with $v_{hand} = {}^{4[1]}v_{1,4}$. We employ Newton's impact law [18] to express the result of the impact,

$$(v_{hand}^+)_x = -e(v_{hand}^-)_x, \tag{2.86}$$

with e denoting Newton's coefficient of restitution, and $(v_{hand}^-)_x$ and $(v_{hand}^+)_x$ denoting the pre- and post-impact velocity component normal to the wall. Since we assume that no slip will occur, we also impose the conditions

$$(v_{hand}^+)_y = 0, \tag{2.87}$$
$$(v_{hand}^+)_z = 0, \tag{2.88}$$

for the tangent components of the post impact velocity. If we assume that the contact is inelastic, i.e., we set $e = 0$ in (2.86), we obtain

$$(v_{hand}^+)_x = 0 \tag{2.89}$$

and consequently

$$J_{hand}\nu^+ = 0. \tag{2.90}$$

For the left foot, we can write the pre- and post-impact twists as

$$\mathrm{v}_{foot}^- = J_{foot}\nu^-, \tag{2.91}$$
$$\mathrm{v}_{foot}^+ = J_{foot}\nu^+, \tag{2.92}$$

with $\mathrm{v}_{foot} = {}^{6[1]}\mathrm{v}_{1,6}$. We only consider situations with the left foot constraint closed, i.e., $J_{foot}\nu^- = 0_6$. If we assume, for now, that the left foot constraint remains closed during impact, we can write

$$J_{foot}\nu^+ = 0_6. \tag{2.93}$$

We can combine (2.83), (2.90) and (2.93) into the following linear system,

$$\begin{bmatrix} M & -J_{hand}^T & -J_{foot}^T \\ J_{hand} & 0 & 0 \\ J_{foot} & 0 & 0 \end{bmatrix} \begin{bmatrix} \nu^+ \\ \lambda_{hand} \\ \Lambda_{foot} \end{bmatrix} = \begin{bmatrix} M\nu^- \\ 0_3 \\ 0_6 \end{bmatrix}. \tag{2.94}$$

The linear system (2.94) implicitly defines the velocity reset map $\Gamma^{2\leftarrow 1}$ of our hybrid system in Section 2.3.3. However, we cannot simply assume that the left foot constraint remains closed during impact. As explained in detail in, e.g., [18], two things can happen after impact for a contact that was closed at the pre-impact time t^- (in our case, the foot contact):

1. The contact actively participates in the impact process, i.e.,

$$\Lambda_{foot} \neq 0_6, \tag{2.95}$$
$$\mathrm{v}^+_{foot} = 0_6. \tag{2.96}$$

2. The contact is superfluous, i.e.,

$$\Lambda_{foot} = 0_6, \tag{2.97}$$
$$(\mathrm{v}^+_{foot})_z \geq 0. \tag{2.98}$$

The first option can only occur if the impulse required to keep the constraint closed is physically valid, i.e.,

$$\Lambda_{foot} \in (\Lambda_{foot})_{val}. \tag{2.99}$$

Since Λ_{foot} is the time integral of an impulsive impact force, we can use the set of valid contact forces $(\mathrm{f}_{foot})_{val}$ as defined in (2.56) to formulate $(\Lambda_{foot})_{val}$. Recall that

$$(\mathrm{f}_{foot})_{val} = \left\{ \mathrm{f}_{foot} \in \mathbb{R}^6 \mid C_{foot}\mathrm{f}_{foot} \leq 0_5 \right\}. \tag{2.100}$$

If we assume that f_{foot} is constant over the infinitely small timeframe δt, we can write

$$\mathrm{f}_{foot}\delta t = \Lambda_{foot} \tag{2.101}$$
$$\mathrm{f}_{foot} = \Lambda_{foot}\frac{1}{\delta t}. \tag{2.102}$$

This allows us to formulate the set $(\Lambda_{foot})_{val}$ as

$$(\Lambda_{foot})_{val} = \left\{ \Lambda_{foot} \in \mathbb{R}^6 \mid C\Lambda_{foot} \leq 0_5 \right\}. \tag{2.103}$$

Since the two scenario's (active or passive constraint) are mutually exclusive, we can conclude that the constraint is active during impact if ${}_{6[1]}\Lambda_{foot} \in {}_{6[1]}(\Lambda_{foot})_{val}$. Should this condition not be fullfilled, the foot constraint will open, and the simulation is to be terminated.

Now that we have a dynamical model suitable for the simulation of the contact task, we can move on to the generation of the reference motion. The next chapter focusses on this topic. It presents a reference motion trajectory and a controller capable of generating this feasible trajectory.

Chapter 3

Hybrid trajectory tracking applied to a humanoid robot model

In this chapter, we present a reference spreading (RS) based hybrid trajectory tracking controller for a humanoid robot model. As explained in Chapter 1, we will in particular consider the model of the iCub robot. The RS hybrid controller is based on the hybrid trajectory tracking controller presented in [7] and [8], which we will first briefly review in this chapter. Thereafter, we present an optimal constrained controller for the iCub, based on the balancing controller of [3]. We will use the optimal constrained controller just to generate a feasible reference trajectory which we subsequently aim to track using the RS hybrid controller. Once we have a reference, we will design the RS state feedback controller for the application of interest, i.e., the iCub.

3.1 Reference spreading control for hybrid systems

In dynamical systems, motions with hard impacts can be difficult to control. When using a traditional feedback-feedforward trajectory tracking controller and standard notion of tracking error (i.e., the difference between the actual and desired state at the current moment in time), large tracking errors occur if the expected and actual impact times do not match [19]. Consider, for example, a system with a later impact time than expected. At the expected impact time, the reference has a jump in velocity. If this velocity jump is not matched by the state, the tracking error will instantaneously increase. After the expected impact time, the controller tries to track the post impact reference while no actual impact has occured, converging the error back to zero. When the actual impact does occur, the state has a velocity jump, causing another instantaneous increase in tracking error. In certain applications, for example, humanoid robots, these unexpected velocity jumps can destabilize the system. Using a reference spreading based state controller can solve this problem.

Reference spreading was first introduced in [7], and creates so-called extended reference trajectories to define a novel notion of tracking error. Consider a hybrid system that switches from mode a to mode b at an event time t_i, close to the expected impact time t_i^*. A conventional control strategy has a reference in mode a for $t \leq t_i^*$, and a reference in mode b for $t > t_i^*$. An RS hybrid controller has a reference in mode a and one in mode b around t_i^*, and alternates between these reference trajectories based on the current (actual) mode of the closed loop system. The reference trajectories about t_i^* are obtained by integrating the vector field of the system for each mode beyond the expected event time. We call the extended references before and after t_i^* the ante and post event extensions, respectively.

To get a better understanding of an RS hybrid controller, we apply it on a simple, one-dimensional mass system with position x, state vector $\chi = [x, \dot{x}]^T$, mass $m = 1$, input force u,

and equations of motion given by

$$\dot{\chi} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \chi + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u \tag{3.1}$$

We again make use of the hybrid time (t, j), as done in Chapter 2, indicating with j the event counter ($j = 0$ at $t = t_0$). The mass can move in a one dimensional space with a wall at $x_{wall} = 0$. When the mass makes contact with the wall, a fully elastic collision occurs. The system can be modeled as a hybrid system with one mode whose continuous dynamics is described by (3.1). The guard function for this system is given by

$$g_1(\chi) = x - x_{wall}. \tag{3.2}$$

The system follows the continuous dynamics for $g_1(\chi) > 0$ and a collision event occurs when $g_1(\chi) = 0$. This collision can be modeled as a discrete event with the velocity reset map (Newton's impact law)

$$\dot{x}^+ = -\dot{x}^-, \tag{3.3}$$

with $\dot{x}^-$ and $\dot{x}^+$ the pre- and post-impact velocity, respectively. We want the mass to perform a desired "periodic bouncing" motion, with reference $\alpha(t)$, defined as

$$\alpha(t) := \begin{bmatrix} \alpha_x(t) \\ \alpha_v(t) \end{bmatrix}, \tag{3.4}$$

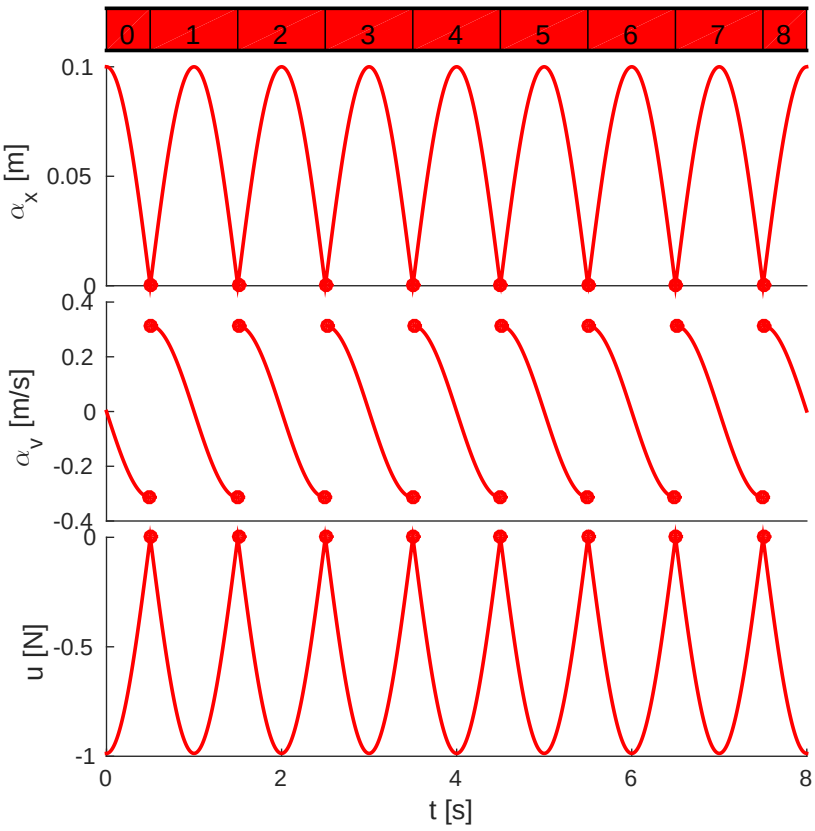


Figure 3.1: The reference trajectory $\alpha(t) = [\alpha_x(t), \alpha_v(t)]^T$ and feedforward $\mu(t)$ of the mass system. The dots in the graph denote the impact events, and the blocks above the graph denote the number of events that happened.

with $\alpha_x(t)$ the position reference and $\alpha_v(t)$ the velocity reference. The feedforward of this motion is denoted with $\mu(t)$. Both the reference $\alpha(t)$ and feedforward $\mu(t)$ are displayed in Figure 3.1. The dots in the graph denote the impact events, and the blocks above the graph denote the reference event counter.

The key idea behind RS hybrid control is to create a hybrid reference that has two reference signals around each impact; one for the pre-impact mode and one for the post-impact mode. The RS hybrid controller alternates between the reference signals based on the current (actaul) mode of the closed loop system. The dual reference around each impact is obtained by extending the pre-impact reference beyond the extended impact time and extending the post-impact reference to before the impact time.

There are different ways to obtain the extended reference signal, among which the simplest one is to continue integrating the vector field, both forward in time and backwards in time, as if the event never happened. The vector field is integrated using the feedforward $\mu(t)$ of the original reference trajectory [7].

We can use this reference extension technique on the reference of the one-dimensional mass system to obtain the hybrid feedforward

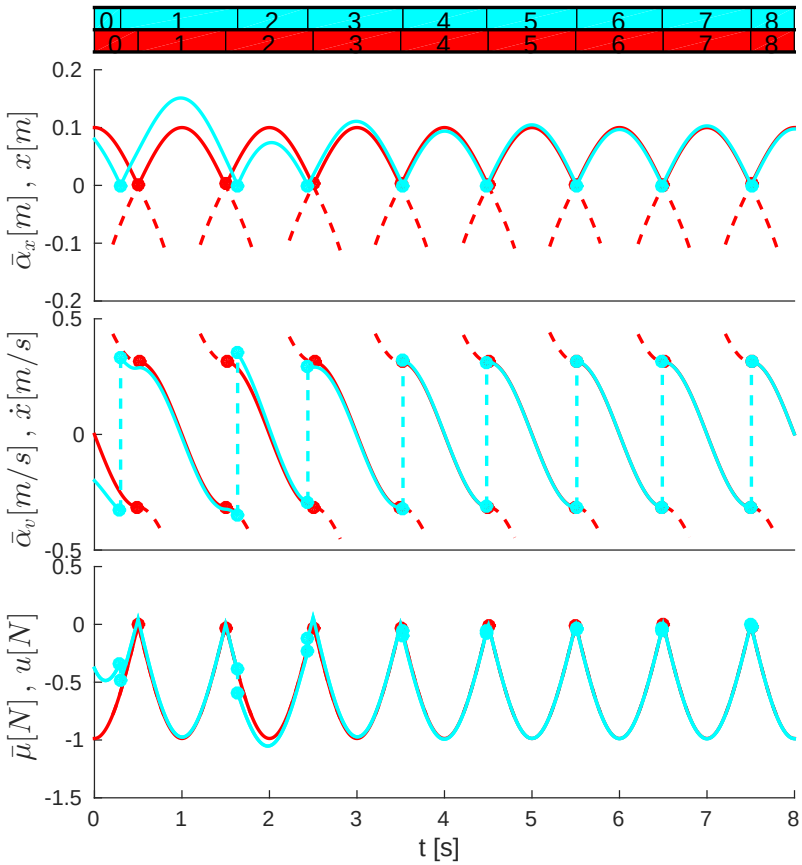


Figure 3.2: A simulation of a one dimensional actuated mass with the RS hybrid controller. The red lines represent the reference signals, with the extensions in dashed lines. A trajectory of the closed-loop system for perturbed initial conditions is shown in light blue

$$\bar{\mu}(t,j) = \mu(t) \tag{3.5}$$

and hybrid reference trajectory

$$\bar{\alpha}(t,j) := \begin{bmatrix} \bar{\alpha}_x(t,j) \\ \bar{\alpha}_v(t,j) \end{bmatrix}, \tag{3.6}$$

with $\bar{\alpha}_x(t,j)$ the hybrid position reference and $\bar{\alpha}_v(t,j)$ the hybrid velocity reference. The extended (hybrid) reference $\bar{\alpha}(t,j)$ and hybrid feedforward $\bar{\mu}(t,j)$ can be seen in Figure 3.2. In this figure, the red lines represent the reference signals, with the extensions shown using dashed red lines. The red blocks above the figure denote the reference event counter.

Figure 3.2 also shows the tracking results of the system simulated with perturbed initial conditions

$$\chi_0 = \chi(t_0, 0) = \begin{bmatrix} 0.08 \\ -0.2 \end{bmatrix} \tag{3.7}$$

and control input

$$u(t,j) = \bar{\mu}(t,j) - \begin{bmatrix} k_p & k_d \end{bmatrix} \left(\bar{\chi}(t,j) - \bar{\alpha}(t,j) \right), \tag{3.8}$$

with $k_p = 0.5$ and $k_d = 3$ denoting feedback gains. The tracking results are diplsayed in light blue, with the light blue blocks above the figure denoting the event counter of the closed loop system (system event counter).

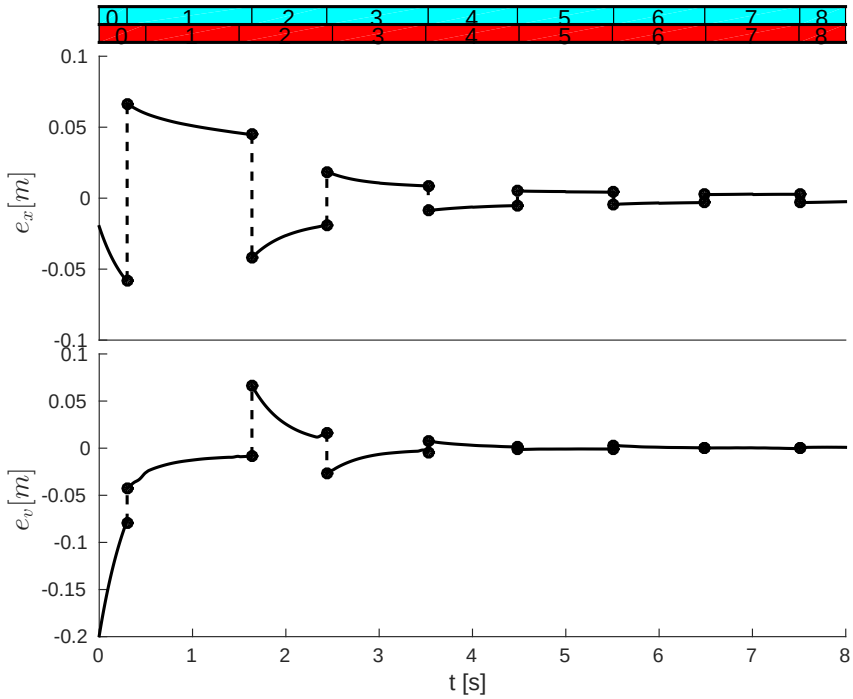


Figure 3.3: The hybrid tracking error of the RS hybrid controller on a simulation of a one dimensional mass system.

As can be seen in Figure 3.2, the state trajectory converges to the reference trajectory. This can be seen more clearly in the tracking error plotted in Figure 3.3, where the error converges

to zero over time. Note that we consider a different notion of tracking error than is common in literature. When regarding an RS hybrid controller, we use an unconventional notion of tracking error, taken from [7], defined as

$$e(t,j) = \begin{bmatrix} e_x(t,j) \\ e_v(t,j) \end{bmatrix} := \chi(t,j) - \bar{\alpha}(t,j). \tag{3.9}$$

When we mention the tracking error in this chapter, we refer to this definition. When we mention the traditional or conventional tracking error, we refer to the tracking error defined as

$$e_s(t,j) = \begin{bmatrix} (e_x)_s(t,j) \\ (e_v)_s(t,j) \end{bmatrix} := \chi(t,j) - \alpha(t). \tag{3.10}$$

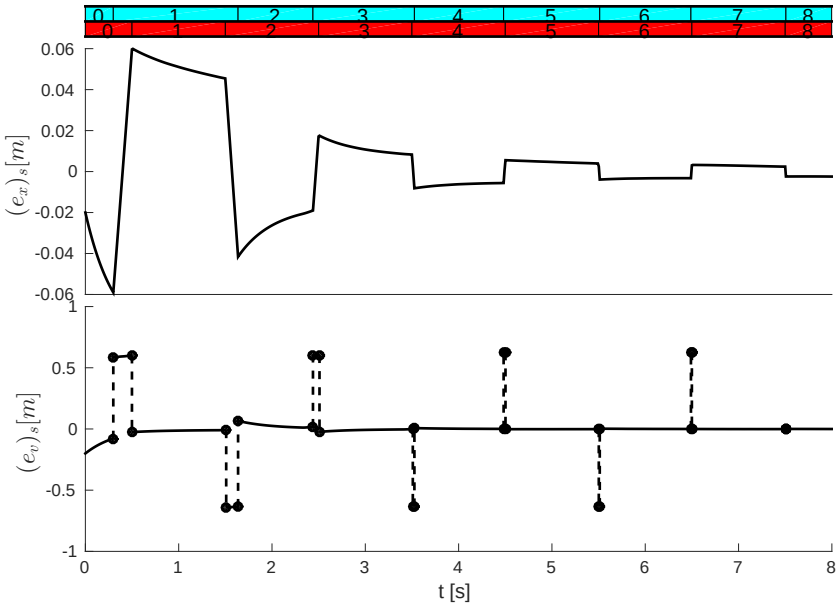


Figure 3.4: The traditional tracking error of the RS hybrid controller on a simulation of a one dimensional mass system.

Figure 3.4 displays the traditional tracking error of the simulation of the one-dimensional mass system. The figure clearly displays large jumps in velocity error at the expected and actual event times, which makes the traditional tracking error less than ideal for dealing with hard impacts.

