SAPIENZA
UNIVERSITÀ DI ROMA

Humanoid Gait Generation via MPC: Stability, Robustness and Extensions

Sapienza University of Rome

Dottorato di Ricerca in Automatica, Bioingegneria e Ricerca Operativa – XXXII Ciclo

Candidate

Nicola Scianca
ID number 1047939

Thesis Advisor

Prof. Giuseppe Oriolo

January 2020

Humanoid Gait Generation via MPC: Stability, Robustness and Extensions
Ph.D. thesis. Sapienza – University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: scianca@diag.uniroma1.it

Abstract

Research on humanoid robots has made significant progress in recent years, and Model Predictive Control (MPC) has seen great applicability as a technique for gait generation. The main advantages of MPC are the possibility of enforcing constraints on state and inputs, and the constant replanning which grants a degree of robustness.

This thesis describes a framework based on MPC for humanoid gait generation, and analyzes some theoretical aspects which have often been neglected. In particular, the stability of the controller is proved. Due to the presence of constraints, this requires proving recursive feasibility, i.e., that the algorithm is able to recursively guarantee that a solution satisfying the constraints is found. The scheme is referred to as Intrinsically Stable MPC (IS-MPC).

A basic scheme is presented, and its stability and feasibility guarantees are discussed. Then, several extensions are introduced. The guarantees of the basic scheme are carried over to a robust version of IS-MPC. Furthermore, extension to uneven ground and to a more accurate multi-mass model are discussed.

Experiments on two robotic platforms (the humanoid robots HRP-4 and NAO) are presented in the concluding section.

Acknowledgements

I would like to thank: my family for their constant support; Prof. Giuseppe Oriolo and Prof. Leonardo Lanari for their guidance throughout these three years; everyone at the Robotics Lab in Rome, in particular my co-authors Daniele De Simone, Paolo Ferrari, Valerio Modugno, Marco Cognetti; the students who I worked with and whose work contributed to this thesis: Filippo M. Smaldone, Alessio Zamparelli, Ahmed Aboudonia; Prof. Francesco Borrelli for hosting me at University of California at Berkeley for six months, which was a very stimulating experience; everyone at the MPC Lab in Berkeley, in particular Ugo Rosolia who I had the great pleasure to collaborate with.

Contents

Chapter 1

Introduction

1.1 About humanoid robots

Humanoid robots have a structure that is closely inspired by human morphology. They typically have an anthropomorphic appearance, with arms and legs which allow them to manipulate objects and to perform human-like locomotion, and they may have human-like senses such as a camera mounted on the head to mimic our visual perception.

As of now humanoid robots still have little applicability outside research. Nevertheless, they have and have always had a prominent place in popular culture. The general public typically associates the word ‘robot’ to some kind of artificial human-like being. Many classic works of science fiction feature robots which have human appearance, sometimes only as a structural resemblance, sometimes to the point they are indistinguishable from us. It is undeniable that humanoid robots evoke great interest from the public.

Significant early progress towards the construction of humanoid robots was made in Japan, starting around 1973 at Waseda University in Tokyo [1]. WABOT-1 was a full-scale anthropomorphic robot which was able to walk on its legs, and its successor WABOT-2 had hands which made him capable of playing a keyboard instrument. The Japanese tradition was picked up by Honda, which in 1986 started a research program to produce humanoids that could coexist with humans in their environments. Their first humanoid robot was P2, revealed to the public in 1996. It was followed by P3 in 1997 and by Asimo in 2000 (see Fig. 1.1).

More modern platforms include HRP-4 from Kawada Robotics, Atlas from Boston Dynamics, Valkyrie from NASA, and many others (see Fig. 1.2). The hardware side has seen great improvements over the years, and some companies are even starting to show interest in employing humanoids for industrial purposes [2].

Despite the significant progress made over the years, humanoids still have not reached the level of performance that is expected from them. Locomotion in complex environments, where the ground is not flat and obstacles are present, is still a challenge. Unstructured environments are exactly the kind of scenarios in which humanoid robots would prove most useful. Examples are disaster areas with high heat or toxic chemicals, where robots might be needed for exploration and rescue. Wheeled robots would not be able to easily navigate a cluttered scene, especially if

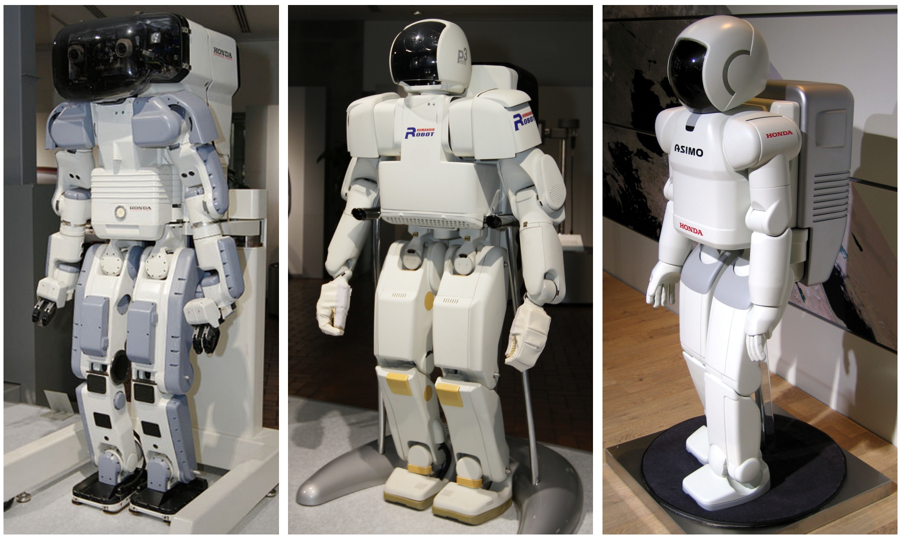


Figure 1.1. Early humanoid robots developed by Honda. Left to right: P2 from 1996, P3 from 1997, and the first version of Asimo from 2000.

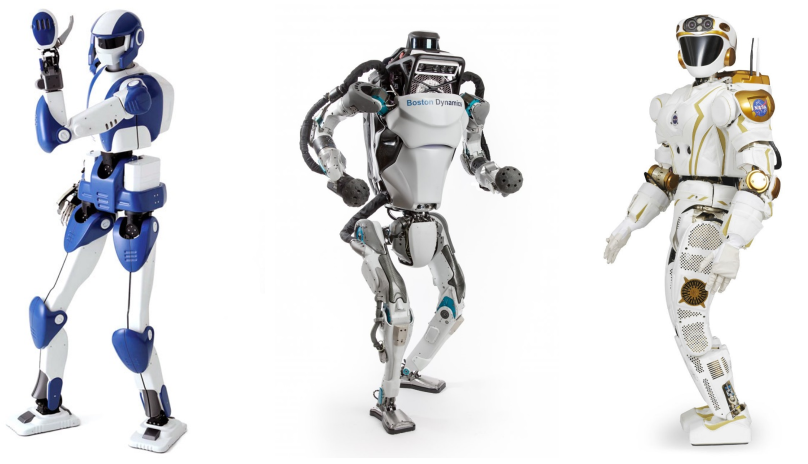


Figure 1.2. Modern humanoid robots: HRP-4 from Kawada Robotics, Atlas from Boston Dynamics and Valkyrie from NASA.

it is necessary to step over obstacles. Humanoids on the other hand can achieve this level of dexterity, at least in principle, but the control techniques need to proceed hand in hand with the technological advances in construction. This emphasizes the need for a strong theoretical foundation of the techniques we develop.