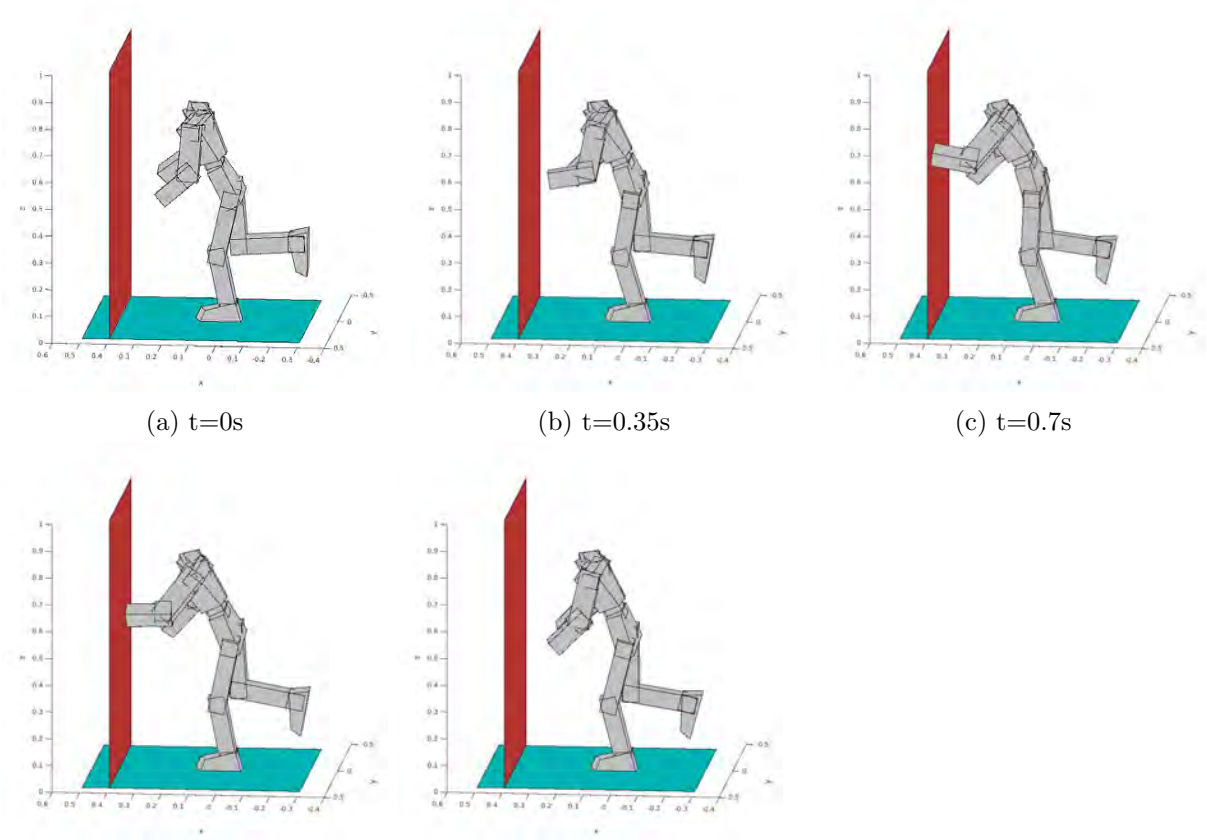
Now that we explained the principles of the RS hybrid controller, we go on by constructing a sensible (extended) reference trajectory for the iCub to trajectory. To this end, we make use of the optimal constrained controller that will be detailed next.

3.2 Optimal constrained controller

To generate the iCub joint reference, we use an optimal constrained controller, based on the balancing controller presented in [3]. Optimal constrained controllers are often used in robotics, and are based on solving online a constrained optimization problem to find a control input that satisfies the state and input constraints. The constraints in the optimization problem are

typically there to prevent, e.g., unilateral contacts from being broken or to avoid actuator saturations. The optimal constrained controller presented in this section is designed for a specific motion task of the iCub. We opt for a contact task that generates hard impacts, yet is relatively easy to model, using only the robot's left hand and foot to interact with the environment.

Consider Figure 3.5. The contact task starts with the robot balancing on its left foot. The robot gradualy moves itself forward with its left hand, until the hand hits a wall in front of the robot. Once the hand makes contact with the wall, the robot pushes itself away from the wall, back to standing on one foot (with the left hand detached from the wall again). From the standing position, the robot repeats the last two steps indefinitely, reaching back to the wall, and pushing itself away again. Five stages of the iCub's motion task are depicted in Figure 3.5



(a) t=0s (b) t=0.35s (c) t=0.7s

(d) t=1.0s (e) t=1.2s

Figure 3.5: Snapshots of the motion task of the robot.

To obtain the reference trajectory, we make use of a task-space optimal constrained controller, since these kind of controllers are well suited for maintaining the balance of a humanoid robot. We use two versions of the optimal constrained controller, one for the iCub balancing on its left foot (free motion mode), and one for he iCub leaning against the wall (contact mode). We discuss these two versions separately.

3.2.1 Optimal constrained controller: free motion mode

The optimal constrained controller for the free motion mode is based on controlling the center of mass (CoM) position, which is directly linked to the robot's balance. Additionally, the left hand position is controlled as a means to perform a desired motion, e.g., the iCub reaching to

the wall. While controlling the position of the CoM and left hand, the controller constrains the joint torques such that the generated contact force on the foot maintains the foot-floor contact. To ensure that the iCub shifts its balance as desired, the controller tracks a reference $\beta_{CoM} \in \mathbb{R}^3$ for the center of mass (frame 7) of the iCub. Let $o_{CoM} = {}^1o_7$ denote the position of frame 7, located at the center of mass of the iCub. The desired acceleration of the CoM can be computed with a feedback law

$$(\dot{v}_{CoM})_{des} = \ddot{\beta}_{CoM} - k_{p,CoM}\left(o_{CoM} - \beta_{CoM}\right) - k_{d,CoM}\left(v_{CoM} - \dot{\beta}_{CoM}\right), \tag{3.11}$$

with $k_{p,CoM}$ and $k_{d,CoM}$ positive constants. In this section, we will write $\mathrm{v}_{CoM} = (v_{CoM}; \omega_{CoM})$ to denote ${}^7\mathrm{v}_{1,7} = ({}^7v_{1,7}; {}^7\omega_{1,7})$, respectively. Recall that the orientation frame [7] is equal to [1], which means that the left trivialized twist (body twist) and the mixed twist of frame 7 are identical. We define the centroidal momentum

$$P := \begin{bmatrix} P_v \\ P_\omega \end{bmatrix}. \tag{3.12}$$

The time derivative of the centroidal momentum is equal to the sum of all external wrenches, namely

$$\dot{P} = \begin{bmatrix} m\dot{v}_{CoM} \\ \dot{P}_\omega \end{bmatrix} = {}_7X^{6[1]}\mathrm{f}_{foot} + F_g, \tag{3.13}$$

with $\mathrm{f}_{foot} := {}_{6[1]}\mathrm{f}$ the contact wrench at the left foot sole, ${}_7X^{6[1]}$ the wrench transformation matrix from frame 6[1] (left foot) to frame 7 (CoM), m the total mass of the iCub and F_g the gravitational wrench vector . To stabilize the angular centroidal momentum, we employ the feedback law

$$(\dot{P}_\omega)_{des} = -P_\omega. \tag{3.14}$$

By employing (3.13) we obtain a function for the desired contact wrench $(\mathrm{f}_{foot})_{des}$ that is required to achieve the desired acceleration of the center of mass, namely,

$$(\mathrm{f}_{foot})_{des} = {}_{6[1]}X^7\left(\begin{bmatrix} m(\dot{v}_{CoM})_{des} \\ (\dot{P}_\omega)_{des} \end{bmatrix} - F_g\right). \tag{3.15}$$

As discusses in Section 2.3.1, not all contact wrenches are physically valid. The set of valid contact wrenches are

$$(\mathrm{f}_{foot})_{val} = \left\{\mathrm{f}_{foot} \in \mathbb{R}^6 \mid C_{foot}\mathrm{f}_{foot} \leq 0\right\}, \tag{3.16}$$

with C_{foot} as in (2.56). To ensure that the foot constraint remains satisfied, we calculate a valid desired constraint wrench f^*_{foot} by solving the quadratic problem

$$\min_{\mathrm{f}^*_{foot}} \quad \left\|\mathrm{f}^*_{foot}T - (\mathrm{f}_{foot})_{des}\right\|^2 \tag{3.17}$$

$$\text{s.t.} \quad C^*_{foot}\mathrm{f}^*_{foot} \leq 0, \tag{3.18}$$

with C^*_{foot} the adjusted constraint matrix, given by

$$C^*_{foot} = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & -w_2k_s & -1 & 0 & 0 \\ 0 & 0 & -w_1k_s & 1 & 0 & 0 \\ 0 & 0 & -l_3k_s & 0 & 1 & 0 \\ 0 & 0 & -\frac{l_2w_1+l_1w_2}{w_1+w_2}k_s & \frac{l_2-l_1}{w_1+w_2} & -1 & 0 \end{bmatrix}, \tag{3.19}$$

where k_s denotes a safety margin. We take $k_s = 0.75$ in the rest of this work.

To make the iCubs hand reach for the wall, we formulate a reference $\beta_{hand} \in \mathbb{R}^3$ for the left hand frame origin $o_{hand} := {}^1o_4$. We can write the desired acceleration of the left hand frame as

$$(\dot{v}_{hand})_{des} = \ddot{\beta}_{hand} - k_{p,hand}\left(o_{hand} - \beta_{hand}\right) - k_{d,hand}\left(v_{hand} - \dot{\beta}_{hand}\right), \tag{3.20}$$

with $v_{hand} = {}^{4[1]}v_{1,4}$ the linear velocity of the left hand, and $k_{p,hand}$ and $k_{d,hand}$ positive constants. For the linear acceleration of the left hand frame, we can write

$$\dot{v}_{hand} = J_{hand}\dot{\nu} + \dot{J}_{hand}\nu, \tag{3.21}$$

with $J_{hand} := [I_3, 0_{3\times3}]{}^{4[1]}J_{1,4/2}$, that is the linear velocity part of the geometric Jacobian ${}^{4[1]}J_{1,4/2}$ associated with the left hand.

Under the assumption that the left foot constraint remains closed, we can write

$$\dot{v}_{foot} = J_{foot}\dot{\nu} + \dot{J}_{foot}\nu = 0_6, \tag{3.22}$$

with $J_{foot} := {}^{6[1]}J_{1,6/2}$ the geometric Jacobian associated with the left foot. The equation of motion of the iCub standing on one foot is

$$M\dot{\nu} + h = S\tau + J_{foot}^T \mathrm{f}_{foot}. \tag{3.23}$$

We can find the joint control torque τ by solving the system

$$\begin{aligned}
S\tau =& - M\dot{\nu} - h + J_{foot}^T \mathrm{f}_{foot} && (3.24)\\
\text{s.t.}\quad & \mathrm{f}_{foot} = \mathrm{f}^*_{foot}, && (3.25)\\
& J_{hand}\dot{\nu} + \dot{J}_{hand}\nu - (\dot{v}_{hand})_{des} = 0_3, && (3.26)\\
& J_{foot}\dot{\nu} + \dot{J}_{foot}\nu = 0_6, && (3.27)
\end{aligned}$$

where f^*_{foot} is computed by solving (3.17) and (3.18). When solving this system for τ, it has more unknown variables than constraint equations and, therefore, it has infinitly many possible solutions. If we denote the pseudo inverse of a matrix A as A^+, we can calculate one such solution as

$$\begin{aligned}
\tau^* = &\left(\begin{bmatrix} J_{foot} \\ J_{hand} \end{bmatrix} M^{-1} S\right)^+ && (3.28)\\
&\left(\begin{bmatrix} J_{foot} \\ J_{hand} \end{bmatrix} M^{-1} h - \begin{bmatrix} J_{foot} \\ J_{hand} \end{bmatrix} M^{-1}\ J_{foot}^T \mathrm{f}^*_{foot} - \begin{bmatrix} \dot{J}_{foot} \\ \dot{J}_{hand} \end{bmatrix} \nu + \begin{bmatrix} 0_{6\times1} \\ (\dot{v}_{hand})_{des} \end{bmatrix}\right).
\end{aligned}$$

Note that this solution gives an equal priority to the conditions (3.25), (3.26) and (3.27) which can be an issue if the conditions cannot be fulfilled at the same time. A controller with a task hierarchy, as provided in ,e.g., [20], could provide a solution in those cases. Here, however, we only employ this controller to generate a reference trajectory of a repeating motion sequence. We circumvent this potential problem by choosing the proper trajectories for the center of mass and left hand.

3.2.2 Optimal constrained controller: contact mode

The optimal constrained controller for the contact mode of the iCub is similar to the controller for the free motion mode. It is based on controlling the center of mass (CoM) position, which is directly linked to the robots balance. Since we have 9 constraints (3 on the hand, 6 on the

foot) and 6 unactuated degrees of freedom (the system is overactuated), we can achieve the same motion with infinitly many joint torque combinations. We use this to our advantage, by choosing the torques such that the foot force constraint will be maintained to ensure the feasibility of the motion. To ensure that the iCub shifts its balance as desired, we track a reference β_{CoM} for the CoM position. We calculate the desired acceleration of the center of mass from a reference trajectory β_{CoM}, using

$$(\dot{v}_{CoM})_{des} = \ddot{\beta}_{CoM} - k_{p,CoM}\left(o_{CoM} - \beta_{CoM}\right) - k_{d,CoM}\left(v_{CoM} - \dot{\beta}_{CoM}\right), \tag{3.29}$$

We can write the centroidal dynamics for the iCub with the left hand in contact as

$$\begin{bmatrix} m\dot{v}_{CoM} \\ \dot{P}_{\omega} \end{bmatrix} = {}_7X^{6[1]}\mathrm{f}_{foot} + {}_7X^{4[1]}\begin{bmatrix} I_3 \\ 0_{3\times3} \end{bmatrix} f_{hand} + F_g. \tag{3.30}$$

Since we have 9 constraints and 6 free floating degrees of freedom, we can find infinitlymany combinations of $(f_{foot})_{des}$ and $(f_{hand})_{des}$ that are in accordance with the desired acceleration of the CoM. To prevent the left foot from detaching from the ground, we opt for a solution where the foot does not provide any torque around x and y, namely

$$\begin{bmatrix} (\mathrm{f}_{foot})_{des} \\ (f_{hand})_{des} \end{bmatrix} = \left(\begin{bmatrix} {}_7X^{6[1]} \\ {}_7X^{4[1]}\begin{bmatrix} I_3 \\ 0_{3\times3} \end{bmatrix} \end{bmatrix} \begin{bmatrix} I_3 & 0_{3\times2} & 0_{3\times3} & 0_{3\times1} \\ 0_{2\times3} & 0_{2\times2} & 0_{2\times3} & 0_{2\times1} \\ 0_{3\times3} & 0_{3\times2} & I_3 & 0_{3\times1} \\ 0_{1\times3} & 0_{1\times2} & 0_{1\times3} & 0 \end{bmatrix} \right) m({}^7\dot{\mathrm{v}} - F_g). \tag{3.31}$$

This choise warrants that, unless $(f_{foot})_z < 0$, $(\mathrm{f}_{foot})_{des}$ is physically valid. We can calculate the control torque by solving for τ the equations of motion

$$M\dot{\nu} + h = S\tau + J_{foot}^T\mathrm{f}_{foot} + J_{hand}^T f_{hand}, \tag{3.32}$$

satisfying the constraints

$$\mathrm{f}_{foot} = (\mathrm{f}_{foot})_{des}, \tag{3.33}$$
$$f_{hand} = (f_{hand})_{des}, \tag{3.34}$$

and

$$\begin{bmatrix} J_{foot} \\ J_{hand} \end{bmatrix}\dot{\nu} = -\begin{bmatrix} \dot{J}_{foot} \\ \dot{J}_{hand} \end{bmatrix}\nu. \tag{3.35}$$

This system has an infinitly many of possible solutions. We choose the solution

$$\tau^* = \left(\begin{bmatrix} J_{foot} \\ J_{hand} \end{bmatrix} M^{-1}S\right)^+ \left(\begin{bmatrix} J_{foot} \\ J_{hand} \end{bmatrix} M^{-1}h - \begin{bmatrix} J_{foot} \\ J_{hand} \end{bmatrix} M^{-1} \begin{bmatrix} J_{foot} \\ J_{hand} \end{bmatrix}^T \mathrm{f}_{foot}^* - \begin{bmatrix} \dot{J}_{foot} \\ \dot{J}_{hand} \end{bmatrix}\nu\right), \tag{3.36}$$

similarly as done in (3.28).

3.2.3 Minimizing joint motion

In both the free motion and contact mode, the control torque τ^* calculated by the optimal constrained controller is one of the infinitely many solutions to an underdetermined problem. This can result in a solution with unwanted levels of joint movement. For example, the iCub could theoretically reach for the wall with its left hand, while simultaniously waving with its right hand. While this motion would fulfill the control objectives, it is needlesly complicated. To prevent such needlesly complicated motions, we aim to reduce excessive joint motion by

adding a postural task in the nullspace of the left hand and left foot tasks. We can compute a matrix N whose image is the nullspace with, e.g., the pseudo inverse of the hand and foot jacobians,

$$N = I_{25} - \begin{bmatrix} J_{foot}S^T \\ J_{hand}S^T \end{bmatrix}^+ \begin{bmatrix} J_{foot}S^T \\ J_{hand}S^T \end{bmatrix} \tag{3.37}$$

The desired joint acceleration for the postural task, $(\ddot{q}_J)_{des}$ is calculated with a feedback law

$$(\ddot{q}_J)_{des} = -k_{p,pos}(q_J - (q_J)_{init}) - k_{d,pos}\dot{q}_J, \tag{3.38}$$

with $k_{p,pos}$ and $k_{d,pos}$ positive constants, and $(q_J)_{init}$ denoting the joint angles at the start of simulation. Consider the general equation of motion (2.43) of the iCub

$$M\dot{v}^* + h = S\tau^* + \sum_{i \in \mathcal{I}_N} J_i^T \mathrm{f_i}. \tag{3.39}$$

If we add a torque pertubation $\Delta\tau$ in the nullspace of the current contact forces, the only direct effect of this pertubation is a pertubation in the acceleration of the iCub. Both the generalized bias forces $h(q, \nu)$ and the mass matrix $M(q)$ are dependent of the current state only, meaning that they are not directly affected by a pertubation in the input torque (they are indirectly effected through the influence of the torque pertubation on the derivative of the state). Since the torque pertubation is applied in the nullspace of the current contact forces, the contact forces are by definition not influenced directly. Therefore, we obtain

$$M(\dot{v}^* + \Delta\dot{v}) + h = S(\tau^* + N\Delta\tau) + \sum_{i \in \mathcal{I}_N} J_i^T \mathrm{f_i}, \tag{3.40}$$

and, consequently,

$$SN\Delta\tau = M\Delta\dot{v}, \tag{3.41}$$

which can be rewritten as

$$N\Delta\tau = (S^T M^{-1} S)^{-1} S^T \Delta\dot{v}. \tag{3.42}$$

We would like to take $\Delta\dot{v} = S(\ddot{q}_J)_{des}$ and solve (3.42) for $\Delta\tau$. However, (3.42) is only solvable if $(S^T M^{-1} S)^{-1} S^T \Delta\dot{v}$ lies in the nullspace in which we projected $\Delta\tau$. To enforce this condition, we take

$$\Delta\dot{v} = S(S^T M^{-1} S) N (S^T M^{-1} S)^{-1} (\ddot{q}_J)_{des}, \tag{3.43}$$

which results in

$$(S^T M^{-1} S)^{-1} S^T \Delta\dot{v} = N(S^T M^{-1} S)^{-1} (\ddot{q}_J)_{des}, \tag{3.44}$$

and subsequently,

$$\Delta\tau = (S^T M^{-1} S)^{-1} (\ddot{q}_J)_{des}. \tag{3.45}$$

We can use (3.45) to apply the control objective from (3.38) in the nullspace of the tasks of the optimal constraint controller, resulting in

$$\tau = \tau^* + N(S^T M^{-1} S)^{-1} (\ddot{q}_J)_{des}. \tag{3.46}$$

Now that we have made an optimal constrained controller for the iCub, we can generate the motion reference.

3.3 Generating the state and input reference trajectory

The iCub is a complex, highly unstable dynamical system. The RS hybrid controller in the form as explained in [7] and [8] has no notion of the balance of the iCub and is therefore not suitable for tracking any arbitrary trajectory on the iCub. Instead, the reference trajectory has to be designed to mainting the balance of the robot. The supporting foot of the robot has to stay in contact with the ground at all times, meaning that the reference should be designed such that this condition is fulfilled at all times. This makes the generation of the reference and its extensions, as needed in the RS hybrid controller, quite challenging.

We use the optimal constrained controller, detailed in the previous section, to generate the joint reference and feedforward, and the corresponding extentions. We consider a periodic motion in which the system switches between two modes. The trajectory segments corresponding to each mode will be treated separately. Moreover, the trajectory segment for free motion will be partitioned in a segment where the robot moves towards the wall to establish contact and a segment where it moves back to the initial condition (for periodicity) after breaking contact. We discuss the different subtasks of the desired motion next.

3.3.1 Reaching for the wall

For the first part of the desired motion, we design the task (i.e., the Cartesian trajectories) for the hand and center of mass such that the hand moves towards and beyond the wall, while the CoM slightly moves to the wall. The trajectories for the hand and CoM, denoted β_{hand} and β_{CoM}, respectively, can be seen in Figure 3.6. In this trajectory, the hand is expected to collide with the wall at $\tau_1 = 0.7s$. Note that this task instructs the hand to move upwards as well, to allow for a more natural motion. The CoM task is designed to keep the hips at a constant height.