1.2 About Model Predictive Control

Several control techniques determine actions as solutions to optimization problems. A classic example is found in optimal control, which is typically employed to compute off-line reference trajectories that can be employed as feedforward control sequences. The solution is computed in its entirety and usually tracked by another controller (e.g., a PID).

The goal of MPC is to use optimal control techniques in a real-time fashion. The prediction is limited to a short horizon, which typically produces a sub-optimal trajectory. However, the prediction window is shifted ahead at each time-step, and the trajectory is always recomputed, with only the first sample of the predicted control sequence being applied at each time. This means that inaccuracies in the prediction are kept under control, as they are absorbed by the constant replanning. Furthermore, it allows imposing constraints on the state and input.

Achieving real-time execution might require, especially on a complicated system such as a humanoid robot, a simplification of the prediction model. The simplified dynamic model is usually a linear system, which allows the controller to run at a frequency of 100 Hz (or even more), which is a suitable frequency for a robotic system.

The presence of constraints might pose problems of *feasibility*, i.e., whether or not a solution exists at a given time step. Classic MPC theory usually tackles this by employing a *terminal constraint*, i.e., a constraint on the state at the last time-step of the prediction, which is constructed so as to guarantee *recursive feasibility* of the scheme. This is a necessary step for proving stability.

MPC was extensively used in the control of humanoid robots in the last few years, with many interesting results being proposed. However, there is progress to be made in the theoretical analysis of these control techniques. This work attempts to take steps in this direction, by formulating an MPC scheme for humanoid gait generation and analysing its properties with regards to stability and feasibility.

1.3 Overview of the thesis

This thesis will describe a gait generation technique for humanoid robots which we refer to as Intrinsically Stable MPC (IS-MPC). The name is due to the use of an explicit *stability constraint* in the scheme, which provides boundedness guarantees on the generated trajectories. IS-MPC was proposed in [3], and later revised and expanded in [4]. Extensions of the basic scheme will also be discussed. What follows is a brief outline of the content of each of the following chapters.

Ch. 2 will present a review of the literature and of the state of the art with respect to MPC for humanoid locomotion.

Ch. 3 will discuss the dynamic balance of humanoids and introduce models that will be employed in later chapters. Each of the models will be used later, either by the basic IS-MPC scheme or by one of its extensions.

Ch. 4 will introduce the stability condition. This is a tool that will be utilized throughout the work, and that is required to construct the aforementioned stability constraint for the MPC scheme.

Ch. 5 will describe IS-MPC in its basic form, and Ch. 6 will introduce automatic footstep placement, which allows reactive stepping. The material for these chapters is found in [3] and [4]. A section based on [5] is also included in Ch. 6, for automatic footstep placement which includes adaptation of the footstep orientations.

Ch. 7 will discuss a robust extension of the scheme. This includes material found in [6], which is concerned with the design of a stability constraint for a perturbed model, providing an indirect compensation of the disturbance. More recent work is also included in the same chapter.

Ch. 8 presents an extension for locomotion on uneven ground. This work was proposed in [7], and later extended in [8] to include a footstep planner capable of generating footstep sequences on uneven ground.

Ch. 9 describes an extension that employs a multi-mass model. This constitutes a more sophisticated model as it accounts for the dynamics of the swinging foot. The original work can be found in [9].

IS-MPC has also been applied to the generation of evasive motions for human-humanoid coexistence [10].

Chapter 2

Literature review

A large part of the literature on humanoid robot control (and legged robots in general) uses the concept of Zero Moment Point (ZMP). First introduced by Vukobratović and Juriĉić [11], its strength is that it gives a geometric interpretation to the problem of balance: the ZMP must lie inside the support polygon.

Still, the full ZMP dynamics are too complex for real-time control. A simplified model, employed by several approaches for locomotion, is the Linear Inverted Pendulum (LIP) [12], which is obtained by the Newton-Euler equations on the full body of the robot, with a few simplifying assumptions. These assumptions require a flat ground, a constant CoM height, and neglecting the rate of change of angular momentum around the robot CoM. The LIP model and the assumptions leading to it will be explained in detail in Sect. 3.2). The concept of Capture Point (CP), the point on the ground on which the robot should step to maintain balance [13] is also closely related to the LIP.

The LIP model allows to control the ZMP dynamics by generating a suitable CoM trajectory and then kinematically tracking this trajectory. Furthermore, the linearity of the model allows to design controllers that use predictive techniques. A major step in this direction was made by [14], which generated a CoM trajectory using a *preview controller*. A ZMP trajectory is designed over the prediction horizon, and the control objective is to track this trajectory while minimizing the CoM jerk (i.e., its third derivative), by solving a finite time optimal control problem over the prediction.

The inclusion of explicit ZMP constraints in the preview controller [15] effectively leads to an MPC formulation. This has several advantages, as the ZMP constraints guarantee dynamic balance, and remove the need to specify a ZMP trajectory to be tracked.

MPC has a very rich literature, but the gait generation problem is rendered more complex by the fact that it deals with time-varying ZMP constraints. This makes the analysis of its properties, more challenging. Classically in MPC design, stability is guaranteed by using a terminal constraint. Terminal constraints for standard control problems such as regulation or tracking are easily found in the literature, but the problem of gait generation does not fall into those categories, as the trajectory is not pre-specified.

Terminal constraints were used for humanoid balancing in [16] and for gait generation in [17]. The latter makes use of a LIP model, requiring its unstable component to stop at the end of the control horizon, and it is referred to as *capturability constraint* (from the concept of capture point [18]). This constraint has also been used in [19], where it is imposed only at the foot landing instant, and in [20], which addresses locomotion in a multi-contact setting.

Terminal constraints may have a detrimental effect on *feasibility*, i.e., the existence of solutions for the optimization problem which is at the core of any MPC scheme [21]. A particularly desirable property is *recursive feasibility*, which entails that if the optimization problem is feasible at a certain iteration it will remain such in future iterations. It appears that this crucial issue has seldom been explored for MPC-based gait generation, with the notable exceptions of [22, 23].

MPC for gait generation has been widely adopted, and several control schemes have been proposed over the years. In [24] an MPC controller is formulated on the CP dynamics, while in [25] the controller is a nonlinear MPC which includes an obstacle avoidance constraint. In [26] the authors formulate an MPC with binary variables to approximate foot orientation constraints and solve it using Mixed-Integer Quadratic Programming. In [27] the authors propose a LIP-based MPC with a low-level controller for torque-controlled humanoids, and in [28] an MPC includes additional considerations to reduce the robot angular momentum, for increased robustness. An interesting enhancement of [15] allows the MPC to autonomously assign the position of the footsteps. This is known as *automatic footstep placement*, first proposed in [29] as a means of adapting a pre-existing plan, and then developed in [30, 31] to directly place foot positions based on a reference angular velocity. It allows the robot to perform *reactive stepping* to accommodate perturbations, all while keeping the constraints linear and thus with no significant increase in computation time.

An important line of research is concerned with maintaining balance, or stable locomotion, when the robot is subject to perturbations. Robust MPC can be formulated as the problem of satisfying the constraints under the effect of bounded disturbances, for which there exist numerous formulations in the literature [32, 33, 34]. A similar idea is used on humanoids in [35], where safety margins are derived to cope with a given set of uncertainties.

Another path to robustness is to build a disturbance observer and design a controller based on the disturbance estimate. A first example is [36], where an external force is estimated from its effect on the humanoid. Other examples include [37] which incorporates an observer in a preview controller, or techniques based on the divergent component of motion [38, 39].

A strong research topic is constituted by locomotion on uneven ground, for which gait generation using the LIP model is too restrictive, as vertical motions of the CoM introduce a nonlinear term in the dynamics. A possible approach, presented in [40], is to bound the nonlinear term, which can then be taken into account by redesigning the ZMP constraints. It is possible to accept the nonlinearity and solve the nonlinear MPC problem that derives from it, as in [41], or [42] where capturability-based concepts are extended to the 3D case.

Using a pre-assigned vertical CoM motion simplifies the problem and results in a time-varying frequency of the inverted pendulum. This is used in [43, 44] and also [45] for the transition between bipedal and quadrupedal locomotion. It is possible however to have a linear model without pre-specifying the CoM height, thus maintaining a structure that is very similar to that of the LIP. This requires limiting the possible trajectories of the CoM [46, 47]. [48] employs a related approach by defining a 3D extension of the DCM and the Virtual Repellent Point (VRP), which is a close relative of the ZMP for the 3D case.

Chapter 3

Models for humanoid locomotion

Humans, as well as human-inspired robots, achieve locomotion by walking. We walk by constantly exhanging forces with the environment, exploiting friction with the ground to move our body forward in space. This requires regularly switching the contact point with the ground while moving, to produce a delicate *dynamic balance* problem.

The contact is normally mediated by our feet. In particular, a foot in contact with the ground is called *support foot.* When both feet are on the ground we are in a *double support* phase. If this is not the case, we are in a *single support* phase, and the foot that is not in contact with the ground is called the *swing foot.*

Humanoid robots also realize walking by alternating single and double support phases. When designing a controller, the challenge is to make sure that during any of these phases the robot does not lose its balance and fall. The most common way of achieving this is to control the ZMP so that it is at all times within the convex hull of the contact points between the robot and the ground. This region is called the *support polygon.*

The support polygon changes size and shape between different phases of the walk. During single support it can be considered equivalent to the contact surface of the support foot, while during double support it includes the contact surfaces of both feet, as well as the area between them, making it considerably larger. In fact, the double support phase can be thought of as having the function of *transferring* the ZMP from one foot to the other, thus connecting separate single support phases.

For a slower walk, the ZMP is very close to the projection of the CoM on the ground. This is usually called a *static walk*, as the robot could be stopped at any point and it would be statically stable. On the converse, a fast walk exhibits CoM and ZMP trajectories which are very distinct, and is commonly referred to as *dynamic walk.*

Since we are interested in producing a dynamic walk, we need to derive a mathematical model which describes the dynamics of the ZMP. This chapter will start by deriving this dynamic model for a general case. This model will still be too complex for designing a real-time controller, and a few assumptions will be made, leading to the commonly employed Linear Inverted Pendulum (LIP). Later sections of this chapter will discuss other simplified models which turn out to be useful in a range of situations.

3.1 Forces acting on a humanoid robot

The first step is characterizing the forces that a humanoid robot exchanges with the environment. Under normal circumstances, the forces we need to take into account, also shown in Fig. 3.1, are:

- gravity $m\boldsymbol{g} = (0, 0, -mg)^T$, applied at the CoM $\boldsymbol{p}_c$;
- contact forces with the ground $\boldsymbol{f}^i = (f^i_x, f^i_y, f^i_z)^T$, where the horizontal components f^i_x and f^i_y are due to friction and the vertical component is the upward ground reaction force. The point of application of each force $\boldsymbol{f}^i$ is $\boldsymbol{p}^i_f$.

These forces are balanced by the Newton equation

$$m\ddot{\boldsymbol{p}}_c = m\boldsymbol{g} + \sum_i \boldsymbol{f}^i. \tag{3.1}$$

The non-tilting condition is given by moment balance around a generic point $\boldsymbol{p}_o$

$$(\boldsymbol{p}_c - \boldsymbol{p}_o) \times m\ddot{\boldsymbol{p}}_c + \dot{\boldsymbol{L}} = (\boldsymbol{p}_c - \boldsymbol{p}_o) \times m\boldsymbol{g} + \boldsymbol{M}_f, \tag{3.2}$$

with $\boldsymbol{L}$ the angular momentum around the CoM, and

$$\boldsymbol{M}_f = \sum_i (\boldsymbol{p}^i_f - \boldsymbol{p}_o) \times \boldsymbol{f}^i \tag{3.3}$$

the moment of the contact forces. The ZMP is defined as the point which, by computing moments w.r.t. this point, gives $\boldsymbol{M}_f = 0$. We denote the ZMP as $\boldsymbol{p}_z$. Replacing in (3.2) yields

$$m(\boldsymbol{p}_c - \boldsymbol{p}_z) \times (\ddot{\boldsymbol{p}}_c - \boldsymbol{g}) + \dot{\boldsymbol{L}} = 0. \tag{3.4}$$

The ZMP is not required to lie on the ground surface. In fact, a line of points exists, all of which have the characteristic of giving $\boldsymbol{M}_f = 0$. However, it is usually defined as the intersection of this line with the ground, and here we will employ this definition. Later we will extend the ZMP to a point in space in order to derive a 3D model (see Sect. 3.4).