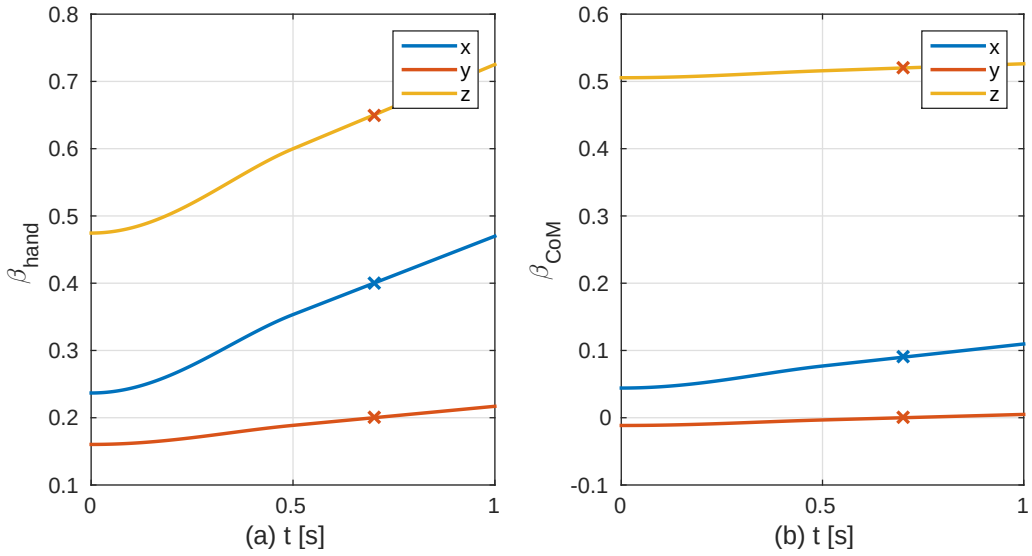
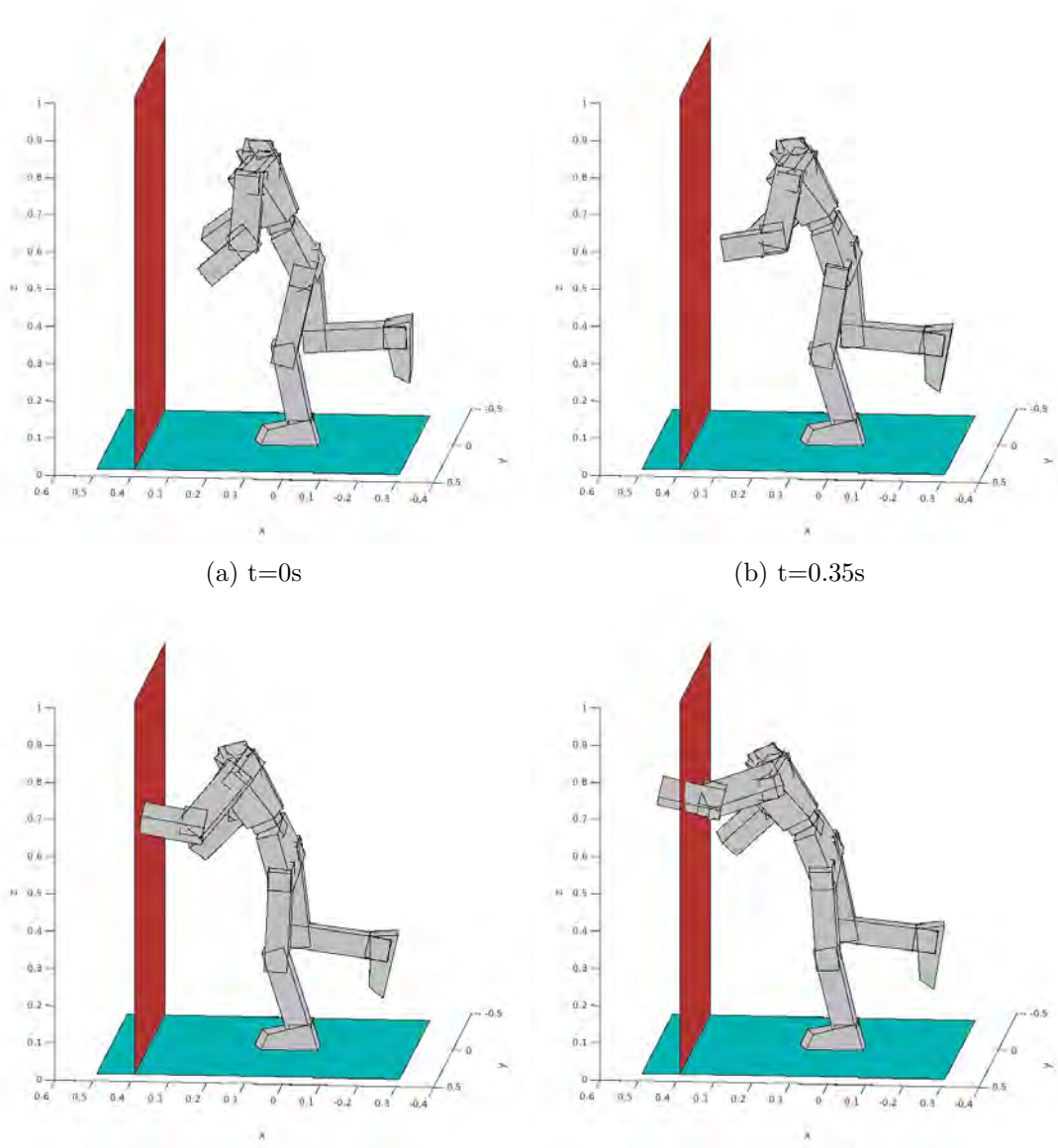


Figure 3.6: Desired Cartesian coordinates for the hand and CoM for the iCub moving its hand towards the wall. The expected time of impact is displayed as an x in the graph.

We use the optimal constrained controller as detailed in Section 3.2.1 to simulate a motion tracking the tasks of Figure 3.6. For this simulation, we take $k_{p,hand} = 100$, $k_{d,hand} = 10$, $k_{p,CoM} = 400$, and $k_{d,CoM} = 10$. At some time $(t_i)_1$ during the simulation, the hand will collide with the wall. After this impact time, we continue the simulation for 0.3s, as if the collision

never happened. This will create a post-impact motion (as the impact did not happen), that will serve as the post-event extension for the hybrid controller reference. Figure 3.7 displays snapshots of the resulting iCub movement.



(a) t=0s

(b) t=0.35s

(c) t=0.7s, time of impact

(d) t=0.9s, post impact time

Figure 3.7: Snapshots of the first part of the iCub desired motion: the iCub reaches for the wall. Note that snapshot (d) is taken from the simulation used to generate the post-event reference extension to be used in the hybrid controller, and is generated assuming there is no wall.

3.3.2 Pushing away from the wall

During the second part of the desired reference motion, the left hand is in contact with the wall (we assume the contact to be nonslipping). Therefore, we specify the desired task just as the Cartesian coordinates of the CoM. This trajectory is such that the iCub pushes against the wall and eventually brakes contact. Due to the nature of the optimal constrained controller as detailed in Section 3.2.2, breaking the hand contact can be achieved by specifying a desired CoM motion towards the hand, resulting in a pulling force on the hand. The desired reference motion starts at τ_1 (the impact time of the previous motion subsequence), and the hand is expected

to break contact at $t = 0.86s$. Figure 3.8 displays the CoM task used for this simulation. Note that, unlike Figure 3.6, no desired trajectory for the left hand is displayed, since no such reference is used in the optimal constrained controller in contact mode.

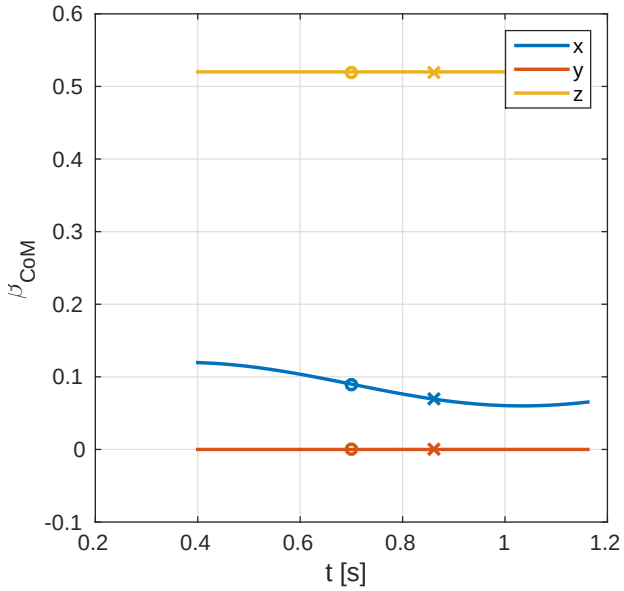


Figure 3.8: The task trajectories for the CoM for the iCub pushing itself from the wall. The start time and expected break of contact time are displayed as an o and x in the graph, respectively.

Note that the task is defined before the time of impact τ_1 of the hand with the wall (around t=0.7s). We use this part of the desired reference motion to generate the ante-event motion reference, to be used in the hybrid controller, using a reverse time simulation. The reverse time simulation starts at the impact time τ_1, using the post impact state as initial condition. If we use positive feedback terms, the optimal constrained controller has an error that converges to zero over time. If we go back in time, however, this error will diverge from zero. This means that using the optimal constrained controller in a reverse time simulations would result in unstable behaviour.

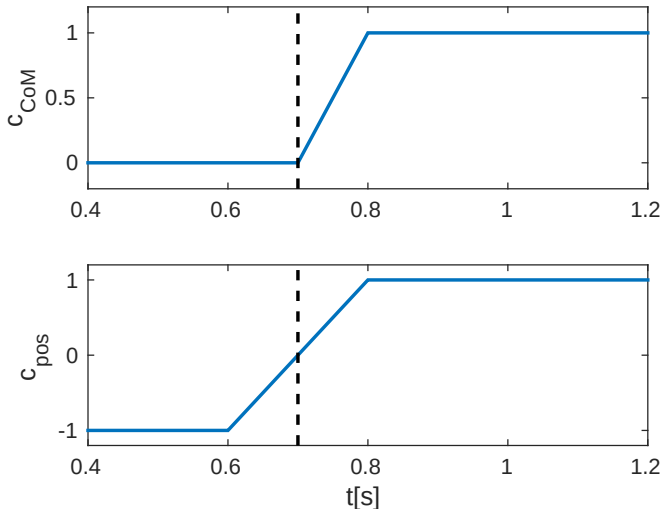
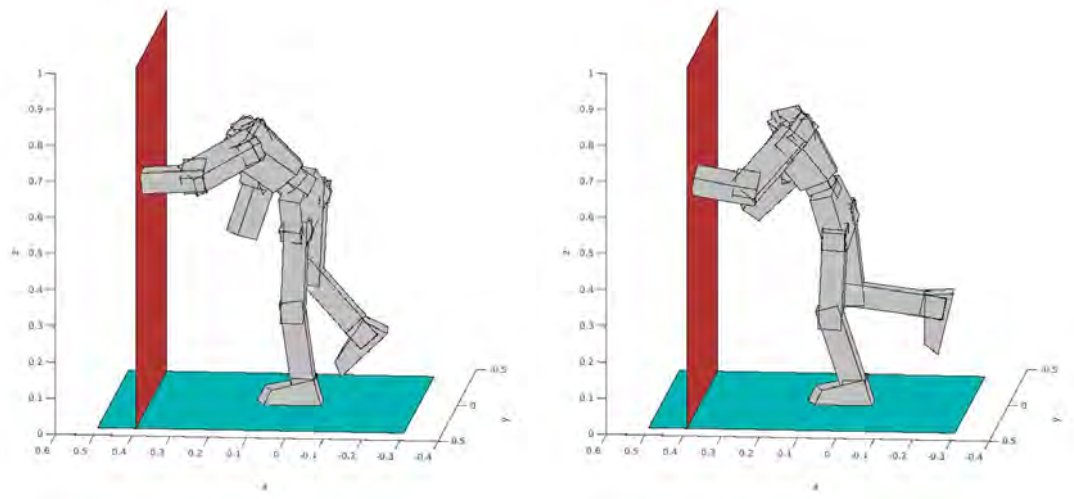


Figure 3.9: The scaling functions $c_{CoM}(t)$ and $c_{pos}(t)$ for the feedback law of the CoM and postural tasks, respectively.

Therefore, in the reverse time simulation, we opt to remove all feedback besides that of the postural task. We use negative values for the postural feedback, which will result in a tracking error diverging from zero in forward time. Subsequently, the error converges to zero in reverse time, which results in a stable reverse time simulation.

To prevent a jump in the input signal at τ_1, caused by the difference between the feedback for forward time and reverse time simulations, we let the feedback constants change gradually over a time frame. For this, we use the scaling functions $c_{CoM}(t)$ and $c_{pos}(t)$ depicted in Figure 3.9. We define the scaled spring and damper constants $k^*_{p,CoM} := c_{CoM}\, k_{p,CoM}$, $k^*_{d,CoM} := c_{CoM}\, k_{d,CoM}$, $k^*_{p,pos} := c_{pos}\, k_{p,pos}$ and , $k^*_{p,pos} := c_{pos}\, k_{p,pos}$, which we will use instead of the regular spring and damper constants in (3.11) and (3.38). We generate the ante-event motion of the second part of the motion task with a reverse time simulation, starting at the event time τ_1 and simulating



(a) t=0.4s, reverse simulation

(b) t=0.7s, start time

(c) t=0.86s, time of release

(d) t=1.16s, post release time

Figure 3.10: Snapshots of the second part of the iCub movement task: the iCub pushes itself away from the wall. Note that snapshots (a) and (d) are taken from the ante- and post-event reference trajectory extensions to be used in the hybrid controller. The extensions are generated under the assumption that the hand constraint is not broken.

for a 0.3s timeframe, while allowing non-feasible contact forces to preserve contact with the wall.

We generate the rest of the second part of the motion task with a forward time simulation starting at the event-time τ_1 with post impact state. At a certain time τ_2 (around $t = 0.86$), the hand will break contact with the wall. We continue the simulation for some time after τ_2, as if the hand is still attached to the wall. This will create a post-event motion, which will be used as post-event reference for the RS hybrid controller. Figure 3.10 displays snapshots of the second part of the motion task.

3.3.3 Moving to initial position

The third part of the motion task consists of the iCub moving back to its initial state. The task trajectory for the hand and CoM displayed in Figure 3.11, reflect this goal.

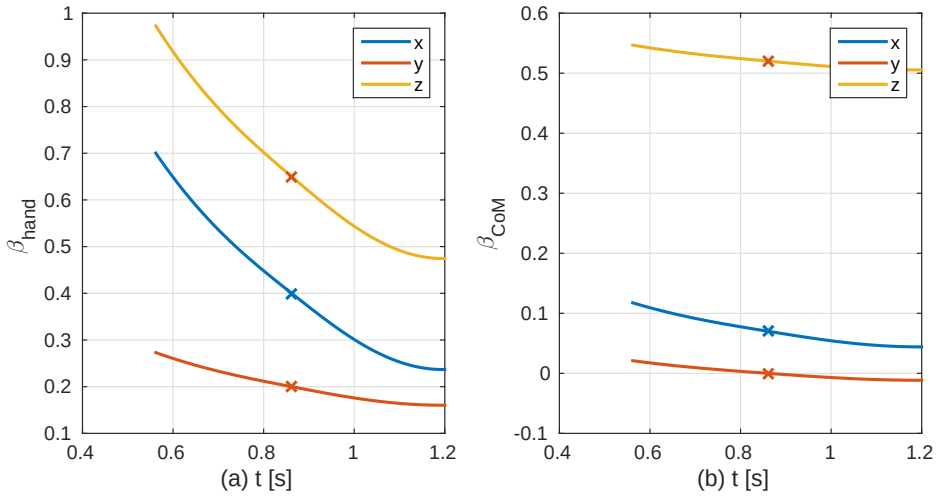


Figure 3.11: The task trajectories for CoM for the iCub moving back to initial condition. The start time is displayed with an x in the graph.

The reference is generated by performing a simulation using the optimal constrained controller as detailed in Section 3.2.1.

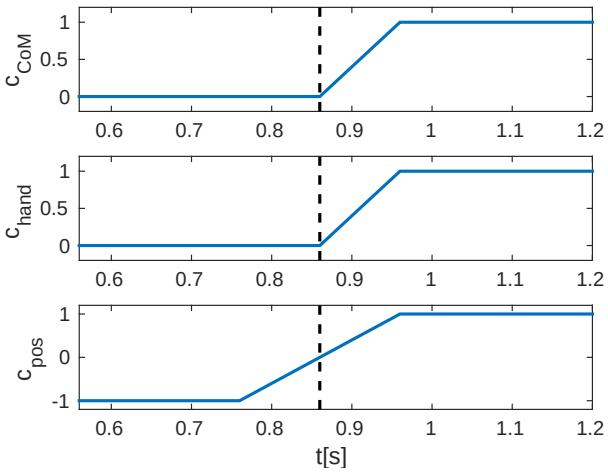
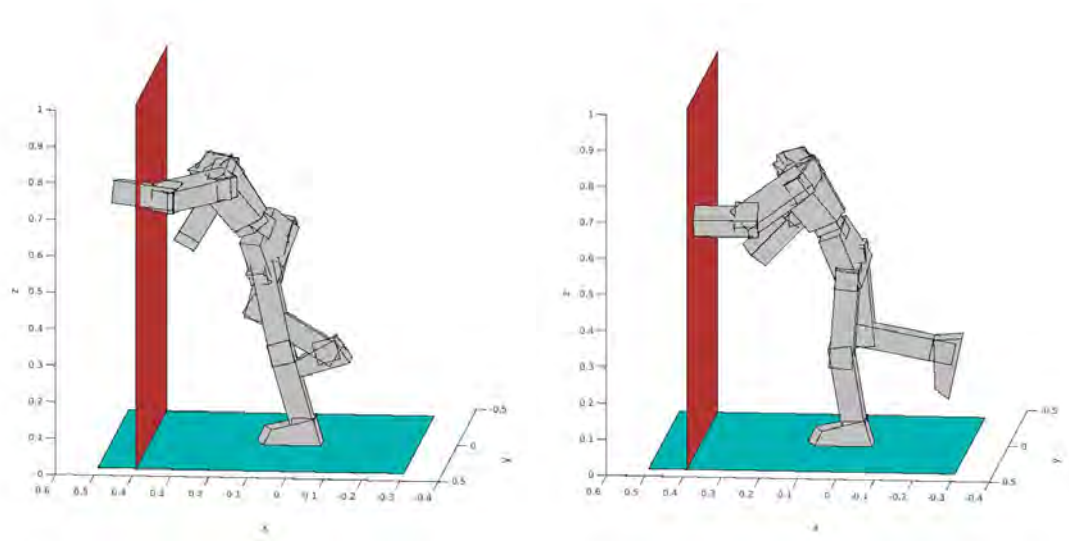


Figure 3.12: The scaling functions $c_{CoM}(t)$, $c_{hand}(t)$ and $c_{pos}(t)$ for the feedback law of the CoM, hand, and postural tasks, respectively.

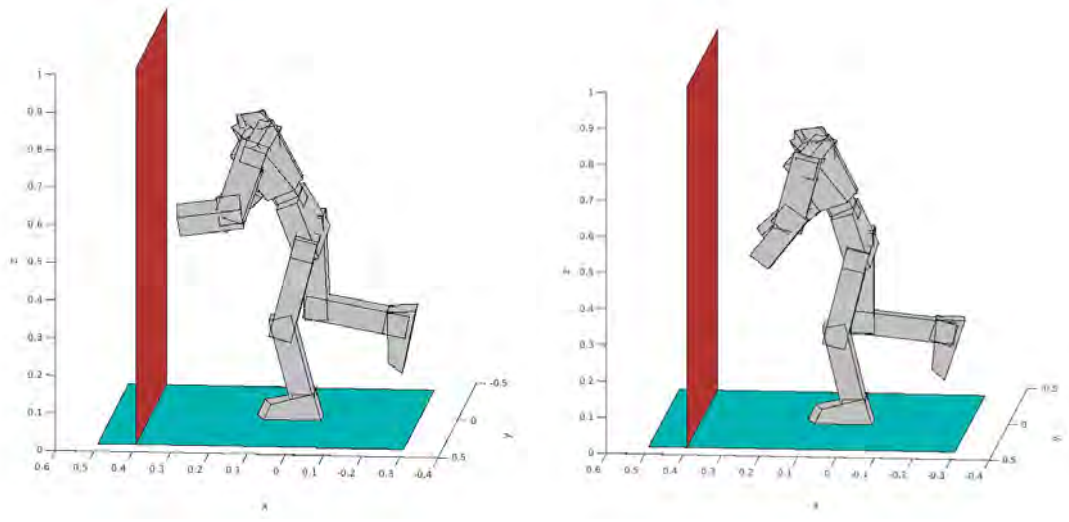
Since this part of the task has no end event time, no post-event simulation will be performed. The simulation starts at τ_2, the event time of the hand breaking contact with the wall. As initial conditions, we use the state of the previous motion subsegment at τ_2. Note that, for the hand breaking contact with the wall, the pre- and post-event state are the same .

Similarly as described in the previous section, to generate the ante-event reference, we perform a reverse time simulation, starting at τ_2. To prevent unstable behavior in the reverse time simulation, we disable the hand an CoM feedback in reverse time, while using negative spring and damper terms for the postural task feedback. Similarly to the previous desired reference motion segment, we use the scaled feedback constants in (3.11) and (3.38). We define $k^*_{p,hand} := c_{hand}\, k_{p,hand}$ and $k^*_{d,hand} := c_{hand}\, k_{d,hand}$. The scaling functions $c_{CoM}(t)$, $c_{hand}(t)$ and $c_{pos}(t)$ are displayed in Figure 3.12. In Figure 3.13, snapshots of the resulting motion are shown.



(a) t=0.56s, reverse simulation

(b) t=0.86s, start time



(c) t=1.06s

(d) t=1.2s

Figure 3.13: Snapshots of the third part of the iCub movement task: the iCub moving back to the initial state. Note that snapshot (a) reperesents the ante-event reference extension to be used in the hybrid controller, and is simulated with the hand constraint disabled.

3.3.4 The complete desired motion with extensions

Combining the simulations of Sections 3.3.1, 3.3.2 and 3.3.3 results in the CoM and hand (extended) task trajectory as shown in Figure 3.14. In this figure, the dashed lines indicate the post- and ante-event segments.

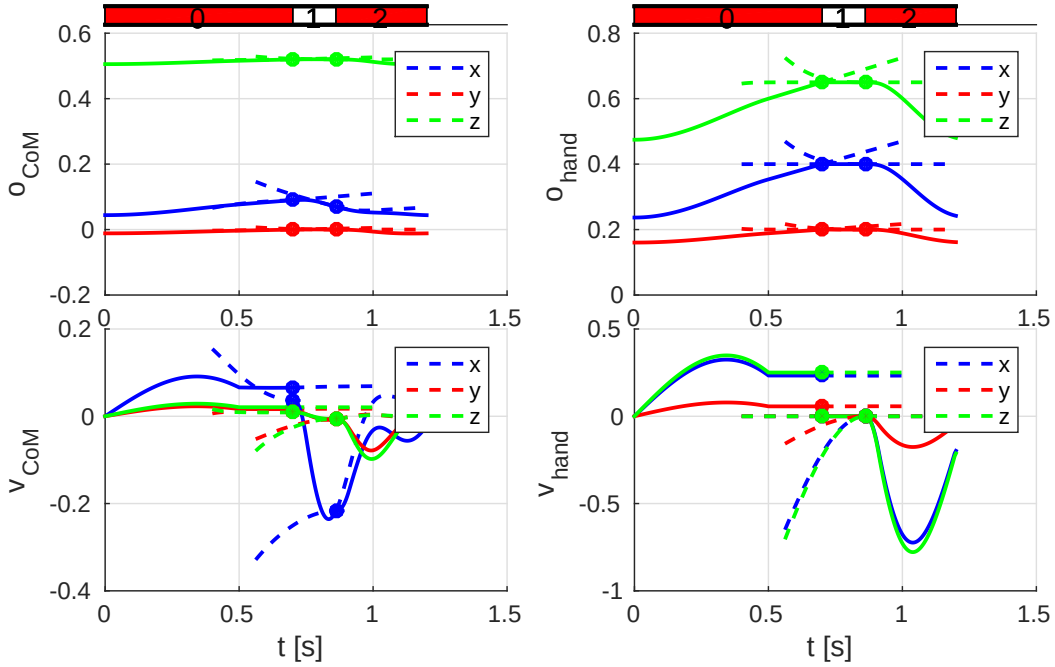


Figure 3.14: The CoM and left hand trajectory of the iCub during the motion task simulation. The dashed lines indicate the post- and ante-event simulations, and the dots indicate an event.

At the impact time ($\tau_1 = 0.7s$), we can see the velocity jump due to the impact. This is most visible in the plot of v_{hand}, where the velocity in all directions jumps to zero. At the second event time (the hand detaching from the wall), no jump in velocity should occur, since no impulsive forces are involved in breaking a contact. As can be seen in the figure, the velocity does indeed not jump at the second event time.

To obtain a periodic motion, we perform a simulations for several motion cycles by repeating the desired motion reference signals. The resulting trajectory can be seen in Figure 3.15. In the velocity plots especially, it can be seen that the first motion cycle is different from the other cycles. This is caused by the fact that the initial velocities are equal to zero, while the later motion cycles start with a nonzero velocity. In the figure, there are no visible diferences between the later cycles of the motion.

Since we use the joint reference trajectories of this motion as input for the hybrid controller, the trajectories of Figure 3.15 will not be used as a direct input for the hybrid controller. Instead, we will use these trajectories to obtain a tracking error for the hand and CoM, which will be used as performance criterea for the RS hybrid controller.

Now that we have obtained a periodic hybrid reference trajectory for the hybrid controller, we can formulate the feedback law of the hybrid controller.

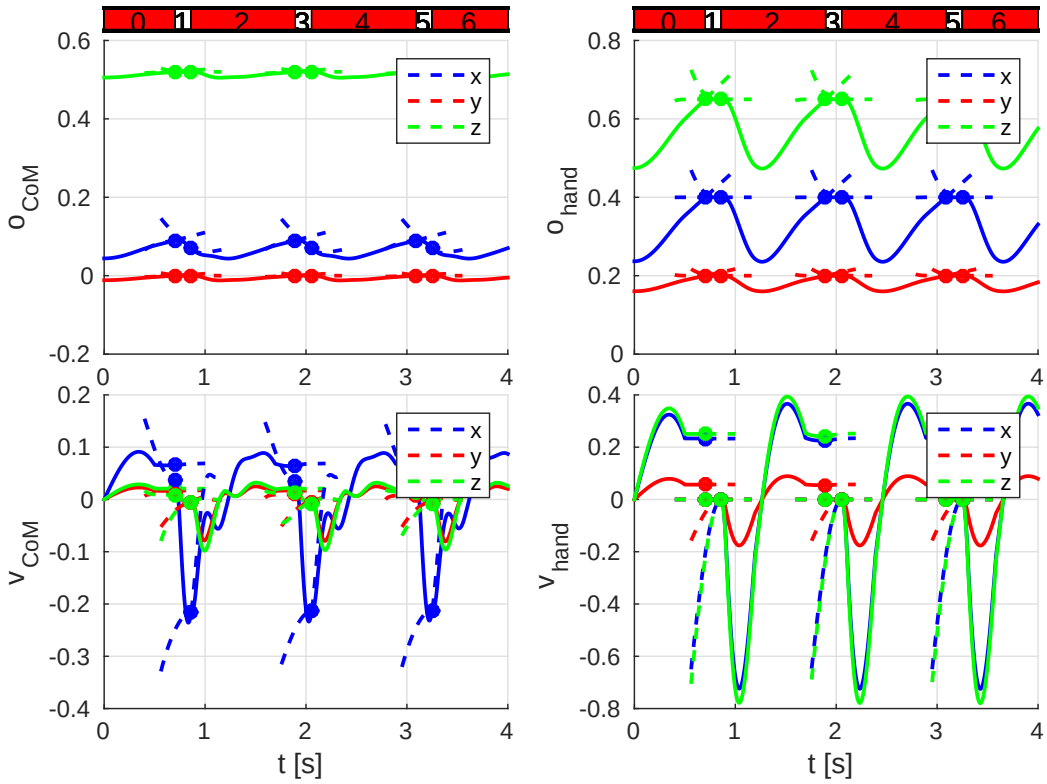


Figure 3.15: The CoM and left hand trajectory of the iCub during the periodically repeated motion task simulation. The dashed lines indicate the post and ante-event simulations, and the dots indicate an event.

3.4 The iCub hybrid control law

From the motion task simulations, we can obtain a hybrid reference trajectory for all the joints of the iCub (described in Chapter 2). Let $\bar{q}_J(t,j)$, $\dot{\bar{q}}_J(t,j)$, and $\bar{\tau}_J(t,j)$ denote the joint angles, joint velocities, and joint feedforward torque of the extended hybrid reference trajectory, respectively. As the RS hybrid controller, we use the simple feedback law

$$\tau_J(t,j) = \bar{\tau}_J(t,j) - K_p\left(q_J(t,j) - \bar{q}_J(t,j)\right) - K_d\left(\dot{q}_J(t,j) - \dot{\bar{q}}_J(t,j)\right), \tag{3.47}$$

with $\tau_J(t,j)$ the control torque and $K_p = K_p(j)$ and $K_d = K_d(j)$ diagonal feedback gain matrices. We use different feedback gain matrices for free motion and contact mode. These will be denoted $K_{p,free}$ and $K_{d,free}$ (used during free motion mode), and $K_{p,contact}$ and $K_{d,contact}$ (used during contact mode), respectively. We aim for a feedback with a critically damped response. This prevents overshoot and oscilations, both of which can be detrimental to the balance of the iCub or can create unwanted contacts. Furthermore, we aim for a reasonably fast convergence.

For the choice of the matrices $K_{p,free}$ and $K_{d,free}$ in free motion mode, we aim for a convergence rate where the joint angle error is reduced by approximately 60% after $0.5s$ for all joints.

Torso		Left arm		Rigth arm		Left leg		Right leg	
Joint nr.	k_p	Joint nr.	k_p	Joint nr.	k_p	Joint nr.	k_p	Joint nr.	k_p
1	30	4	15	9	15	14	25	20	15
2	25	5	13	10	13	15	25	21	13
3	25	6	13	11	13	16	50	22	13
		7	13	12	13	17	50	23	13
		8	13	13	13	18	90	24	13
						19	90	25	13

Table 3.1: The feedback gains $k_{p,free}$ of the RS hybrid controller for all iCub joints in free motion mode. The joint numbers of each limb are assigned with the lowest number closest to the root link.

The resulting feedback gains of $K_{p,free}$ can be seen in Table 3.1. The proportional gains have been determined by individual tuning of each joint to achieve the convergence rate previously mentioned. Each joint is tuned seperately, using a simulation with perturbed initial condition for the joint that is currently tuned. Making use of the schematic overview of the robots joints given in Figure 2.1b, we assigned numbers to the joints of the iCub. Each limb has a set of subsequent joint numbers, with the lowest number closest to the root link (e.g., joint 14 is the left hip joint in y direction, while joint 19 is the left ankle joint in x direction). We take $K_{d,free} = 2\sqrt{K_{p,free}}$ to aim for a critically damped closed loop response.

For the contact mode, we tune the feedback matrix gains in a similar fashion. Again, we aim for a convergence rate where the (joint angle) error is reduced by approximately 60% after $0.5s$ for all joints. To ensure that the left foot contact torques are limited, we opt to choose the left arm gains relatively high, and the left leg gains relatively low. Again, each joint is tuned seperately, using a simulation with perturbed initial condition for the joint that is currently tuned. The resulting feedback gains of $K_{p,contact}$ can be seen in Table 3.2. Similarly to the free motion, we take $K_{d,contact} = 2\sqrt{K_{p,contact}}$ for the damping matrix.

Torso		Left arm		Rigth arm		Left leg		Right leg	
Joint nr.	k_p	Joint nr.	k_p	Joint nr.	k_p	Joint nr.	k_p	Joint nr.	k_p
1	15	4	25	9	15	14	25	20	15
2	15	5	25	10	13	15	25	21	13
3	25	6	25	11	13	16	50	22	13
		7	30	12	13	17	50	23	13
		8	30	13	13	18	90	24	13
						19	90	25	13

Table 3.2: The feedback gains $k_{p,contact}$ of the RS hybrid controller for all iCub joints in contact mode. The joint numbers of each limb are assigned with the lowest number closest to the root link.

Given the state and input extended reference trajectories calculated in Section 3.3 and the gains given in Tables 3.1 and 3.2, we can study the performance of the RS hybrid controller. This is done in the next chapter, which contains simulations for several scenarios, serving both as proof of principle and performance analysis.

Chapter 4

Simulation study: performance of the RS hybrid controller on the iCub

In this chapter, we apply the RS hybrid controller detailed in Chapter 3 to simulate a desired motion, also described in Chapter 3, on the iCub. We consider different scenarios corresponding to perturbation in the initials condition and model parameters. These simulations will serve as a proof of concept, as well as a basic performance and robustness analysis.

The iCub is simulated with the hybrid dynamical model as detailed in Chapter 2. The model has two modes: the free motion mode, where the iCub stands purely on its left foot, and the contact mode, where the iCub leans against a wall with its left hand, in addition to standing on its left foot. As explained in Section 3.4, the core of the RS hybrid controller is the feedback law

$$\tau_J(t,j) = \bar{\tau}_J(t,j) - K_p(j)\left(q_J(t.j) - \bar{q}_J(t,j)\right) - K_d(j)\left(\dot{q}_J(t,j) - \dot{\bar{q}}_J(t,j)\right), \tag{4.1}$$

with (t,j) the hybrid time, $\tau_J(t,j)$ the control torque, $\bar{\tau}_J(t,j)$, $\bar{q}_J(t,j)$ and $\dot{\bar{q}}_J(t,j)$ the feed-forward, joint angles and joint velocities of the hybrid reference trajectory, and K_p and K_d diagonal feedback matrix gains. The hybrid reference trajectory, detailed in Section 3.3, is the same for all simulations. This reference trajectory performs a contact motion, in which the iCub, standing on one foot, reaches for a wall with its hand. After the hand collides with the wall, the robot pushes itself away from the wall, back to its initial configuration. In this Chapter, we use the repeated reference signal described in Section 3.3.4, to allow for a variable simulation time.

The feedback matrices K_p and K_d, which have different values for the free motion mode and contact mode, will be tuned in each simulation. In Section 3.4, we found the feedback values of Table 4.1 through the tuning of each individual joint for a joint angle error reduction of approximately 60% in $0.5s$. We take the feedback matrix gains of Table 4.1 as basis for our simulations.

Free motion mode joint gains															
Torso			Left arm			Right arm			Left leg			Right leg			
Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d
1	30	11	4	15	7.7	9	15	7.7	14	25	10	20	15	7.7
2	25	10	5	13	7.2	10	13	7.2	15	25	10	21	13	7.2
3	25	10	6	13	7.2	11	13	7.2	16	50	14.1	22	13	7.2
			7	13	7.2	12	13	7.2	17	50	14.1	23	13	7.2
			8	13	7.2	13	13	7.2	18	90	19	24	13	7.2
									19	90	19	25	13	7.2

Contact mode joint gains															
Torso			Left arm			Right arm			Left leg			Right leg		
Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d
1	15	7.7	4	25	10	9	15	7.7	14	25	10	20	15	7.7
2	15	7.7	5	25	10	10	13	7.2	15	25	10	21	13	7.2
3	25	10	6	25	10	11	13	7.2	16	50	14.1	22	13	7.2
			7	30	11	12	13	7.2	17	50	14.1	23	13	7.2
			8	30	11	13	13	7.2	18	90	19	24	13	7.2
									19	90	19	25	13	7.2

Table 4.1: The feedback gains of the RS hybrid controller for all iCub joints.

We start our investigation of the controller performance from simulations with randomly perturbed initial conditions with respect to the reference motion. Thereafter, we will selectively perturb the initial conditions to force a nonnegligible mismatch between the nominal and closed loop event-times. Next, we try to simulate some realistic model imperfections, by changing the actual wall position, adding encoder bias, and using affine actuator functions. In each of these scenarios we will briefly analyze the performance of the RS hybrid controller, and tune the feedback gains if necessary.

4.1 Perturbations of initial conditions

We simulate the RS hybrid controller applied on the nominal iCub hybrid dynamical model starting from perturbed initial conditions. In our first simulation, the perturbation of the initial conditions are chosen at random.

4.1.1 Random perturbation of the initial conditions

In this simulation, we apply a random valued perturbation (between -5 and 5 degrees per joint) to the initial conditions of the left and right arms and right leg. The left leg and torso are perturbed by a smaller signal (between -0.5 and 0.5 degrees per joint) to ensure that the initial configuration of the iCub is statically in balance. The RS hybrid controller works successfully if the hybrid state error converges over time and the left foot contact wrench is physically valid throughout the simulation. To give a concise indication of the joint errors, we use the 2-norm of the joint angle errors

$$\|e_{q_J}\| := \|q(t,j) - \bar{\alpha}_q(t,j)\| \tag{4.2}$$

and the 2-norm of the joint velocity errors

$$\|e_{\dot{q}_J}\| := \|\dot{q}(t,j) - \bar{\alpha}_{\dot{q}}(t,j)\|. \tag{4.3}$$

The physical validity of the left foot contact wrench can be derived from the zero tipping moment point (ZMP) p, as discussed in Section 2.3.1.

First, we perform the simulation of the nominal case with randomly perturbed joints and the feedback gains of Table 4.1, resulting in the ZMP positions of Figure 4.1. In this figure, the red lines indicate the convex hull of the foot sole contact area. The color of the plot changes from blue (t=0) to green (t=2). The beginning of each mode is denoted with a circle, while a square denotes the end of a mode. The simulation is terminated just after $t = 1.8s$, when the ZMP leaves the convex hull of the sole contact area, which means that the contact wrench on the left foot is physically invalid. The joint angle and velocity error norms and control torque

norm of this simulation are displayed in Figure 4.2. The blocks in the upper bar (cyan) indicate the event counter and mode of the closed-loop system, while the lower blocks (red) indicate the event counter and mode of the reference trajectory. The joint angle error in this figure displays a large increase from $t = 1.2s$ onwards, which hints at the robot falling over.

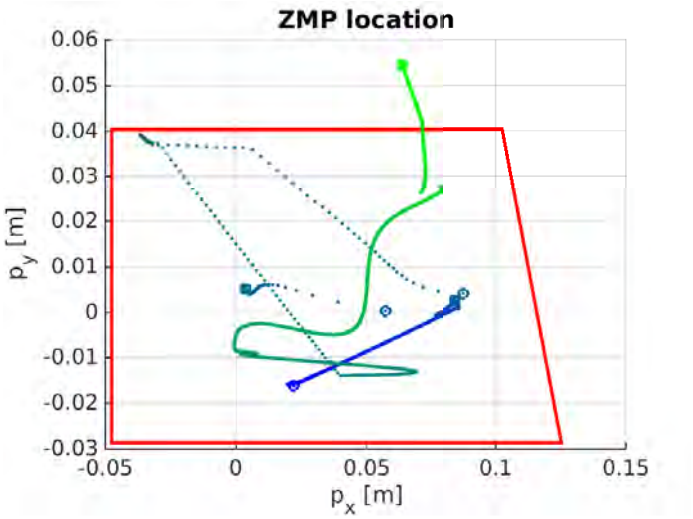


Figure 4.1: The x and y coordinates of the zero tipping moment point (ZMP) p throughout a simulation with randomly perturbed initial conditions. The points are plotted with a time interval of $10^{-3}s$. The red lines indicate the convex hull of the foot sole contact area. As indicated by the ZMP moving outside of this convex hull, the respective contact wrench is physically invalid.

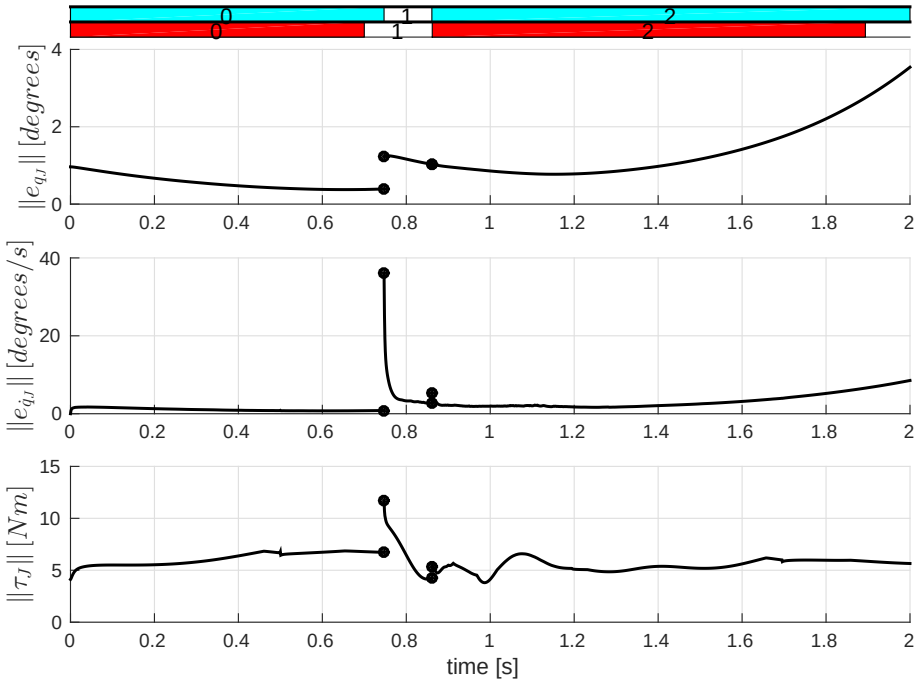


Figure 4.2: The joint angle and velocity error norms and torque norm of a simulation with randomly perturbed initial conditions. A large increase in joint angle error can be seen from $t = 1.2s$ onwards, which hints at the robot falling over.