If the contacts are coplanar (i.e., the ground is flat), by dividing (3.4) by the vertical projection of (3.1), we obtain that

$$\frac{m(\boldsymbol{p}_c - \boldsymbol{p}_z) \times (\ddot{\boldsymbol{p}}_c - \boldsymbol{g}) + \dot{\boldsymbol{L}}}{m(\ddot{z}_c + g)} = \frac{\sum_i (\boldsymbol{p}^i_f - \boldsymbol{p}_z) \times \boldsymbol{f}^i}{\sum_i f^i_z} = \boldsymbol{p}_p - \boldsymbol{p}_z = 0, \tag{3.5}$$

where

$$\boldsymbol{p}_p = \frac{\sum_i \boldsymbol{p}^i_f \times \boldsymbol{f}^i}{\sum_i f^i_z} \tag{3.6}$$

is the Center of Pressure (CoP). The CoP, for coplanar contacts, defines the point of application of the resultant of the contact forces. Implicitly it requires a flat

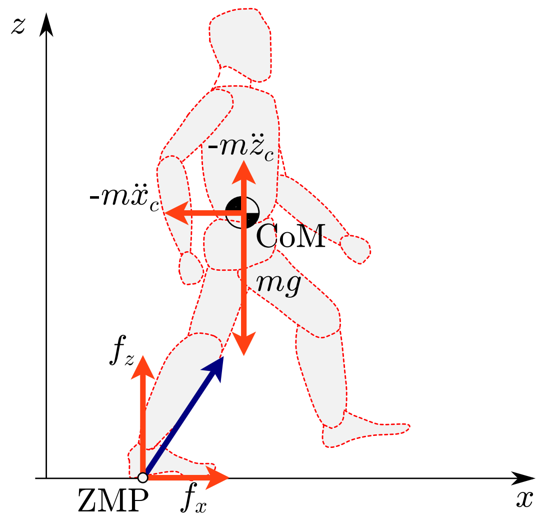


Figure 3.1. Forces acting on a humanoid robot: gravity, inertial forces, and the resultant of the contact forces.

ground assumption, however, it can be extended to a pair of contacts at different heights by defining a virtual surface varying continuously from the first to the second surface [49].

Since, from (3.5), we find that the ZMP on the ground surface coincides with the CoP, a condition for balance is given by keeping the ZMP within the robot support polygon (the convex hull of the contact points).

It is useful to project the vector equation (3.8) on the x and y axes. Consider the projection on the y axis

$$m(z_c - z_z)\ddot{x}_c - m(x_c - x_z)(\ddot{z}_c + g) + \dot{L}_y = 0, \tag{3.7}$$

which by reordering the terms gives

$$\ddot{x}_c = \frac{\ddot{z}_c + g}{z_c - z_z}(x_c - x_z) - \frac{\dot{L}_y}{m z_c}. \tag{3.8}$$

This equation (along with its equivalent obtained by projection on x) relates the ZMP to the CoM position and acceleration, and to the variation of angular momentum given by changes of the internal configuration of the robot.

The rate of change of angular momentum $\dot{L}$ can in principle be expressed as a function of joint positions q and their derivatives $\dot{q}$ and $\ddot{q}$ [50]. However, we are interested in deriving a simplified model which can be used in a real-time MPC setting, which will be the subject of the following section.

3.2 The Linear Inverted Pendulum

The dynamics expressed by (3.8) include nonlinearities which make it unsuitable for a real-time MPC formulation. What follows is a set of simplifying assumptions that are widely employed for controller design purposes [12]. Some of these will be removed in later sections to derive different and more sophisticated models.

Assumption 1 *The ZMP lies on the ground, i.e.,* $z_z = 0$.

This assumption has the ZMP coincide with the CoP. This simplifies the balance condition as it can now be expressed as a geometric condition on the x-y plane, namely, the ZMP needs to be inside to the support polygon at all times.

Assumption 2 *The contribution of the rate of change of angular momentum* $\dot{L}$ *can be neglected.*

This is a very widely employed simplification as it removes most nonlinear contributions to the dynamics.

Assumption 3 *The CoM height is constant equal to* $\bar{z}_c$

This assumption removes the only remaining nonlinearity which is the coupling between the horizontal and vertical components of the CoM. What results is the linear model

$$\ddot{x}_c = \eta^2(x_c - x_z) \tag{3.9}$$

where $\eta = \sqrt{g/\bar{z}_c}$.

This model can be interpreted as a Linear Inverted Pendulum (LIP) by considering the ZMP as the input, or as a Cart-Table (CT) by taking the CoM acceleration as the input. We adopt the first interpretation which leads to the following state-space formulation

$$\begin{pmatrix} \dot{x}_c \\ \ddot{x}_c \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ \eta^2 & 0 \end{pmatrix} \begin{pmatrix} x_c \\ \dot{x}_c \end{pmatrix} + \begin{pmatrix} 0 \\ -\eta^2 \end{pmatrix} x_z \tag{3.10}$$

This system has two modes, with associated eigenvalues $\pm\eta$. These can be decoupled by the change of coordinates [51]

$$\begin{aligned} x_u &= x_c + \dot{x}_c/\eta \\ x_s &= x_c - \dot{x}_c/\eta \end{aligned} \tag{3.11}$$

which leads to the decoupled dynamics

$$\begin{aligned} \dot{x}_u &= \eta(x_u - x_z) \\ \dot{x}_s &= -\eta(x_s - x_z). \end{aligned} \tag{3.12}$$

x_s and x_u represent the stable and unstable components of the LIP dynamics. We will focus almost exclusively on the unstable component, as we are interested in preventing the CoM from diverging with respect to the ZMP.

This decoupling of the dynamics has seen numerous applications, and the unstable component x_u has been discussed under different names, which usually emphasize different interpretations. In particular, it is equivalent to the Capture Point [13], the Extrapolated Center of Mass [52] and the Divergent Component of Motion [53].

3.3 The perturbed LIP model

A robotic system is usually subject to disturbances of many different kinds. Some can be internal, in the form of inaccuracies due to model simplification. Others can be external, such as forces applied on the system.

To represent these kinds of disturbances, we define a perturbed LIP model. With respect to (3.9), this model has the additional disturbance term $w(t)$

$$\ddot{x}_c = \eta^2(x_c - x_z) + w. \tag{3.13}$$

As for the LIP, system (3.13) can be decomposed in stable and unstable component using the same change of coordinates (3.11). The resulting system is given by

$$\begin{aligned} \dot{x}_u &= \eta(x_u - x_z) + w/\eta \\ \dot{x}_s &= -\eta(x_s - x_z) + w/\eta, \end{aligned} \tag{3.14}$$

which is a decoupled system similar to (3.12), where both subsystems are affected by the disturbance.

Let us further characterize the significance of the disturbance term w. Assume it should model both internal and external disturbances, it is possible to derive an expression which characterizes the contribution for the different source of disturbance. Start from the moment balance (3.4) with an additional external force $\boldsymbol{f}_{\text{ext}}$. Suppose for simplicity this force is aligned with the x-axis, i.e., $\boldsymbol{f}_{\text{ext}} = (f_{\text{ext}}, 0, 0)^T$, although a more general expression can be derived.

$$(\boldsymbol{p}_c - \boldsymbol{p}_z) \times m\ddot{\boldsymbol{p}}_c + \dot{\boldsymbol{L}} = (\boldsymbol{p}_c - \boldsymbol{p}_z) \times m\boldsymbol{g} + (\boldsymbol{p}_p - \boldsymbol{p}_z) \times \boldsymbol{f}_{\text{ext}} \tag{3.15}$$

Projecting this equation on the y axis (projection on x is similar) gives

$$m\ddot{x}_c z_c - m\ddot{z}_c(x_c - x_z) + \dot{L}_y = mg(x_c - x_z) + f_{ext} z_c \tag{3.16}$$

after which, by solving for $x_c - x_z$, we get

$$x_c - x_z = \frac{\dot{L}_y}{m(\ddot{z}_c + g)} + \frac{\ddot{x}_c z_c}{\ddot{z}_c + g} - \frac{f_{ext} z_c}{m(\ddot{z}_c + g)} \tag{3.17}$$

This expression can be substituted in the perturbed LIP equation (3.13)

$$\ddot{x}_c = \eta^2 \left(\frac{\dot{L}_y}{m(\ddot{z}_c + g)} + \frac{\ddot{x}_c z_c}{\ddot{z}_c + g} - \frac{f_{ext} z_c}{m(\ddot{z}_c + g)} \right) + w \tag{3.18}$$

and solving for w gives

$$w = \ddot{x}_c \left(1 - \eta^2 \frac{z_c}{\ddot{z}_c + g} \right) - \eta^2 \frac{\dot{L}_y}{m(\ddot{z}_c + g)} + \eta^2 \frac{f_{ext} z_c}{m(\ddot{z}_c + g)}. \tag{3.19}$$

If we have a way of estimating this disturbance term, such as with an observer, we can design a controller which uses the perturbed model for improved accuracy. This will be discussed in later chapters.

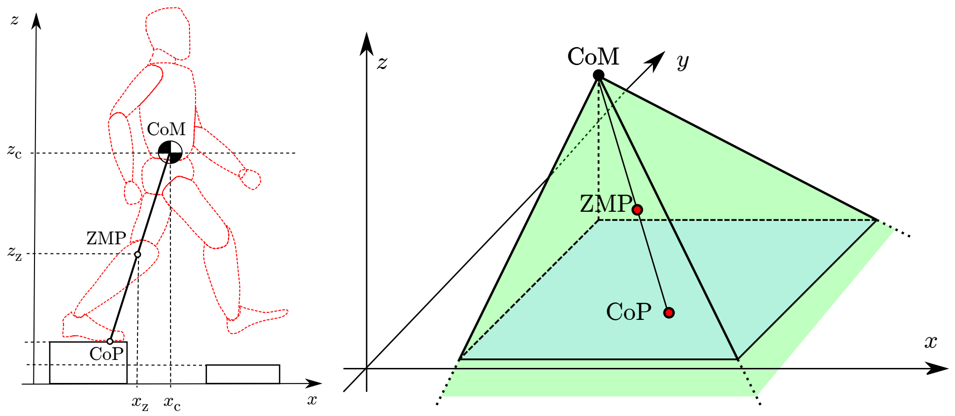


Figure 3.2. Left: 3D extension of the LIP, with the ZMP above the ground. Right: the balance condition in 3D case requires the ZMP to be in the green conic polyhedron.

3.4 Linear 3D model

In Sect. 3.2 we assumed a constant height of the CoM to render the model linear. This is very common, but it has the disadvantage of only allowing walking on flat ground. For generating walks on uneven ground the CoM necessarily needs to move in the vertical direction. In this section we will remove Assumptions 1 and 3, and show that this can still lead to a linear model [7]. The balance condition will need to be modified accordingly as it cannot be expressed as a planar condition.

Removing Assumption 1 lets the ZMP move in the vertical direction, as it is not confined to the ground [54]. However, the balance condition still needs to apply to the CoP, which has to lie inside the support polygon. The geometric interpretation given in Fig. 3.2 will help to understand how these two are related. The ZMP is on the line connecting the CoP to the CoM, so the balance condition translates to the ZMP being inside a *conic polyhedron*, as shown in Fig. 3.2. The top vertex of the polyhedron being the CoM makes this a nonlinear constraint, which will need to be linearized in the controller design phase.

To derive a linear model from (3.8) we set

$$\frac{\ddot{z}_c + g}{z_c - z_z} = \eta^2. \tag{3.20}$$

This turns the x and y component into the usual LIP equations, and adds a third dynamic equation which is directly derived from (3.20)

$$\begin{aligned}\ddot{x}_c &= \eta^2(x_c - x_z)\\ \ddot{y}_c &= \eta^2(y_c - y_z)\\ \ddot{z}_c &= \eta^2(z_c - z_z) - g.\end{aligned} \tag{3.21}$$

The vertical dynamics turns out to be very similar to a LIP, but have an additional constant additive term $-g$. This implies that the equilibrium is obtained when the CoM and ZMP are displaced by g/η^2.

This model can also be decoupled in a stable and unstable component, as it was done for the LIP in Sect. 3.2. The x and y component are unchanged so they give the exact same decoupled equation, and the vertical component results in

$$\begin{aligned}\dot{z}_u &= \eta(z_u - z_z) - g/\eta \\ \dot{z}_s &= -\eta(z_s - z_z) - g/\eta.\end{aligned} \tag{3.22}$$

3.5 Two-mass model

Neglecting the derivative of the angular momentum around the CoM, as required by Assumption 2, is equivalent to modeling the robot as a point mass. This is quite reasonable if the internal configuration of the robot does not change very abruptly. On the other hand, in some situations, this might lead to an inaccurate ZMP prediction, which can easily result in a fall.

A large part of the neglected angular momentum derivative is concentrated in the motion of the swinging leg, as the support leg is relatively still and the arms do not participate significantly in the walk. Therefore we can model the humanoid as a two-mass model, to recuperate part of the contribution from the neglected angular momentum derivative. This does not completely remove Assumption 2, but constitutes some sort of compensation for the model mismatch it introduces.

In this model the system can be represented as a *primary mass* M and a *secondary mass* m. The primary mass represents the contribution of the body, and the secondary mass represents the contribution of the swing leg. Since there is only one leg swinging at any given time, a single mass can alternatively account for both the left and the right leg.

We are interested in deriving a model describing the dynamics of the ZMP of the two-mass system $\boldsymbol{p}_{z,\mathrm{tot}}$, which following Assumption 1 still corresponds to the point of application of the equivalent ground reaction force. To find this model we consider the total moment balance around the ZMP for a two-mass system

$$M(\boldsymbol{p}_M - \boldsymbol{p}_{z,\mathrm{tot}}) \times \ddot{\boldsymbol{p}}_M + m(\boldsymbol{p}_m - \boldsymbol{p}_{z,\mathrm{tot}}) \times \ddot{\boldsymbol{p}}_m = (m+M)(\boldsymbol{p}_c - \boldsymbol{p}_{z,\mathrm{tot}}) \times \boldsymbol{g}. \tag{3.23}$$

where $\boldsymbol{p}_M$ and $\boldsymbol{p}_m$ are the positions respectively of the primary and secondary mass. This equation projected along the y axis gives

$$M z_M \ddot{x}_M - M(g+\ddot{z}_M)(x_M - x_{z,\mathrm{tot}}) + m z_m \ddot{x}_m - m(g+\ddot{z}_m)(x_m - x_{z,\mathrm{tot}}) = 0. \tag{3.24}$$

Define the two quantities

$$\begin{aligned}x_{z,M} &= x_M - \frac{z_M}{\ddot{z}_M + g}\ddot{x}_M \\ x_{z,m} &= x_m - \frac{z_m}{\ddot{z}_m + g}\ddot{x}_m,\end{aligned} \tag{3.25}$$

which can be interpreted as two individual IP equation, of mass respectively M and m.