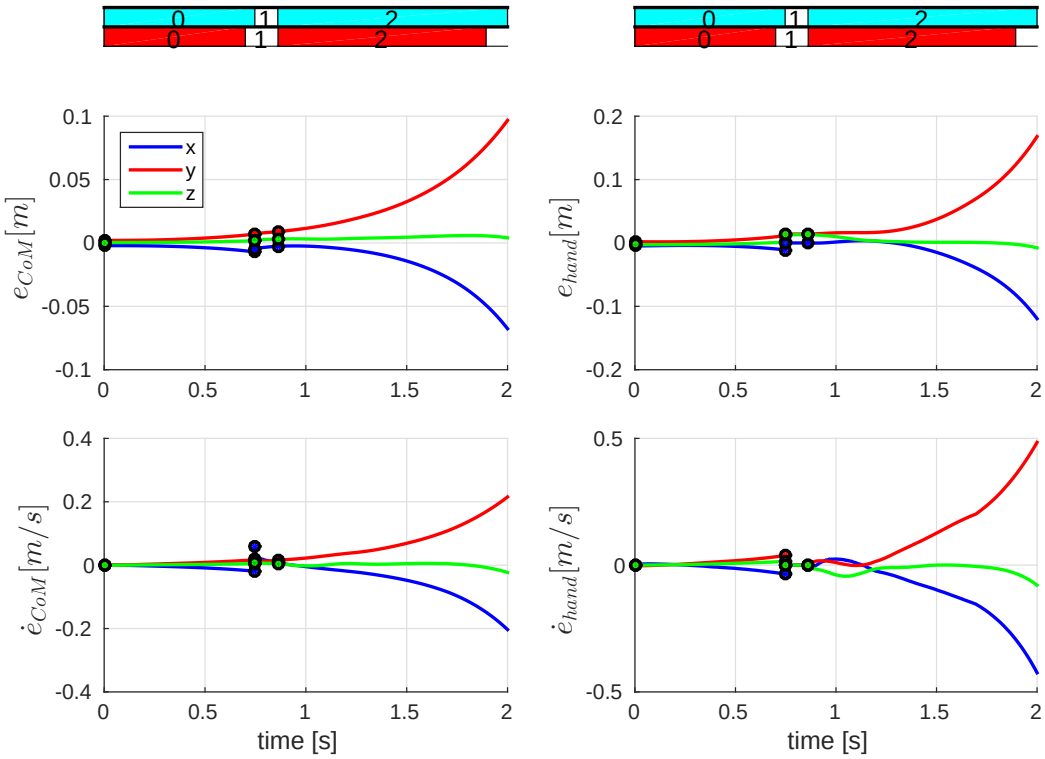


Figure 4.3: The position and velocity errors of the CoM and left hand for a simulation with randomly perturbed initial conditions. As can be seen by the large positional errors in the CoM, the robots falls over.

The notion of the robot falling over is supported by the plots of Figure 4.3, displaying the position and velocity errors of the CoM and left hand. The large error in the CoM x and especially y position indicate that the iCub starts to fall over.

The iCub falls over because the feedback forces are not capable of correcting the joint angle errors. This is reflected in Figure 4.2, where the torque norm barely increases while the robot falls over, indicating that the feedback gains are too low.

Free motion mode joint gains														
Torso			Left arm			Right arm			Left leg			Right leg		
Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d
1	90	19	4	15	7.7	9	15	7.7	14	75	17.3	20	15	7.7
2	75	17.3	5	13	7.2	10	13	7.2	15	75	17.3	21	13	7.2
3	75	17.3	6	13	7.2	11	13	7.2	16	150	24.5	22	13	7.2
			7	13	7.2	12	13	7.2	17	150	24.5	23	13	7.2
			8	13	7.2	13	13	7.2	18	270	32.9	24	13	7.2
									19	270	32.9	25	13	7.2

Contact mode joint gains															
Torso			Left arm			Right arm			Left leg			Right leg			
Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d
1	60	15.5	4	25	10	9	15	7.7	14	100	20	20	15	7.7
2	60	15.5	5	25	10	10	13	7.2	15	100	20	21	13	7.2
3	100	20	6	25	10	11	13	7.2	16	200	28.2	22	13	7.2
			7	30	11	12	13	7.2	17	200	28.2	23	13	7.2
			8	30	11	13	13	7.2	18	360	37.9	24	13	7.2
									19	360	37.9	25	13	7.2

Table 4.2: The feedback gains of the RS hybrid controller for all iCub joints.

To prevent the iCub from falling over, we increase the feedback values corresponding with the left leg and torso. These two body parts are the most vital for the robots balance. We perform the simulation again, this time with the feedback values of Table 4.2. The resulting ZMP positions can be seen in Figure 4.4. From this figure, we can conclude that the left foot contact wrench is physically valid throughout the simulation.

Figure 4.5 displays the joint angle and velocity error norms and the torque norm. Both error norms display a convergence to zero over time. The mismatch between the expected and actual impact time also decreases with the second motion cycle.

Figure 4.5 has a few remarkable thing to be noted. In the velocity error norm, large spikes can be seen at the impact time of each motion cycle. These spikes occur because the velocity jump of the actual impact does not match the velocity jump due to switching between reference trajectory modes. The spikes converge to zero rather fast. This indicates that the damping feedback gains might be too high. A smaller velocity error spike can be seen when the hand detaches from the wall in the first motion cycle. This spike is caused entirely by switching between the reference trajectory modes, since no state jump occurs at this event. In the second motion cycle, this spike is not present.

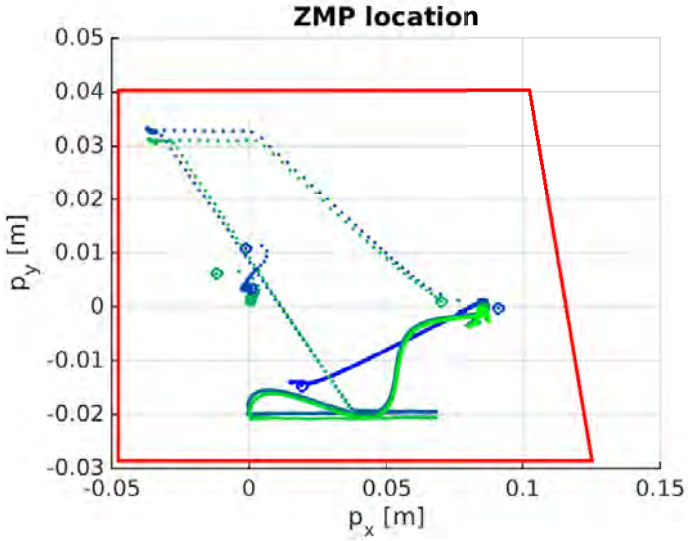


Figure 4.4: The x and y coordinates of the zero tipping moment point (ZMP) p throughout a simulation with randomly perturbed initial conditions and feedback gains of Table 4.2. The ZMP remains within the convex hull throughout the simulation, indicating a valid left foot contact wrench.

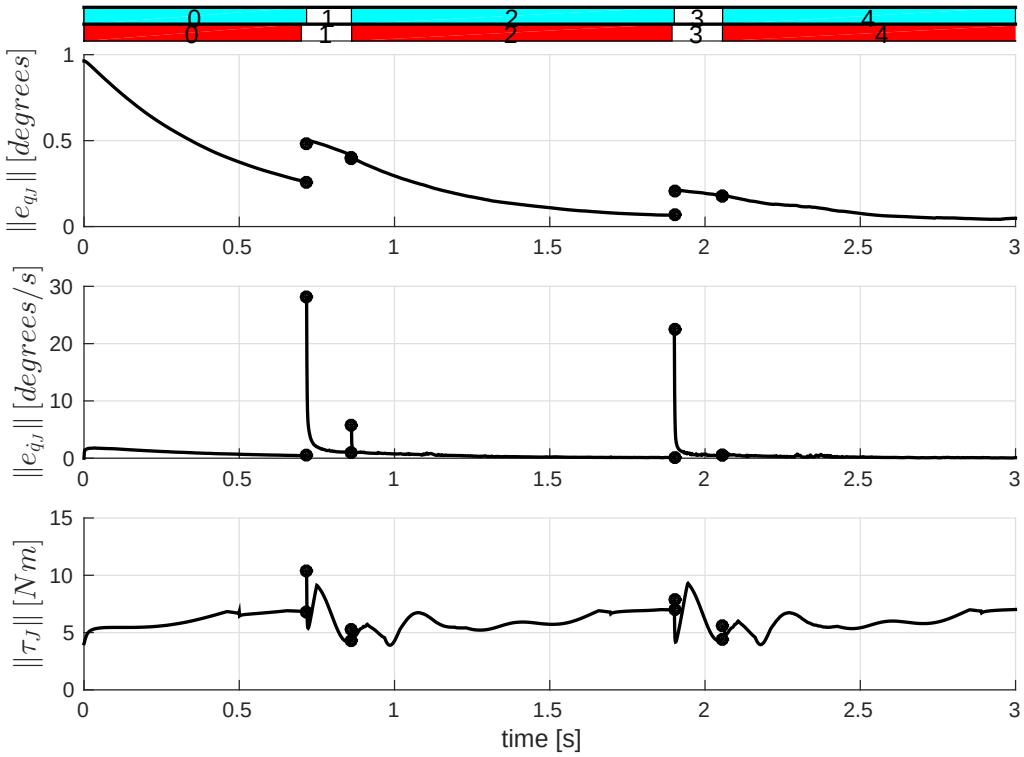


Figure 4.5: The joint angle and velocity error norms and the torque norm of a simulation with randomly perturbed initial conditions and feedback gains of Table 4.2. Both error norms seem to converge to zero over time.

In the torque norm plot, we can see oscillating behavior after the impact time in both motion cycles. These oscillations indicate that the closed loop system is not (over)critically damped, and indicates that certain damping feedback gains might be too low.

In Figure 4.6, the position and velocity errors of the CoM and left hand are displayed. The errors of both trajectories seem to converge to zero, indicating that the hybrid controller has been applied successfully. As is the case in Figure 4.5, velocity error spikes can be seen for the CoM at the impact times in both motion cycles. The left hand has no spikes in velocity error, since it is constrained during the contact mode of the system.

This simulation serves as a proof of concept for the RS hybrid control applied to a humanoid robot model. Therefore, the main goal of this report, as discussed in the introduction, has been achieved. We will perform additional simulations to assess the controllers performance and robustness to model parameter mismatches and model imperfections. Next, we will perform a nominal simulation with a large mismatch between closed loop and nominal impact time.

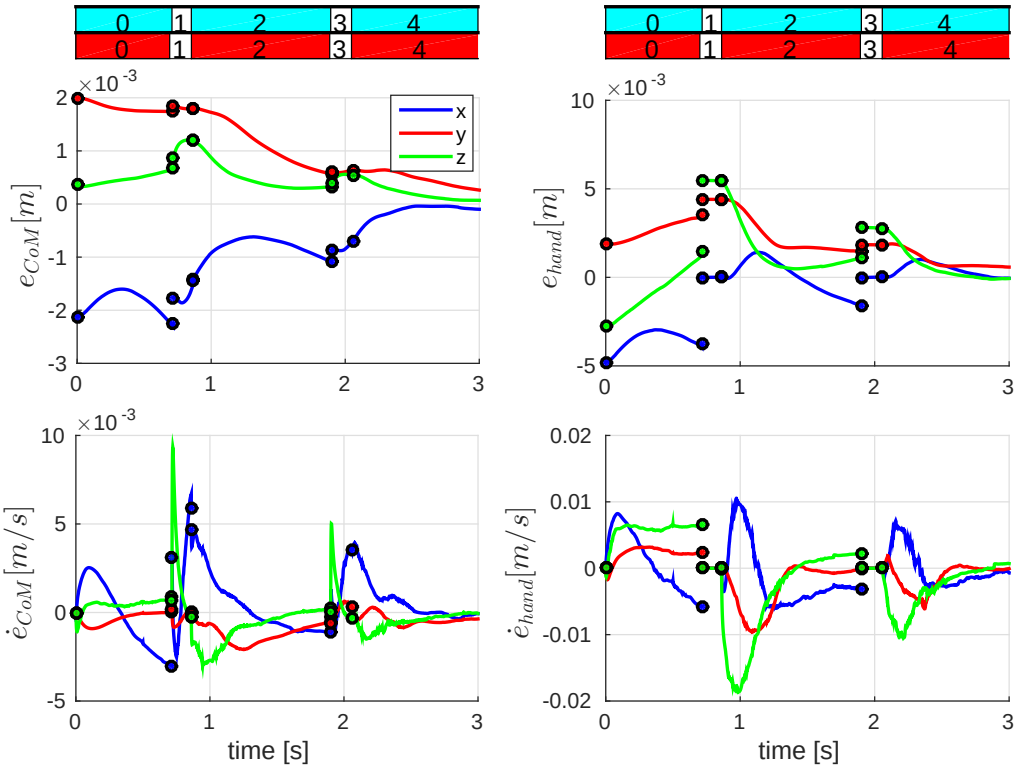


Figure 4.6: The position and velocity errors of the CoM and left hand for a simulation with randomly perturbed initial conditions and feedback gains of Table 4.2. Both trajectories see their errors converging to zero over time.

4.1.2 Selectively perturbed initial conditions: large impact time mismatch

In this simulation, we once again apply a randomized disturbance to the initial condition (± 5 degrees for arms and right leg, $\pm, 0.5$ degrees for left leg and torso). In addition, we add another perturbation to the left arm, designed to make the arm more retracted in its initial position and therefore closer to the robots torso. This will induce a delay of the closed-loop impact time with respect to the nominal one.

If we run this simulation with the feedback constants of Table 4.2, we obtain the ZMP as displayed in Figure 4.7. In this figure, it is clear that the ZMP moves outside of the left soles convex hull, right after the hand collides with the wall. This is a direct result of the time mismatch between the closed loop and nominal impact time. Due to the nature of the reference trajectory extensions, a large mismatch between the closed-loop and nominal event time will result in a large post impact tracking error, which in turn causes a high feedback torque. This feedback torque causes a physically invalid contact wrench on the left sole (i.e., would make the robot tip over if applied on a real robot).

To find the source of the high feedback torque, we look at the different components of the control torque. Figure 4.8 displays the contribution of the position-dependent feedback gain, velocity-dependent feedback gain, and feedforward term to the total control torques for the left leg. The velocity dependent feedback spikes at the event time, and is up to one order of magnitude bigger then the position-dependent feedback. Therefore, it is likely that the velocity feedback is the cause of the invalid contact wrench.

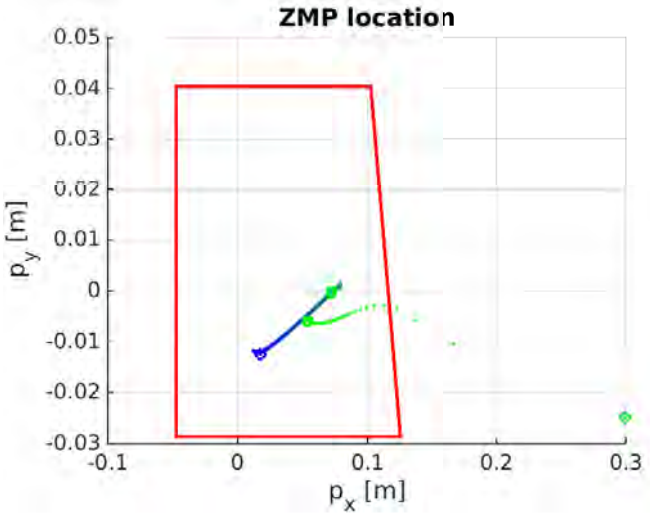


Figure 4.7: The x and y coordinates of the zero tipping moment point (ZMP) p throughout a simulation with the arm more retracted in the initial position. The ZMP leaves the convex hull, indicating an invalid contact wrench.

We retune the feedback gains by increasing the left leg position-dependent feedback gains and lowering the velocity-dependent feedback gains. This equalizes the order of magnitude of the different feedback gains, while lowering the post impact control torque. The retuned feedback matrix gains are given in Table 4.3.

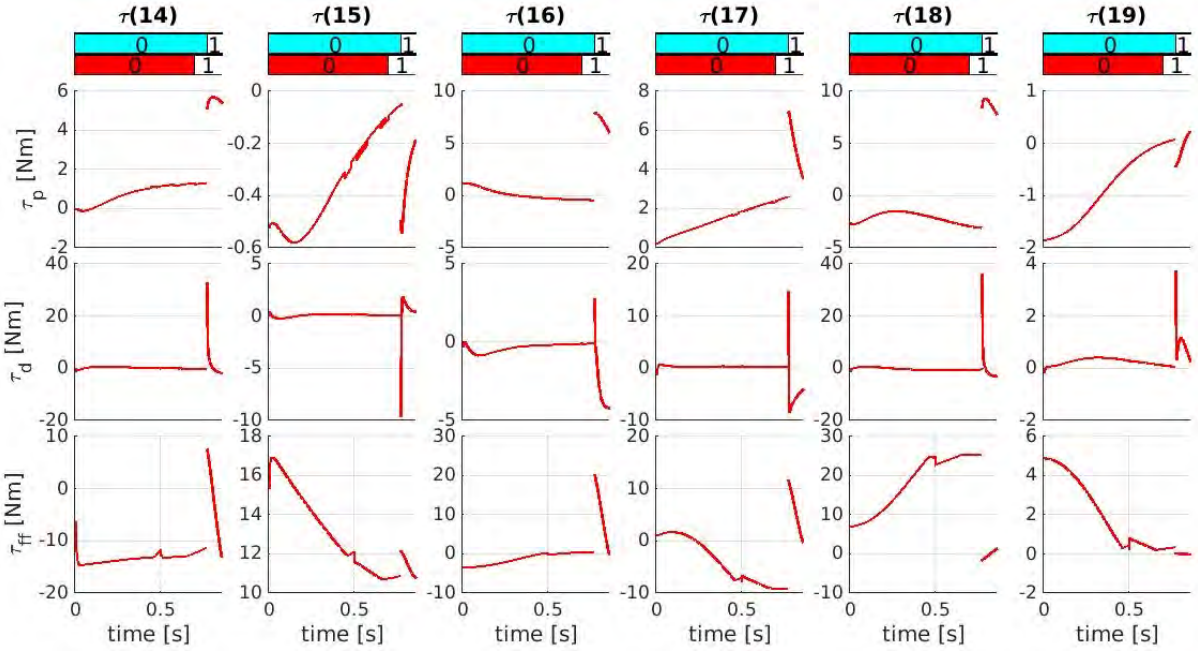


Figure 4.8: The position-dependent feedback, velocity-dependent feedback, and feedforward torques for the left leg joints.

Free motion mode joint gains														
Torso			Left arm			Right arm			Left leg			Right leg		
Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d
1	60	15.5	4	15	7.7	9	15	7.7	14	150	5.4	20	15	7.7
2	50	14.1	5	13	7.2	10	13	7.2	15	150	5.4	21	13	7.2
3	50	14.1	6	13	7.2	11	13	7.2	16	300	23.1	22	13	7.2
			7	13	7.2	12	13	7.2	17	300	23.1	23	13	7.2
			8	13	7.2	13	13	7.2	18	540	62	24	13	7.2
									19	540	15.5	25	13	7.2
Contact mode joint gains														
Torso			Left arm			Right arm			Left leg			Right leg		
Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d
1	30	11	4	25	10	9	15	7.7	14	150	5.4	20	15	7.7
2	30	11	5	25	10	10	13	7.2	15	150	5.4	21	13	7.2
3	50	14.1	6	25	10	11	13	7.2	16	300	23.1	22	13	7.2
			7	30	11	12	13	7.2	17	300	23.1	23	13	7.2
			8	30	11	13	13	7.2	18	540	62	24	13	7.2
									19	540	15.5	25	13	7.2

Table 4.3: The feedback gains of the RS hybrid controller for all iCub joints.

We repeat the previous simulation with the feedback values of Table 4.3. The resulting ZMP plot is displayed in Figure 4.9. The ZMP remains within the convex hull throughout the simulation, indicating a physically valid contact wrench.

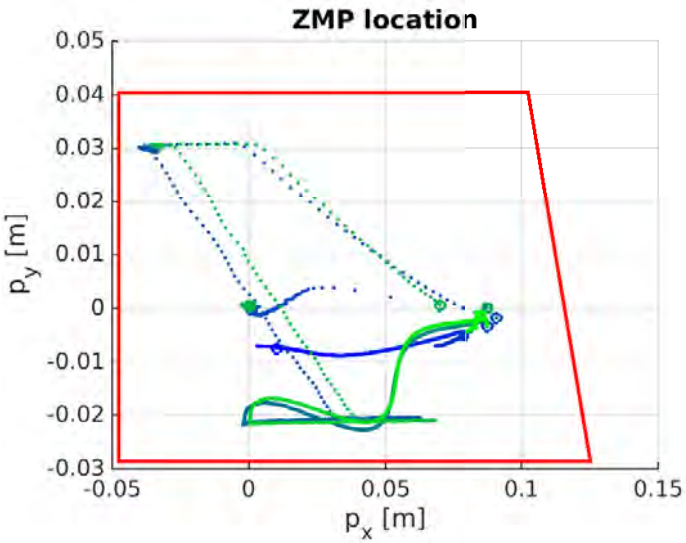


Figure 4.9: The x and y coordinates of the zero tipping moment point (ZMP) p throughout a simulation with the arm more retracted in the initial position. This simulation is performed with the retuned feedback matrix gains of Table 4.3. The ZMP remains inside the convex hull throughout the simulation.

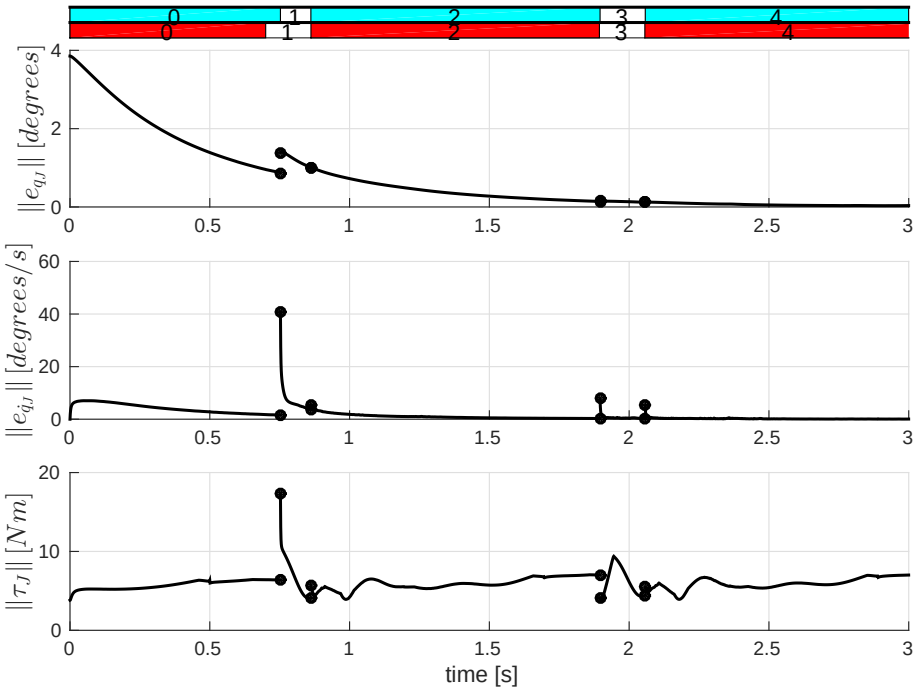


Figure 4.10: The joint angle and velocity error norms and the torque norm of a simulation with the arm more retracted in the initial position. This simulation is performed with the retuned feedback matrix gains of Table 4.3. Both error norms converge to zero over time.

Figure 4.10 displays the joint angle and velocity error norms and the torque norm during the simulation. Both error norms seem to converge to zero over time. Again, we see a large spike in the velocity error norm at the impact time of the first motion cycle, caused by the difference between the closed loop and nominal impact times. The rate of convergence of this spike seems to be lower than that of Figure 4.5, most likely a result of the lowered damping.

There is a significantly smaller spike in the second motion cycle (compared to that of Figure 4.5), which suggests a faster overall error convergence. In the joint angle error norm plot seems to confirm this notion, with a significantly faster error convergence than that of Figure 4.5. In the torque norm plot, we still see an oscillation after each event. The small errors, and subsequently small feedback torques, in the second motion cycle suggest that this oscillation might be induced by the feedforward signal.

Figure 4.11 displays the position and velocity errors of the CoM and left hand for the current simulation. Both trajectories see their errors converging to almost zero in two motion cycles, thereby supporting the notion that the current feedback matrix gains result in a higher rate of convergence. The CoM velocity has a very large spike in the x direction at the first impact time. This is most likely caused by the large mismatch in x coordinates between the closed loop system and the reference trajectory at during impact, resulting in a large mismatch between the closed loop and nominal velocity jump.

In all plots of Figure 4.11, the x coordinate has a significantly larger error than the other coordinates. This is caused by the selective disturbance of the initial conditions, which mainly influence the x coordinate of the hand (and subsequently the CoM).

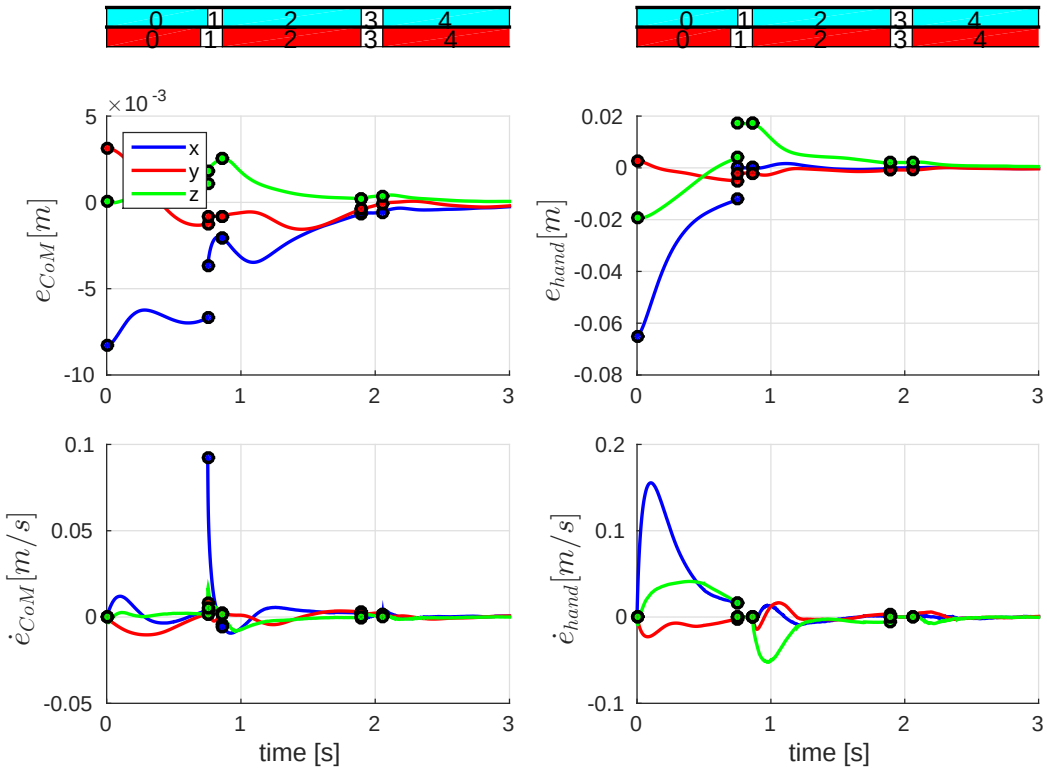


Figure 4.11: The position and velocity errors of the CoM and left hand for a simulation with the arm more retracted in the initial position. This simulation is performed with the retuned feedback matrix gains of Table 4.3. Both trajectories see their errors converging to almost zero in two motion cycles.

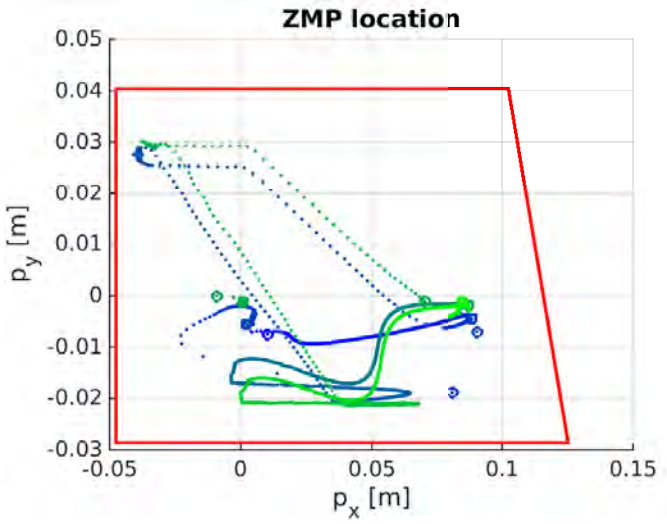


Figure 4.12: The x and y coordinates of the zero tipping moment point (ZMP) p throughout a simulation with the arm extended in the initial position. This simulation is performed with the retuned feedback matrix gains of Table 4.3. The ZMP remains inside the convex hull throughout the simulation.

Alternatively, we can also perform a simulation with an extended initial position of the arm. We do this by applying the same randomized disturbance to the initial condition (± 5 degrees for arms and right leg, ± 0.5 degrees for left leg and torso) and applying an additional disturbance to the left arm that is designed to make it more stretched. We performed such a simulation with the feedback values of Table 4.3, the same feedback gains we used the previous simulation.

The resulting ZMP is displayed in Figure 4.12. The ZMP stays within the convex hull throughout the simulation. One can clearly see a repeating pattern in the plot, which is caused by the periodic nature of the reference trajectory.

Figure 4.13 displays the joint angle and velocity error norms and the torque norm of the simulation. At the impact time of the first motion cycle, we again see a large velocity error norm spike. In the second motion cycle, the velocity error norm has a significantly smaller spike, similarly to Figure 4.10. Unlike in our previous simulations, the joint angle error norm plot does not display a jump at any point in this simulation.

The torque norm plot of Figure 4.13 displays larger oscillations than our previous simulations. The most likely cause for this is the nature of the feedforward signal. In the creation of the ante-event reference extension, we use a controller that is stable in a reverse time simulation (as detailed in Section 3.3). This might result in a less suitable reference extension, compared to the post-event reference extensions, which are generated using a normal controller.

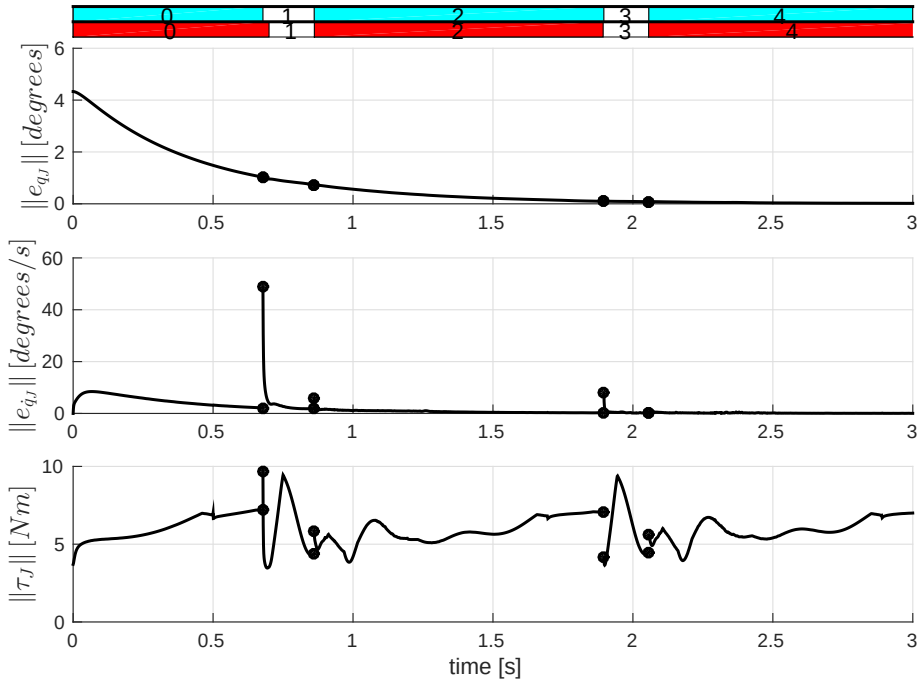


Figure 4.13: The joint angle and velocity error norms and the torque norm of a simulation with the arm extended in the initial position. This simulation is performed with the retuned feedback matrix gains of Table 4.3. Both error norms converge to zero over time.

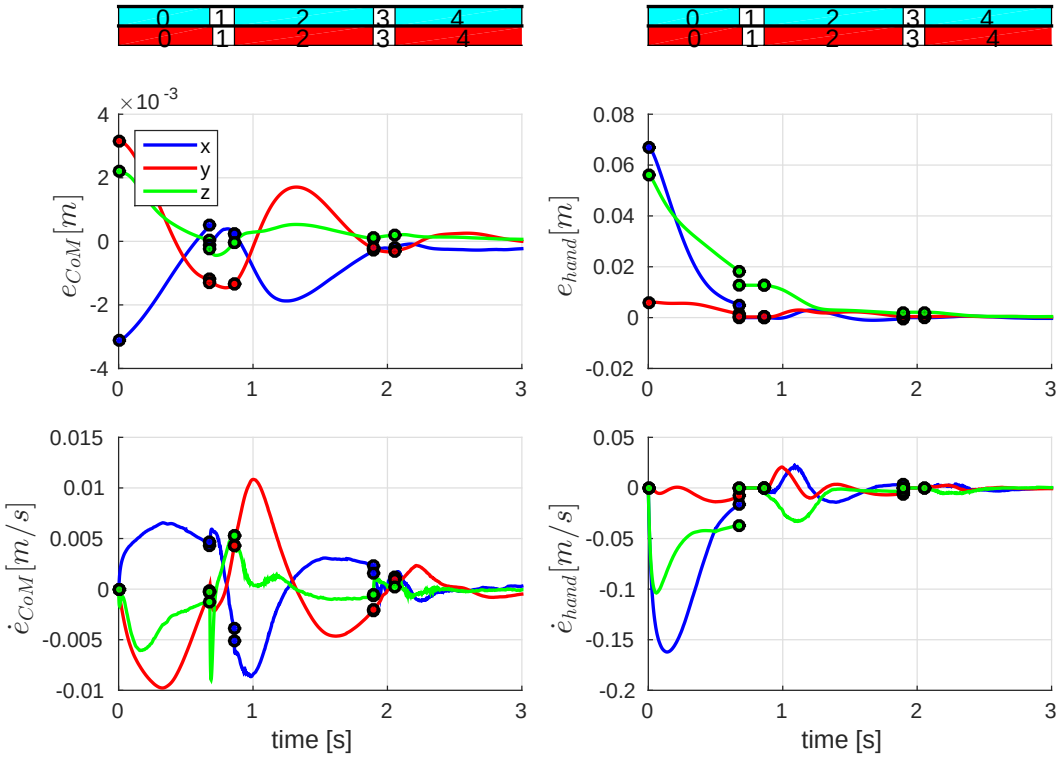


Figure 4.14: The position and velocity errors of the CoM and left hand for a simulation with the arm extended in the initial position. This simulation is performed with the retuned feedback matrix gains of Table 4.3. Both trajectories see their errors converging to almost zero in two motion cycles.

Figure 4.14 displays the position and velocity errors of the CoM and left hand for the current simulation. Both trajectories see their errors converging to almost zero over time in a similar fashion as our previous simulations.

With the feedback values of Table 4.3, we have witnessed a large region of attraction, making the RS hybrid controller to exhibit favourable robustness properties to perturbed initial conditions. The controllers applicability on a real robot is, however, limited by the balance of the iCub. We require high position-dependent feedback gains on the left leg to ensure that the iCub does not fall over. The velocity feedback has to be small enough that a jump in the joint velocity error does not trigger a feedback force that breaks the left foot constraint. To further test the robustness of the system, we study the effect of several model imperfections to the hybrid dynamical model of the iCub. These model imperfections will be discussed next.

4.2 Model imperfections

In this section, we perform simulations with several model imperfections to obtain an indication of the robustness of the RS hybrid controller. These simulations will all have perturbed initial conditions (± 5 degrees for arms and right leg, ± 0.5 degrees for left leg and torso) and are aimed at modeling realistic model errors for humanoid robots. In these simulations, we will use the RS hybrid controller with the feedback constants of Table 4.3. As we will see, these feedback gains will require further adjustments.

We start this section by modeling a randomized encoder bias. Secondly, we will add imperfect actuator functions to the model. The last simulation of this section will have a perturbation on the wall position (which is essentially the same as positioning the iCub at a different distance from the wall).

4.2.1 Encoder bias

Encoder bias is one of the most basic common model imperfections on a robotic system. During the assembly of robots, encoders will always be oriented slightly different than intended. This slight mismatch in orientation results in a constant encoder error called encoder bias. We can model this encoder bias by adding randomized constants to the input state of the controller. We use an encoder bias with an arbitrary value between 0.5 and -0.5 degrees for each joint, while we use the feedback constants of Table 4.3. The resulting ZMP position is displayed in Figure 4.15

The ZMP remains inside the convex hull throughout the simulation, indicating a physically valid simulation. Figure 4.16 displays the joint angle and velocity error norms and the torque norm of the simulation. The joint angle error norm does not seem to converge to a zero. Due to the encoder bias, the feedback is effectively following an alternate (biased) reference trajectory with a constant, slight difference from the intended reference trajectory.

Note that the mismatch between actual and expected event times does not converge to zero, since the biased reference trajectory does not have the same event times as the intended reference trajectory. This causes a spike in both the joint angle and velocity error norms in each motion cycles. We can again see an oscillation in the torque norm after the moment of impact in each motion cycle, most likely caused by the nature of the feedforward signal.

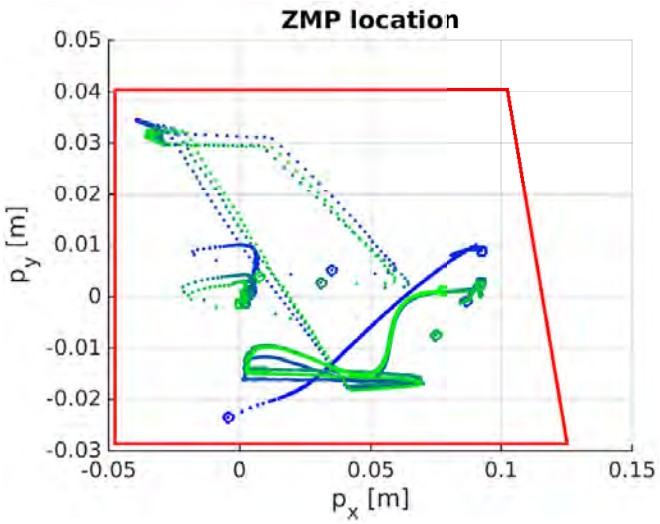


Figure 4.15: The x and y coordinates of the zero tipping moment point (ZMP) p throughout the simulation with arbitrary encoder bias. The ZMP remains inside the convex hull throughout the simulation, indicating a physically valid simulation.

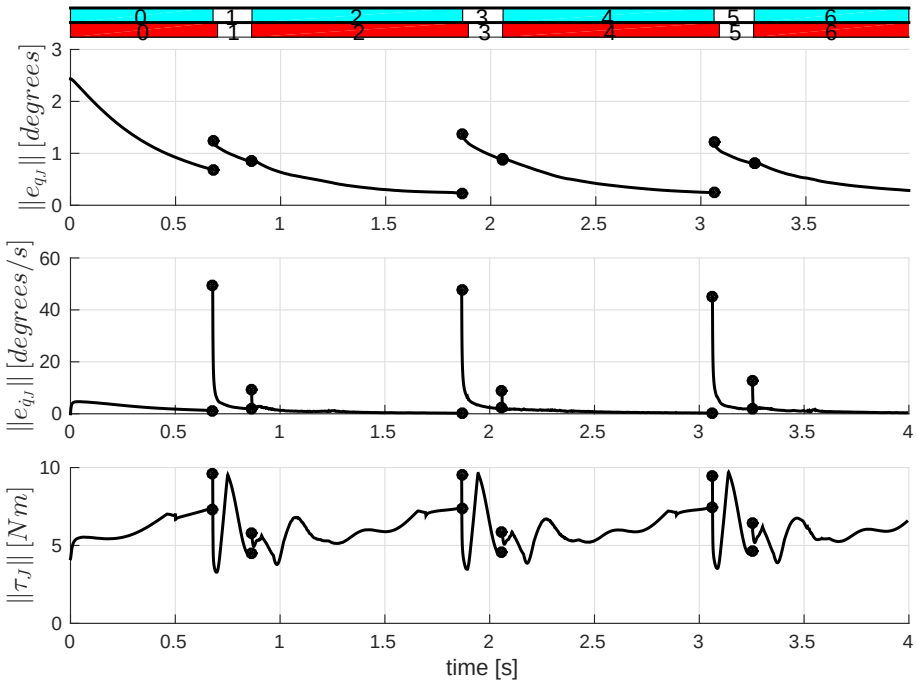


Figure 4.16: The joint angle and velocity error norms and the torque norm of a simulation with arbitrary encoder bias. The error norms do not converge to zero and spike up at the moment of impact in each motion cycle.

Figure 4.17 displays the position and velocity errors of the CoM and left hand. The errors do not seem to converge to zero over time. Instead, they create a pattern that seems to repeat itself for each motion cycle. This phenomenon is a result of the fact that the controller is effectively trying to track a biased reference. With the perturbation in initial conditions vanishing over time, this leaves us with an error that is similar for each motion cycle.

All in all, the controller seems robust to the addition of a small encoder bias. Since the encoder bias can not be seen by the controller, it cannot be compensated for in the control strategy. For the iCub, the performance of the RS hybrid controller relies heavily on the actual values of the encoder error (as opposed to their order of magnitude). If all encoder errors turn out in such a way that the iCub leans sideways, the performance of the controller decreases drastically. When we perform several simulations with different iterations of randomized encoder biases in the same range, the performance of the controller varies from near perfect tracking to an invalid simulation where the left foot contact gets broken.

In the next section, we perform simulations on a model with affine actuator functions.