Equation (3.24) can now be rearranged as

$$(\ddot{z}_M + g)(x_{z,\mathrm{tot}} - x_{z,M}) = \frac{m}{M}(\ddot{z}_m + g)(x_{z,m} - x_{z,\mathrm{tot}}) \tag{3.26}$$

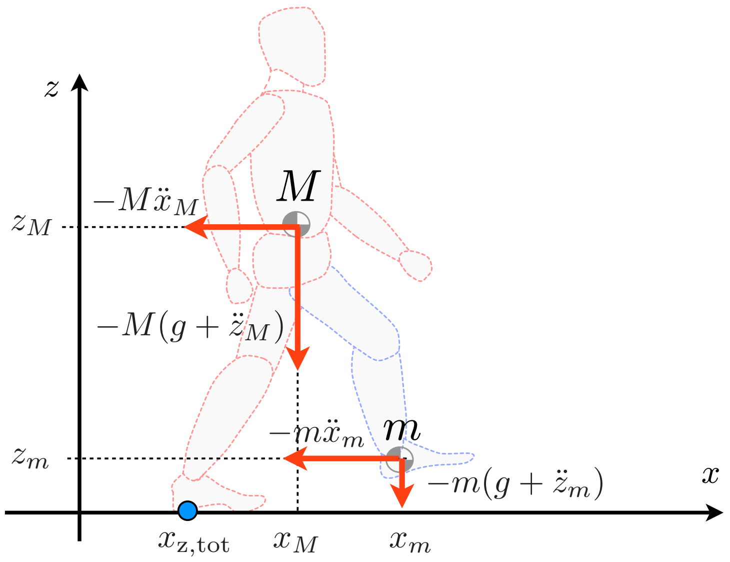


Figure 3.3. 2-mass model with the secondary mass corresponding to the swinging foot.

Here, $x_{z,\text{tot}} - x_{z,M}$ represents the contribution of m to the total ZMP. We can solve (3.26) for $x_{z,\text{tot}}$ as

$$x_{z,\text{tot}} = \left(1 + \frac{m}{M}\frac{\ddot{z}_m + g}{\ddot{z}_M + g}\right)^{-1}\left(x_{z,M} + \frac{m}{M}\frac{\ddot{z}_m + g}{\ddot{z}_M + g}x_{\text{z,m}}\right) \tag{3.27}$$

It is important to stress that this total ZMP differs from the ZMP of an IP with mass $m + M$ concentrated in (x_c, z_c)

$$x_c = \frac{Mx_M + mx_m}{m + M}, \qquad z_c = \frac{Mz_M + mz_m}{m + M} \tag{3.28}$$

i.e., in the CoM of our 2-mass model. In fact, the corresponding ZMP $x_{z,\text{z,IP}}$ satisfies the following moment balance

$$(m + M)z_c\ddot{x}_c - (m + M)(g + \ddot{z}_c)(x_c - x_{\text{z,IP}}) = 0 \tag{3.29}$$

which, using (3.28), can be shown to differ from (3.24). The difference $x_{z,\text{tot}} - x_{\text{z,IP}}$ can be regarded as an increase in ZMP prediction accuracy obtained by moving from the 1-mass IP to the 2-mass model.

The previous modeling approach can be extended to the case of N masses with a single primary mass M (e.g. [55]). Denoting by (x_i, z_i) the coordinates of the i-th secondary mass m_i, the total ZMP can be rewritten in a form that generalizes (3.27):

$$x_{z,\text{tot}} = \left(1 + \sum_{i=1}^{N-1}\frac{m_i}{M}\frac{\ddot{z}_i + g}{\ddot{z}_M + g}\right)^{-1}\left(x_{z,M} + \sum_{i=1}^{N-1}\frac{m_i}{M}\frac{\ddot{z}_i + g}{\ddot{z}_M + g}x_{z,i}\right)$$

where $x_{z,i}$ represents the equivalent of (3.25) for each mass m_i.

Going back to the case $N = 2$, assume a constant height z_M for the primary mass to obtain a linear system and thus (3.26) leads to the well-known LIP equation which will be called *equivalent LIP*

$$x_M - \frac{1}{\eta_M^2}\ddot{x}_M = x_{z,M} \tag{3.30}$$

where $\eta_M^2 = g/z_M$ and

$$x_{z,M} = x_{z,\text{tot}} + \frac{m}{M}\left(\frac{\ddot{z}_m + g}{g}\right)(x_{z,\text{tot}} - x_{z,m}) \tag{3.31}$$

In [56, 57] the total desired ZMP $x_{z,\text{tot}}$ was assigned and $x_{z,M}$ becomes a known function of time which can be interpreted as an equivalent ZMP. The gait generation problem then reduces to finding a bounded solution to (3.30) forced by $x_{z,M}$.

The model that will be used in later chapters employs an additional assumption of constant height of the primary mass z_M. Thus $x_{z,M}$ is related to $x_{z,\text{tot}}$ through the following equation, from (3.27)

$$x_{z,\text{tot}} = \left(1 + \frac{m}{M}\frac{\ddot{z}_m + g}{g}\right)^{-1}\left(x_{z,M} + \frac{m}{M}\frac{\ddot{z}_m + g}{g}x_{z,m}\right). \tag{3.32}$$

Chapter 4

The stability condition

The LIP model (3.9) lets us compute a CoM trajectory that realizes a given ZMP. In the general case however, this CoM trajectory will be divergent, due to the instability of the system. This chapter will discuss conditions for which this instability does not manifest, for the LIP model and then for the alternative models presented in Ch. 3.

4.1 The stability condition for the LIP model

As already noted in Sect. 3.9, the LIP can be decomposed into a stable and an unstable component. The resulting system is (3.12), with x_s and x_u denoting respectively the stable and unstable parts. The state evolution for the decoupled components, starting from time t_0 to a generic time t, is given by

$$\begin{aligned} x_u(t) &= x_u(t_0)e^{\eta(t-t_0)} - \eta \int_{t_0}^{t} e^{\eta(t-\tau)} x_z(\tau)d\tau \\ x_s(t) &= x_s(t_0)e^{-\eta(t-t_0)} + \eta \int_{t_0}^{t} e^{-\eta(t-\tau)} x_z(\tau)d\tau. \end{aligned} \tag{4.1}$$

Although in general the state evolution of x_u for a given ZMP will be divergent, a special initialization of the state can be found which guarantees a bounded evolution (see [58], or [59] for the nonlinear case). This initialization is expressed by

$$x_u^*(t_0) = \eta \int_{t_0}^{\infty} e^{-\eta(\tau-t_0)} x_z(\tau)d\tau \tag{4.2}$$

which we refer to as the *stability condition.*

The stability condition gives guarantees of boundedness of the CoM trajectory with respect to the ZMP. Before analyzing these guarantees, we shall take a look at some useful properties which will simplify the following computations. For compactness, we use the following notation

$$\eta \int_{t_0}^{\infty} e^{-\eta(\tau-t_0)} x_z(\tau)d\tau = x_u^*(t_0; x_z(t)). \tag{4.3}$$

Property 1 *Linearity in $x_z(t)$:*

$$x_u^*(t_0; ax_z^a(t) + bx_z^b(t)) = ax_u^*(t_0; x_z^a(t)) + bx_u^*(t_0; x_z^b(t)).$$

Property 2 *For a step function $x_z(t) = \delta_{-1}(t - t_0)$, we get*

$$x_u^*(t_0; \delta_{-1}(t - t_0)) = 1.$$

Property 3 *For a ramp function $x_z(t) = \rho(t - t_0)$, we get*

$$x_u^*(t_0; \rho(t - t_0)) = 1/\eta.$$

Properties 1-3 are easily derived by explicit computation of the integral in (4.3).

Property 4 *If $x_z(t) = 0$ for $t < t_0$, we get*

$$x_u^*(t_0; x_z(t - T)) = e^{-\eta T} x_u^*(t_0; x_z(t)), \quad T \geq 0.$$

Proof.

$$\begin{aligned} x_u^*(t_0; x_z(t - T)) &= \eta \int_{t_0}^{\infty} e^{-\eta(\tau - t_0)} x_z(\tau - T) d\tau \\ &= \eta \int_{t_0 - T}^{\infty} e^{-\eta(\tau' - t_0 + T)} x_z(\tau') d\tau' \\ &= \eta e^{-\eta T} \int_{t_0 - T}^{\infty} e^{-\eta(\tau' - t_0)} x_z(\tau') d\tau' \\ &= e^{-\eta T} \eta \int_{t_0}^{\infty} e^{-\eta(\tau' - t_0)} x_z(\tau') d\tau' \\ &= e^{-\eta T} x_u^*(t_0; x_z(t)). \qquad \blacksquare \end{aligned}$$

Property (4) (*time shifting)* shows how the stability condition for the time-shifted function $x_z(t - T)$ can be written in terms of the stability condition for the original function $x_z(t)$.

As we mentioned, the stability condition guarantees a bounded CoM evolution if certain conditions are met. This will be described by the following two propositions, the first dealing with bounded ZMP trajectories (the gait is confined to a finite space), the second dealing with ZMP trajectories with bounded derivatives.

Proposition 1 *Assume that the ZMP trajectory is bounded, i.e., $|x_z(t)| \leq C$. Then the CoM trajectory $x_c(t)$ is bounded if and only if the stability condition* (4.2) *is verified.*

Proof. Consider the state evolution at time t of the unstable component, starting from t_0

$$\begin{aligned} x_u(t) &= x_u(t_0) e^{\eta(t - t_0)} - \eta \int_{t_0}^{t} e^{\eta(t - \tau)} x_z(\tau) d\tau \\ &= e^{\eta(t - t_0)} \left(x_u(t_0) - \eta \int_{t_0}^{t} e^{-\eta(\tau - t_0)} x_z(\tau) d\tau \right). \end{aligned}$$

Choosing the initial state (4.2) yields the following state trajectory

$$\begin{aligned} x_u^*(t) &= e^{\eta(t-t_0)}\left(\eta\int_{t_0}^{\infty} e^{-\eta(\tau-t_0)}x_z(\tau)d\tau - \eta\int_{t_0}^{t} e^{-\eta(\tau-t_0)}x_z(\tau)d\tau\right) \\ &= e^{\eta(t-t_0)}\left(\eta\int_{t}^{\infty} e^{-\eta(\tau-t_0)}x_z(\tau)d\tau\right) \\ &= \eta\int_{t}^{\infty} e^{-\eta(\tau-t)}x_z(\tau)d\tau, \end{aligned} \tag{4.4}$$

which, by using the bound on the ZMP and computing the integral, can be bounded as $|x_u^*(t)| \leq C$. The state evolution of the stable component is

$$x_s(t) = x_s(t_0)e^{-\eta(t-t_0)} + \eta\int_{-\infty}^{t} e^{-\eta(t-\tau)}x_z(t)d\tau, \tag{4.5}$$

which by letting $t_0 \to -\infty$ gives

$$x_s(t) = \eta\int_{-\infty}^{t} e^{-\eta(t-\tau)}x_z(t)d\tau. \tag{4.6}$$

This expression can be bounded similarly to $x_u(t)$, and gives

$$|x_s(t)| \leq C. \tag{4.7}$$

Combining the two bounds, and recalling that $x_u(t) + x_s(t) = 2x_c(t)$, proves the sufficiency of (4.1). The necessity is proved by noting that any other initial condition

$$x_u(t_0) = x_u^*(t_0) + \Delta x_u^0, \quad \Delta x_u^0 \neq 0$$

gives a divergent x_u

$$x_u(t) = x_u^*(t) + \Delta x_u^0 e^{\eta(t-t_0)}.$$

Since $x_s(t)$ is still bounded, this proves the proposition. ∎

If the ZMP is not bounded, but its derivative is, the following proposition can be proved.

Proposition 2 *Assume that the ZMP derivative is bounded $|\dot{x}_z(t)| \leq D$. Then the CoM trajectory $x_c(t)$ is bounded with respect to $x_z(t)$ if the stability condition* (4.2) *is satisfied.*

Proof. Rewrite the ZMP trajectory as

$$x_z(t+T) = x_z(t) + \int_{t}^{t+T} \dot{x}_z(\tau)d\tau \tag{4.8}$$

The state trajectory in (4.4) is written as

$$x_u^*(t) = \eta\int_{t}^{\infty} e^{-\eta(\tau-t)}\left(x_z(t) + \int_{t}^{\tau} \dot{x}_z(s)ds\right)d\tau \tag{4.9}$$

which can be bounded as

$$\eta \int_t^{\infty} e^{-\eta(\tau-t)} \left(x_z(t) - (\tau - t)D\right) d\tau \leq x_u^*(t) \leq \eta \int_t^{\infty} e^{-\eta(\tau-t)} \left(x_z(t) + (\tau - t)D\right) d\tau \tag{4.10}$$

using Properties 1, 2 and 3, and bringing $x_z(t)$ to the mid-side leads to

$$-D/\eta \leq x_u^*(t) - x_z(t) \leq D/\eta. \tag{4.11}$$

which bounds the unstable component w.r.t. the ZMP.