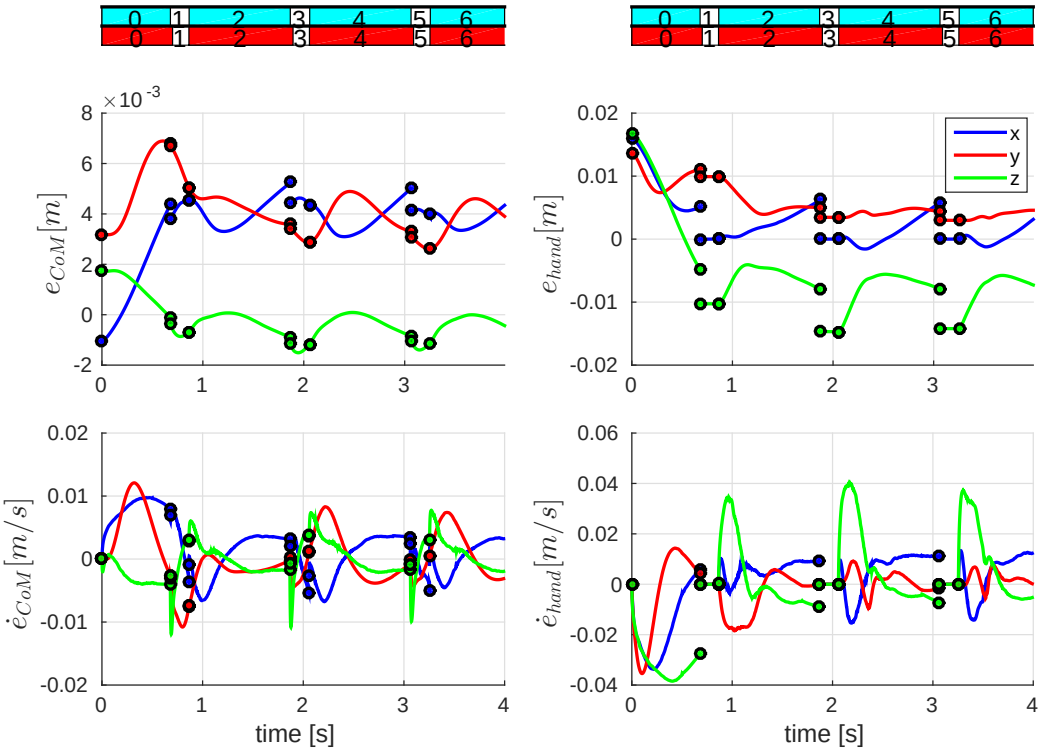


Figure 4.17: The position and velocity errors of the CoM and left hand for a simulation with arbitrary encoder bias. The errors do not seem to converge to zero, but instead display a periodic pattern.

4.2.2 Imperfect actuators

In dynamical models, actuators are often modeled as ideal actuators, described by

$$\tau_J = \tau_{J,des}, \tag{4.4}$$

with $\tau_{J,des}$ the desired torques and τ_J the actual torques delivered by the actuators. However, real actuators never supply exactly the desired torque. We can model the difference between the desired and actual actuator torques with an affine actuator function

$$\tau_J = \xi + (I_{n_J} + \Psi)\tau_{J,des}, \tag{4.5}$$

with $\xi \in \mathbb{R}^{n_J}$ a randomized static bias vector and $\Psi \in \mathbb{R}^{n_J \times n_J}$ a randomized diagonal proportional bias matrix. In the previous simulations of this report, the norm of joint torques is equal to 6.24. With this value in mind, we use a randomized vector with values between $-0.06Nm$ and $0.06Nm$ as ξ. For now, we take $\Psi = 0$.

We use these values in a simulation with perturbed initial conditions (± 5 degrees for arms and right leg, ± 0.5 degrees for left leg and torso), the feedback constants of Table 4.3, and no encoder bias. The resulting ZMP position can be seen in Figure 4.18. The ZMP remains inside the convex hull throughout the simulation, indicating a physically valid simulation.

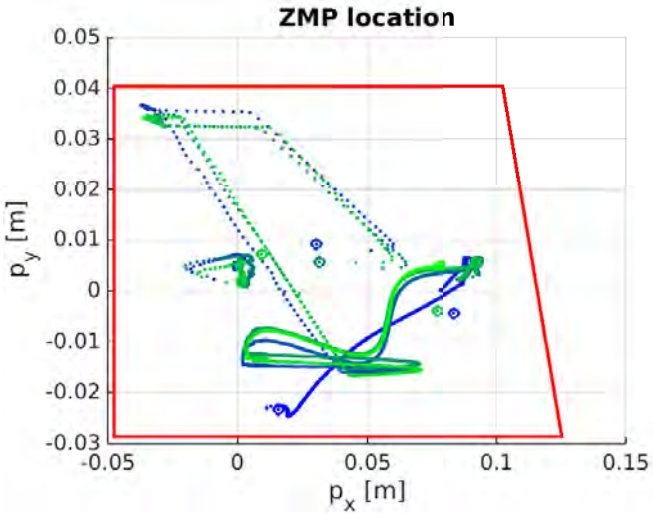


Figure 4.18: The x and y coordinates of the zero tipping moment point (ZMP) p throughout the simulation with arbitrary actuator bias. The ZMP remains inside the convex hull throughout the simulation, indicating a physically valid simulation.

Figure 4.19 displays the joint angle and velocity error norms and the torque norm of the simulation. The results are remarkably similar to those of Figure 4.16, even though we used a different iteration of randomized perturbation of the initial conditions. The actuator function bias will be compensated by the feedback of the controller. Since the controller does not have an integrator, the actuator bias will eventually be counteracted by a feedback torque induced by a more or less static error in the joint angles. Therefore, it is not very surprising that the simulation results resemble those of the actuator bias simulation. Adding an integrator to the control law might result in better error convergence, but lies outside of the scope of this report.

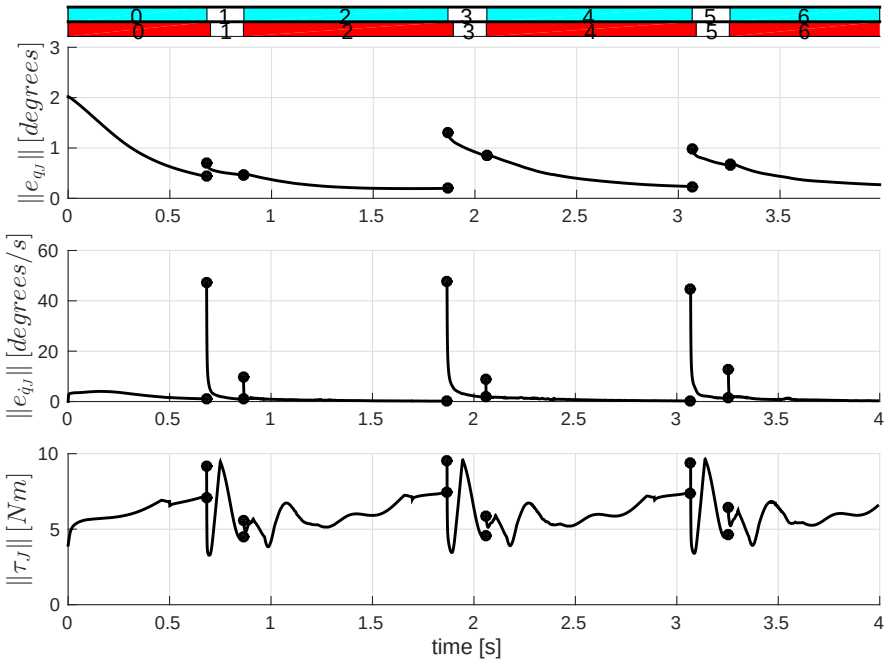


Figure 4.19: The joint angle and velocity error norms and the torque norm of a simulation with arbitrary actuator bias. The results are very similar to those of the actuator bias simulation.

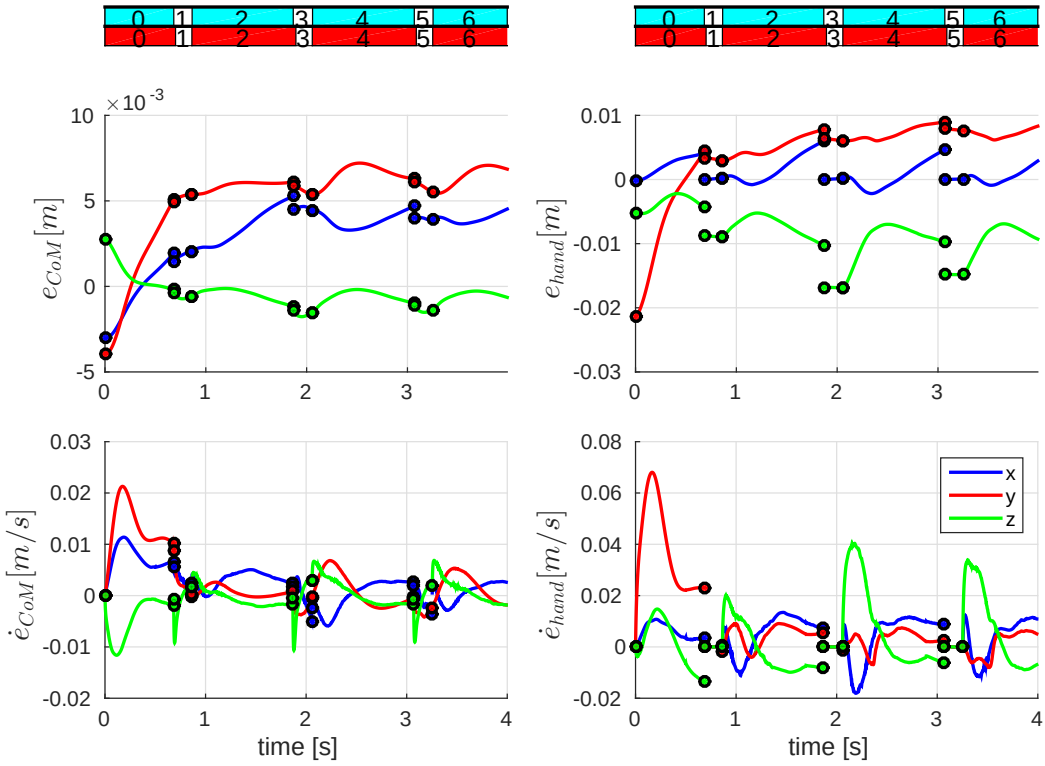


Figure 4.20: The position and velocity errors of the CoM and left hand for a simulation with arbitrary actuator bias. The results are very similar to those of the actuator bias simulation, displaying a periodic pattern.

Figure 4.20 displays the position and velocity errors of the CoM and left hand. Again. the results are very similar to those of the actuator bias simulation. The plots display a periodic pattern that is similar to that of Figure 4.17.

The results of actuator bias simulations is heavily dependent on the current iteration of the randomized actuator functions. For example, when an actuator in the left leg has a high bias, it can have a devastating effect on the controllers performance. An actuator in the right ankle, on the other hand, has little influence on the performance of the controller. Similarly to the encoder bias simulations, performing simulations with different iterations of the randomized actuator functions gives results varying from near perfect tracking to invalid simulations where the left foot contact gets broken.

As an alternative scenario, we can change the slope of the actuator function by take $\xi = 0$ and for Ψ a randomized diagonal matrix with values between -0.01 and 0.01. We can use these values in a simulation with a new iteration of perturbed initial conditions (± 5 degrees for arms and right leg, ± 0.5 degrees for left leg and torso), the feedback constants of Table 4.3, and no encoder bias. We are not able to tune the feedback gains such that a we can obtain a physically valid simulation.

The largest problem of these, so-called, affine actuator functions is that the applied feedforward differs from the desired feedforward. This means that we have to rely on the feedback to converge to the reference trajectory. However, the feedback torques do not take the balance of the iCub into account, and increasing them does, in this case, not contribute to a stable system.

Due to the unstable nature the iCub system, even a slight change in the applied feedforward destabilizes the system.

When we try to lower the range of Ψ, we start to obtain physically valid simulations for values between -0.002 and 0.002. This indicates rather unfovourable robustness properties against actuators with an affine actuator function. Using a state dependent feedforward could be a viable method for increasing the robustness of the controller, but this lies out of the scope of this research.

The last model imperfection we consider in this report is changing the position of the wall, which we will discuss next.

4.2.3 Changing the wall position

Changing the position of the wall is essentially the same as changing the foot position of the iCub. Since humanoid robots like the iCub judge the distance towards a wall using either laser sensors or cameras, an error of a few centimeters is a quite realistic scenario.

In our first simulation of this section, we consider a wall at $x_{wall} = 0.4 + \delta x_{wall}$ with $\delta x_{wall} = 0.01$, and use perturbed initial conditions (± 5 degrees for arms and right leg, ± 0.5 degrees for left leg and torso), the feedback constants of Table 4.3, and no encoder bias. The resulting ZMP position is given in Figure 4.21.

The ZMP leaves the convex hull at $t = 0.86s$, just after the hand breaks contact with the wall. The iCub cannot properly track the reference trajectory during contact mode, since the contact point of the left hand is different from the contact point in the reference. Therefore, the iCub enters the free motion mode with relatively large errors in joint configuration and velocity. These errors trigger a feedback torque that breaks the left foot contact. Figure 4.22 displays the contribution of the position-dependent feedback gain, velocity-dependent feedback gain, and feedforward term to the total control torques for the left leg.

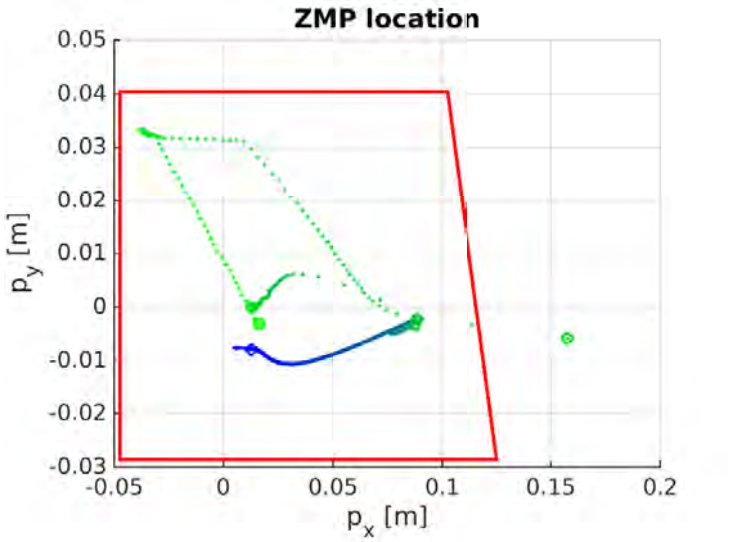


Figure 4.21: The x and y coordinates of the zero tipping moment point (ZMP) p throughout a simulation with wall position $x_{wall} = 0.41m$. The ZMP leaves the convex hull at $t = 0.86s$, right after the hand breaks contact with the wall.

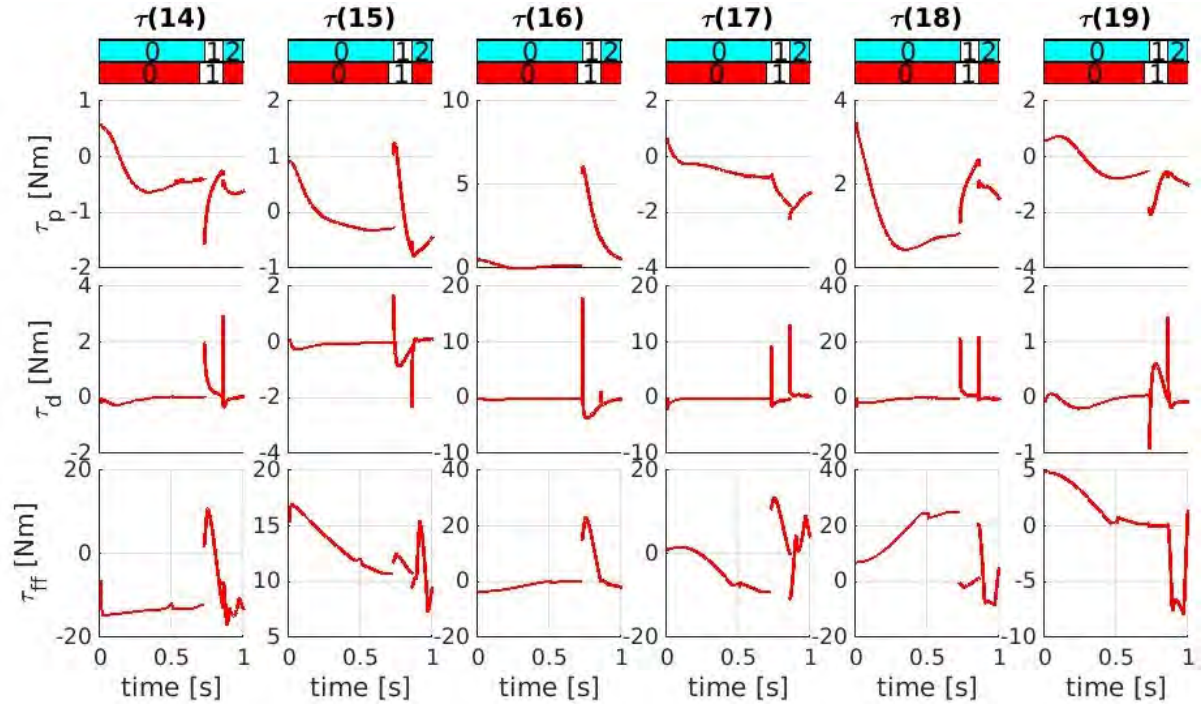


Figure 4.22: The position-dependent feedback, velocity-dependent feedback, and feedforward torques for the left leg joints during a simulation with wall position $x_{wall} = 0.41m$.

The velocity-dependent feedback torques displays large spikes at both event times. It is likely that these spikes are the cause for the control torques that break the left foot contact. To make the controller more robust, we can decrease the joint velocity feedback of the left leg in free motion mode, to decrease the spike in the control torque.

Additionally, we can increase the feedback gains for the left leg in contact mode, while simultaneously decreasing the feedback for the torso and left arm. With this, we aim to distribute the joint angle errors in contact mode, which are inevitable due to the difference between the closed-loop and nominal contact point of the hand. By decreasing the feedback gains of the left arm and torso, these limbs will have larger joint errors, allowing the left leg to maintain lower joint errors levels, and subsequently, lower feedback torques. The adjusted feedback gains are given in Table 4.4.

Free motion mode joint gains														
Torso			Left arm			Right arm			Left leg			Right leg		
Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d
1	60	15.5	4	15	7.7	9	15	7.7	14	100	3	20	15	7.7
2	50	14.1	5	13	7.2	10	13	7.2	15	100	3	21	13	7.2
3	50	14.1	6	13	7.2	11	13	7.2	16	200	1.8	22	13	7.2
			7	13	7.2	12	13	7.2	17	200	1.8	23	13	7.2
			8	13	7.2	13	13	7.2	18	360	2.4	24	13	7.2
									19	360	2.4	25	13	7.2

Contact mode joint gains															
Torso			Left arm			Right arm			Left leg			Right leg		
Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d	Joint	k_p	k_d
1	7.5	5.5	4	2.5	3.2	9	15	7.7	14	150	24.5	20	15	7.7
2	7.5	5.5	5	2.5	3.2	10	13	7.2	15	150	24.5	21	13	7.2
3	12.5	7.1	6	2.5	3.2	11	13	7.2	16	300	69.3	22	13	7.2
			7	3	3.5	12	13	7.2	17	300	69.3	23	13	7.2
			8	3	3.5	13	13	7.2	18	540	139	24	13	7.2
									19	540	139	25	13	7.2

Table 4.4: The feedback gains of the RS hybrid controller for all iCub joints.

We repeat the previous simulation with the feedback gains of Table 4.4 to obtain the ZMP position displayed in Figure 4.23.

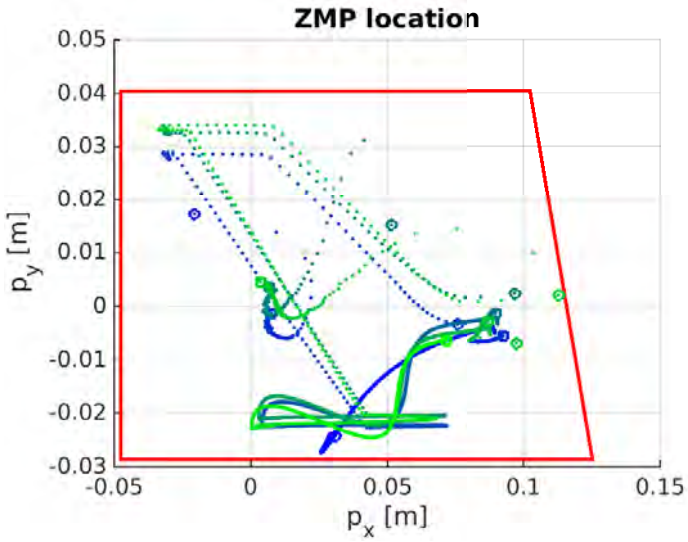


Figure 4.23: The x and y coordinates of the zero tipping moment point (ZMP) p for a simulation with wall position $x_{wall} = 0.41m$ and feedback gains of Table 4.4. The ZMP remains within the convex hull throughout the simulation.

The ZMP remains inside the convex hull of the sole contact area. Figure 4.24 displays the joint angle and velocity error norms and the torque norm for the simulation. The error norms appears to slowly diverge from zero, with an increasing mismatch between the closed loop and nominal event times with each motion cycle. The event time mismatch and jump in error norms during impact is caused by the fact that the reference is designed for a specific wall position. If the wall is placed at another distance, then the joint configuration of the contact mode reference cannot be achieved, since the contact point is not in the expected location.

The desired contact task is still performed in this simulation, as the robot does not fall over and applies the contacts as desired. However, the divergent nature of the error norm plots seem to indicate that the system is unstable.

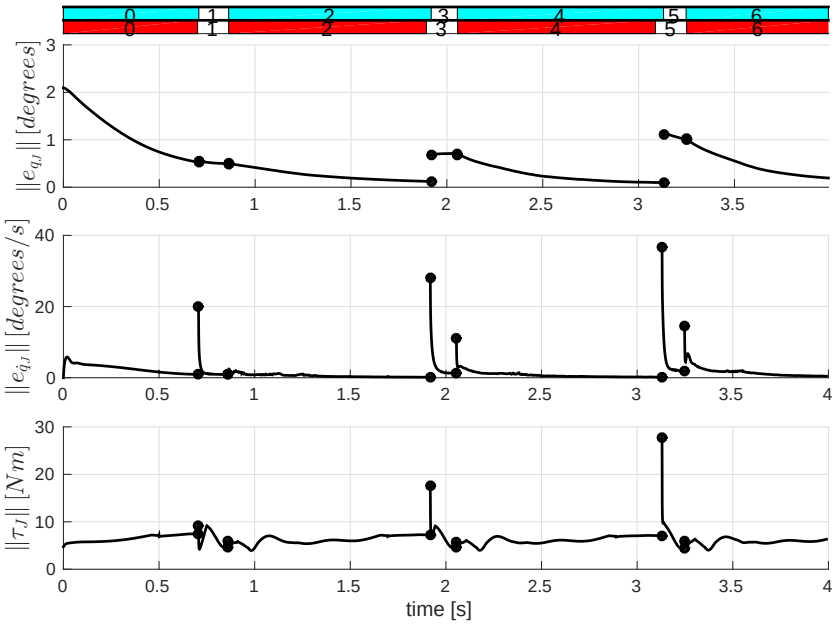


Figure 4.24: The joint angle and velocity error norms and the torque norm for a simulation with wall position $x_{wall} = 0.41m$ and feedback gains of Table 4.4. The error norms appears to slowly diverge from zero, with an increasing mismatch between closed loop and nominal event times with each motion cycle.

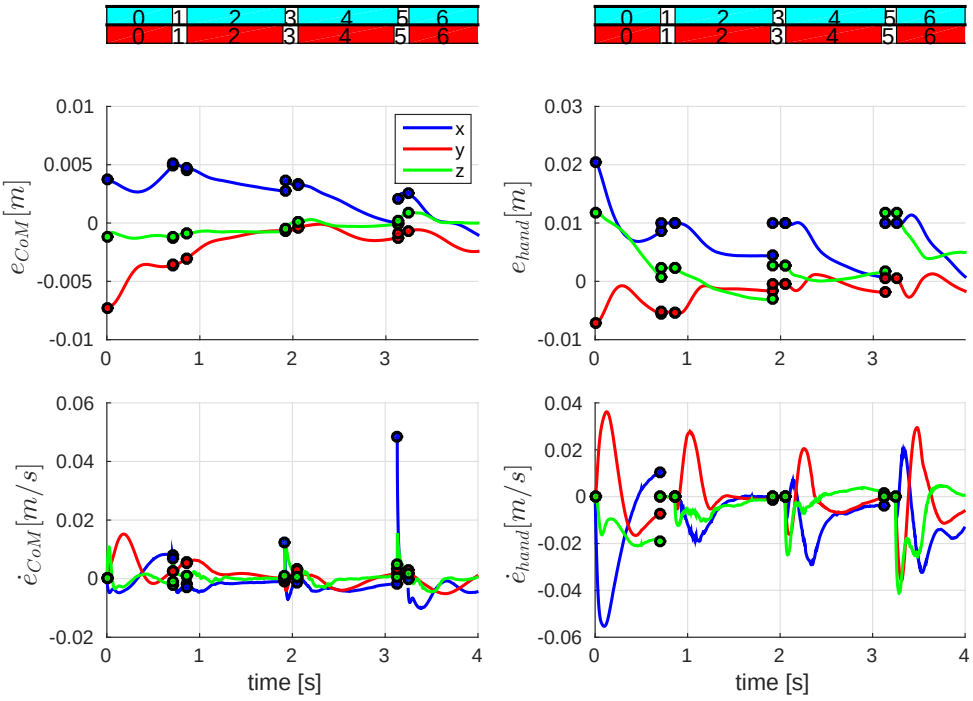


Figure 4.25: The position and velocity errors of the CoM and left hand for a simulation with wall position $x_{wall} = 0.41m$ and feedback gains of Table 4.4. The desired contact task is performed, as indicated by the relatively small position errors.

Figure 4.25 displays the position and velocity errors of the CoM and left hand. The plots support the notion that the desired contact task is performed, as indicated by the relatively small position errors. However, the error signals do not seem to converge over time. In the velocity plot of the CoM, one can see a large error spike around $t = 3.2s$. This spike is caused by the increasing mismatch between the closed loop and nominal event times.

As an alternative scenario, we can place the wall closer to the iCub. In our next simulation, we consider a wall at $x_{wall} = 0.4 + \delta x_{wall}$ and $\delta x_{wall} = -0.01$. If we use the feedback gains of Table 4.4, the resulting simulation is physically invalid.

If we go back to using the feedback gains of Table 4.3, the simulation is physically valid. The resulting ZMP position is displayed in Figure 4.26.

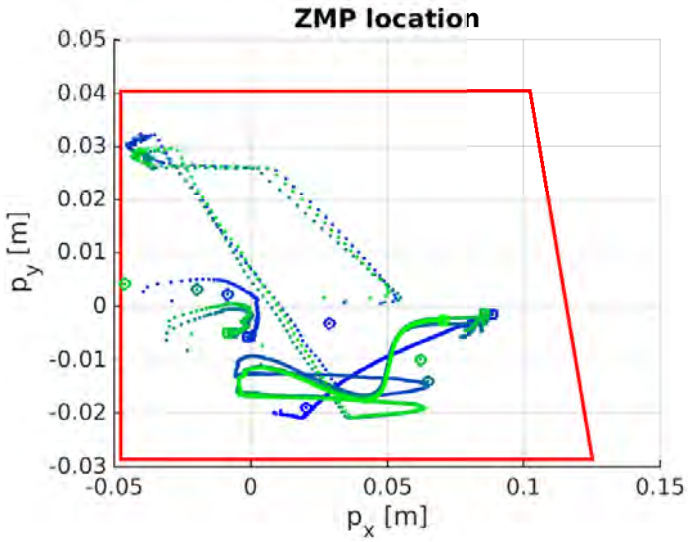


Figure 4.26: The x and y coordinates of the zero tipping moment point (ZMP) p for a simulation with wall position $x_{wall} = 0.39m$ and feedback gains of Table 4.3. The ZMP remains within the convex hull throughout the simulation.

The ZMP remains inside the convex hull of the sole contact area (although with small margins). Figure 4.24 displays the joint angle and velocity error norms and the torque norm for the simulation. The error norms appears to slowly converge to zero, with an decreasing mismatch between the closed loop and nominal event times with each motion cycle. The performance seems to be significantly better than in our simulation with $x_{wall} = 0.41m$.

Figure 4.25 displays the position and velocity errors of the CoM and left hand. The error signals seems to converge over time, albeit slowly.

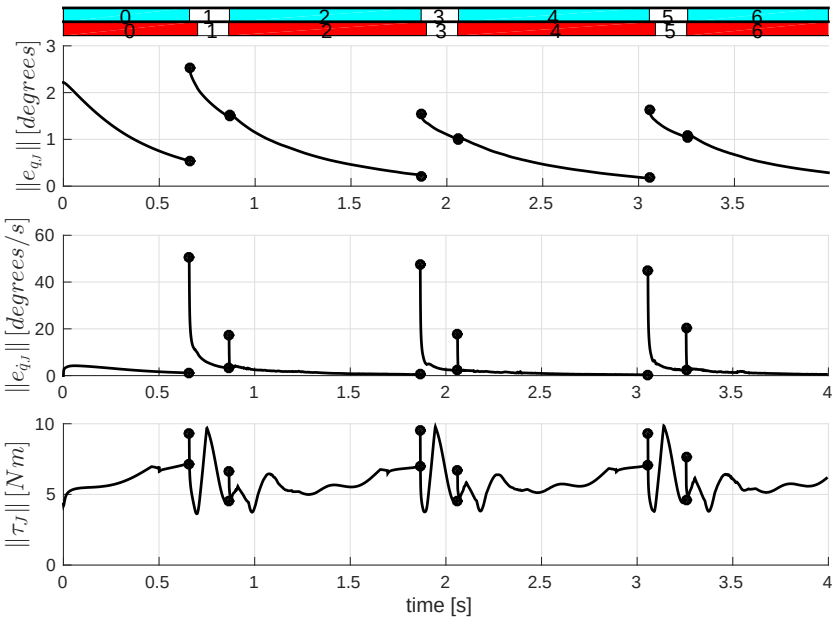


Figure 4.27: The joint angle and velocity error norms and the torque norm for a simulation with wall position $x_{wall} = 0.39m$ and feedback gains of Table 4.3. The error norms appears to slowly converge to zero, with an decreasing mismatch between the closed loop and nominal event times with each motion cycle.

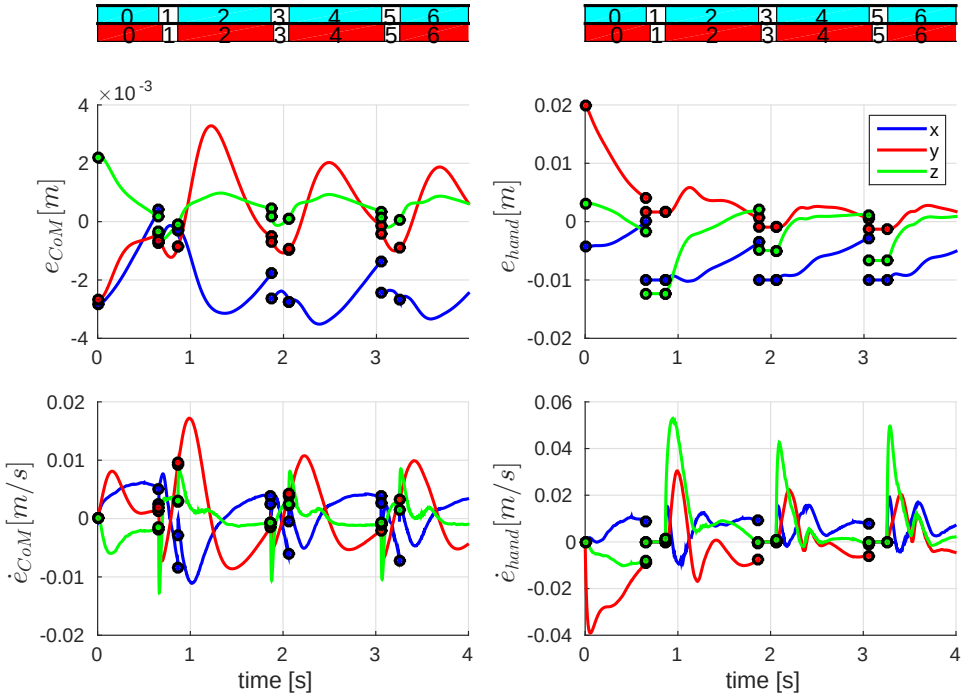


Figure 4.28: The position and velocity errors of the CoM and left hand for a simulation with wall position $x_{wall} = 0.39m$ and feedback gains of Table 4.3.

Although we can stabilize the contact motion of the iCub if we move the wall, no particular set of feedback gains seems to be particularly robust. We have to tune the feedback gains for each individual simulation. The performance of the controller is also quite poor. Both issues are caused by the fact that the reference is designed for a specific wall position. The joint configuration of the reference trajectory in contact mode cannot be achieved if the wall is not at the expected distance. This makes the RS hybrid controller suboptimal for dealing with this problem.

4.3 Discussion

The RS hybrid controller seems to be quite robust if no model imperfections are taken into account. Different perturbations in the initial conditions can be handled by the same controller with a single set of feedback gains. With the right feedback gains, the error is reduced to almost zero in a timeframe of 3 seconds, which is quite impressive for such an complex, unstable system.

When model imperfections are taking into account, the robustness of the RS hybrid controller is quite poor. A relatively small bias in encoders and actuators can be handled by the same controller with a single set of feedback gains. However, if we change the wall position, we have to tune the feedback gains for each individual scenario. Affine actuator functions, unless very small, could not be stabilized by the controller at all. The controller has no sense of balance or end effector position, but uses only a predefined feedforward and a joint feedback. This makes the RS hybrid controller less suitable for dealing with model imperfections, especially those that influence the feedforward.

Now that we provided a proof of concept and a brief performance analysis for the RS hybrid controller on the iCub, we will finish the report with the conclusions and recommendations.

Chapter 5

Conclusions and recommendations

5.1 Conclusions

In this report, we provided a proof of concept for the reference spreading based hybrid controller (RS hybrid controller) on the iCub humanoid robot. With this result, the main goal of this manuscript has been achieved. As detailed in Section 1.4 of the introduction, each chapter details one of the major steps in reaching this goal.

- Chapter 2 formulates a dynamical model of the iCub.
- Chapter 3 generates a hybrid reference trajectory for the contact task of the iCub.
- Chapter 4 applies the RS hybrid controller on simulations of the iCub.

The conclusions of each of these steps are detailed below.

Formulating a dynamical model of the iCub

In Chapter 2, a multibody notation was presented. Using this notation, a hybrid dynamical model for the iCub humanoid robot was presented, specifically designed for modeling a single motion task. The hybrid dynamical model has two modes, a free motion mode, where the robot stands on a single leg, and a contact mode, where the robot, standing on one leg, uses one of his hands to support itself against a wall. This model has been made with relatively simple contact dynamics, using unilateral constraints and a velocity reset map.

The dynamical model of the iCub was designed for one specific contact task. In this task, the iCub starts standing on one leg, and reaches for the wall with one hand. Once the robot touches the wall, the iCub pushes itself away, back to the initial configuration. This task can be repeated indefinitely. The hybrid dynamical model formulated in Chapter 2 is capable of simulating this contact task.

Generating a hybrid reference trajectory for the contact task of the iCub

In Chapter 3, we detailed the principle of the RS hybrid controller. The RS hybrid controller is a recently developed control strategy for hybrid dynamical systems with state jumps. It uses an extended hybrid reference trajectory, with the reference of each mode reaching beyond its expected timeframe. The controller selects the reference mode corresponding to the current mode of the actual system.

In Chapter 3, we created an extended hybrid reference trajectory for the contact task of the iCub, using a task-level optimal constrained controller. This optimal constrained controller calculates

the optimal control torque for performing a task, while constraining the control torque such that no contacts with the environment are broken. A control law for the RS hybrid controller was presented.

Applying the RS hybrid controller to the iCub in a simulations.

Chapter 4 presented several case studies for the RS hybrid controller of the iCub. The RS hybrid controller can successfully stabilize a simulation of the nominal case of the motion task. With this result, our goal of providing a proof of concept has been achieved. The controller displays favourable robustness properties for perturbations in the initial conditions and displays a relatively large region of attraction. The RS hybrid controller is less suitable for cases with model uncertainties. The controller displayed poor performance and unfavourable robustness properties for case studies with affine actuators or a perturbation on the wall position.

5.2 Recommendations

In this section, we provide recommendations for future research, based on the findings of this report.

The main recommendation of this report is to research the possibilities of more advanced RS hybrid controllers. Although our proof of concept shows promising results on simulations of the iCub, the shortcomings of the currently developed theory prevent the RS hybrid controller from being applicable on real-life humanoid robots. The control law used in this report is too simple, and lacks any sense of the iCubs balance. Using a state-dependent feedforward would be a good first step in improving the RS hybrid controller.

A particularly interesting topic of research would be to combine the RS hybrid controller and the optimal constrained controller detailed in this report. By converting the task trajectory of the optimal constrained controller to an extended hybrid trajectory, one could introduce the advantages of RS hybrid control to the already established optimal constrained control theory. Alternatively, one could try to combine the RS hybrid controller with a different kind of output feedback.

It would be interesting to research the RS hybrid controller in its current form on a more stable system. The case studies in this report were limited by the balancing issues of the iCub. Designing a task where the robot stands on both legs, or using a robot with 3 or 4 legs, could provide interesting insight on the performance and robustness of the RS hybrid controller. Additionally, it might be an interesting extension of the research in this report to study the RS hybrid controller on models in which no fixed mode switching sequences are incorporated.

Bibliography

[1] Jennifer Goetz, Sara Kiesler and Aaron Powers, Matching Robot Appearance and Behavior to Tasks to Improve Human-Robot Cooperation *Proceedings of the 2003 IEEE international Workshop on Robot end Human Interactive Communication*, page 6, 2003.

[2] Giorgio Metta. G. Sandini, D. Vernon, L.o Natale and F. Nori, The iCub humanoid robot: an open platform for research in embodied cognition *PerMIS '08 Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems*, pages 50-56, 2008.

[3] F. Romano, D. Pucci, S. Traversaro, J.b Eljaik and F. Nori, Whole-body Force Control of the iCub Humanoid Robot for Balancing Tasks *Conference: Workshop at the IEEE International Conference on Robotics and Automation*, oages 1-2, 2015.

[4] A. Parmiggiani , M. Randazzo , M. Maggiali , F. Elisei , G. Bailly and G. Metta, An articulated talking face for the iCub *IEEE-RAS International Conference on Humanoid Robots (Humanoids)* , page 2 , 2014.

[5] Sven Behnke, Humanoid Robots - From Fiction to Reality *KI-Zeitschrift*, pages 1-5, 2008.

[6] F. Nori, S. Traversaro , J. Eljaik , F. Roman , A. Del Prete and D. Pucci, iCub whole-body control through force regulation on rigid non-coplanar contacts *Frontiers in ROBOTICS AND AI*, page 1, 2015.

[7] Saccon, A., van de Wouw, N. and Nijmeijer, H, Sensitivity analysis of hybrid systems with state jumps with application to trajectory tracking *53rd IEEE Conference on Decision and Control*, page 2-6, 2014.

[8] Rijnen, M.W.L.M., Saccon, A., and Nijmeijer, H, On optimal trajectory tracking for mechanical systems with unilateral constraints. *54th IEEE Conference on Decision and Control*, pages 1-6, 2015.

[9] Tomomichi Sugihara, Yoshihiko Nakamura, Hirochika Inoue, Realtime Humanoid Motion Generation through ZMP Manipulation based on Inverted Pendulum Control *Robotics and Automation, 2002. Proceedings. ICRA '02.*, page 1, 2002.

[10] Joel Chestnutt, Manfred Lau, German Cheung, James Kuffner, Jessica Hodgins, and Takeo Kanade, Footstep Planning for the Honda ASIMO Humanoid *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, page 2, 2005.

[11] Jung-yup Kim , Ill-woo Park and Ju-ho Oh, Experimental realization of dynamic walking of the biped humanoid robot KHR-2 using zero moment point feedback and inertial measurement *Advanced Robotics , Vol. 20, No. 6, pp. 707 736*, pages 10-18, 2006.

[12] Bruno Siciliano Oussama Khatib, Springer Handbook of Robotics *Springer-Verlag Berlin Heidelberg*, pages 35-62, 2008.

[13] Silvio Traversaro, Alessandro Saccon, Multibody Dynamics Notation *Eindhoven University of Technology, internal communication*, pages 1-14 , 2016.

[14] Philippe Sardain and Guy Bessonnet, Forces Acting on a Biped Robot. Center of PressureZero Moment Point *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICSPART A: SYSTEMS AND HUMANS*, page 2 , 2004.

[15] Rafal Goebel, Ricardo G. Sanfelice and Andrew R. Teel, Hybrid Dynamical Systems: Modeling, Stability, and Robustness *Princeton university press*, pages 1-22, 2012.

[16] J. Baumgarte, Stabilization of Constraints and Integrals of Motion in Dynamical Systems *Computer Methods in Applied Mechanics and Engineering 1*, pages 1-16, 1972.

[17] Bernard Brogliato, Nonsmooth mechanics - Models, Dynamics and Control *Springer-Verlag London*, pages 1-53, 1999.

[18] Remco Leine and Nathan van de Wouw, Stability and Convergence of Mechanical Systems with Unilateral Constraints, Chapter 5 *Springer*, pages 79-106, 2008.

[19] J. J. Benjamin Biemond, Nathan van de Wouw, W. P. Maurice H. Heemels, and Hend rik Nijmeijer, Tracking Control for Hybrid Systems With State-Triggered Jumps *IEEE Transactions on Automatic Control*, page 1, 2012.

[20] Andrea Del Prete, Control of Contact Forces using Whole-Body Force and Tactile Sensors: Theory and Implementation on the iCub Humanoid Robot *Ph.D. Dissertation, Universitá di Genova, Istituto Italiano di Tecnologia*, pages 35-102, 2013.