The stable component is found to be

$$x_s(t) = \eta \int_{-\infty}^{t} e^{-\eta(t-\tau)} \left(x_z(t) + \int_t^{\tau} \dot{x}_z(s) ds\right) d\tau, \tag{4.12}$$

for which a similar calculation leads to

$$-D/\eta \leq x_s(t) - x_z(t) \leq D/\eta. \tag{4.13}$$

Summing (4.11) and (4.13), and recalling that $x_u(t) + x_s(t) = 2x_c(t)$, leads to

$$|x_c(t) - x_z(t)| \leq D/\eta, \tag{4.14}$$

which proves the proposition. ■

Note that the stability condition defines an initial state that is dependent on future values of the input, and is therefore *non-causal.* Chapter (5) will reformulate this into a causal relationship which will be used to construct an MPC constraint.

4.2 Gait design using the stability condition

The stability condition (4.2) has been employed in the design of humanoid robot gaits [51, 60]. The design of a gait consists of the following steps:

- pre-plan a set of footsteps;
- generate a ZMP trajectory $(x_z(t), y_z(t))^T$ that is inside the support foot during single support phases, and transitions from one foot to the other during double support phases (e.g., a parametric polynomial trajectory);
- use (4.2) to compute the initial condition $(x_u^*(t_0), y_u^*(t_0))^T$ associated with the ZMP trajectory
- pick the initial value of the stable component $(x_s(t_0), y_s(t_0))^T$ to match the actual initial position of the CoM, using the coordinate change (3.11)

  $$x_s(t_0) = 2x_c(t_0) - x_u^*(t_0), \tag{4.15}$$

  and analogously for y;
- using (3.11) again, compute the initial velocity as

  $$\dot{x}_c(t_0) = \eta \frac{x_u^*(t_0) - x_s(t_0)}{2} \tag{4.16}$$

  and analogously for y;

- starting from the initial condition $(x_c(t_0), \dot{x}_c(t_0), y_c(t_0), \dot{y}_c(t_0))^T$ integrate the LIP equation (3.9) using the planned ZMP trajectory $(x_z(t), y_z(t))^T$, to produce a CoM trajectory $\boldsymbol{p}(t)$;
- track the obtained CoM trajectory with a standard kinematic controller.

The above procedure starts from a ZMP trajectory which satisfies the balance condition and computes a suitable CoM trajectory. All trajectories can be expressed in closed form, which makes the process very fast. The starting position is tied to the result of the stability condition 4.2. This requires designing a motion that brings the robot CoM to the proper initial condition, but it makes online replanning harder. Later chapters will show how MPC is able to overcome this limitation by using the stability condition to construct a constraint on future inputs, rather than to compute the initial state.

4.3 Stability conditions for alternative models

This section will give expressions of the stability constraint for the alternative models presented in Ch. 3.

Perturbed LIP model

The perturbed LIP model (3.13) described in Sect. 3.3 also admits a stability condition

$$x_u^*(t_0) = \eta \int_{t_0}^{t} e^{-\eta(\tau - t_0)} x_z(\tau) d\tau + \frac{1}{\eta^2} \int_{t_0}^{t} e^{-\eta(\tau - t_0)} w(\tau) w\tau. \tag{4.17}$$

Here the non-causality of the condition is manifested not only in the dependence on future values of the input x_z, but also in the knowledge of the future value of the disturbance. This means that the closed form expressions of Sect. 4.1 can only be obtained if the disturbance is known a priori. This has some applicability, for example on a robot walking on a known slope. In a reference frame parallel with the slope the horizontal component of gravity can be considered as a disturbance and thus computed exactly. Otherwise, unknown disturbances will be taken into account in the MPC, thanks to its constant replanning.

Linear 3D model

The linear 3D model was described in Sect. 3.4. Here the stability condition is given by a set of three equations. The expressions along x and y are equivalent to (4.2), as the model (3.21) is identical to the LIP, but the vertical equation differs in the presence of an additive term

$$z_u^*(t_0) = \eta \int_{t_0}^{t} e^{-\eta(\tau - t_0)} z_z(\tau) d\tau + \frac{g}{\eta^2} \tag{4.18}$$

Recall that in this model the ZMP is allowed to leave the ground and the balance condition is modified accordingly. If the ZMP is fixed on the ground, from this expression we recover $z_u^*(t_0) = g/\eta^2$. In this case whatever initial CoM height we choose will stay constant, and will determine the value of $\eta = \sqrt{g/z_u^*(t_0)}$, which is equivalent to enforcing Assumption 3 as we did for the standard LIP model.

Two-mass model

The two-mass model derived in Sect. 3.5 is expressed as an equivalent LIP. This makes it straightforward to derive the stability condition, as it has the same expression of (4.2) but uses the variables of the equivalent LIP (3.5)

$$x_{u,M}^*(t_0) = \eta \int_{t_0}^{t} e^{-\eta(\tau - t_0)} x_{z,M}(\tau) d\tau \tag{4.19}$$

where $x_{z,M}(t)$ is defined in (3.25).

Chapter 5

IS-MPC: the basic scheme

This chapter will describe the MPC scheme which constitutes the main contribution of this thesis. This scheme features an explicit *stability constraint* which is in charge of guaranteeing non-divergence of the CoM trajectory with respect to the ZMP (internal stability). For this reason we adopted the name Intrinsically Stable MPC (IS-MPC). This chapter will specifically be about the scheme in its basic form [3, 4], while further extensions or variations will be presented in later chapters.

An overview of the scheme is given in Fig. 5.1. The goal is to follow a set of high-level velocities, given as the driving (v_x, v_y) and steering ω velocities of an omnidirectional single-body mobile robot chosen as a template model for motion generation. These velocities come from an external source, e.g., a human operator or an external controller.

At the current time t_k, it is assumed that the input reference velocities are available over a horizon T_p called *preview horizon*. A module is in charge of planning the footstep sequence over this preview horizon. The footstep sequence includes timing, position and orientations of the future F footsteps.

This footstep sequence is sent to the IS-MPC stage which will produce a ZMP and CoM trajectory. The IS-MPC stage works over a shorter horizon than the footstep planner, called the *control horizon* T_c. It should be noted that in the basic version of this scheme the MPC stage works with *given footsteps*, which means that it is not allowed to modify the footstep plan. This should not be confused with having knowledge of the entire future footstep plan, as it is only planned up to the preview horizon.

A separate module generates a trajectory for the swing foot. It receives information from the footstep plan, and uses parametric polynomials to generate a trajectory going from the starting pose of the swing foot to the pose of the first predicted footstep. The CoM and swing foot trajectories are kinematically tracked by the robot using a classic *stack of tasks* technique.

5.1 Footstep generation

The footstep generator runs synchronously with the IS-MPC scheme. It is in charge of determining the timing as well as the position and orientation of the footsteps, so to realize as closely as possible the high-level reference velocities. The timing is

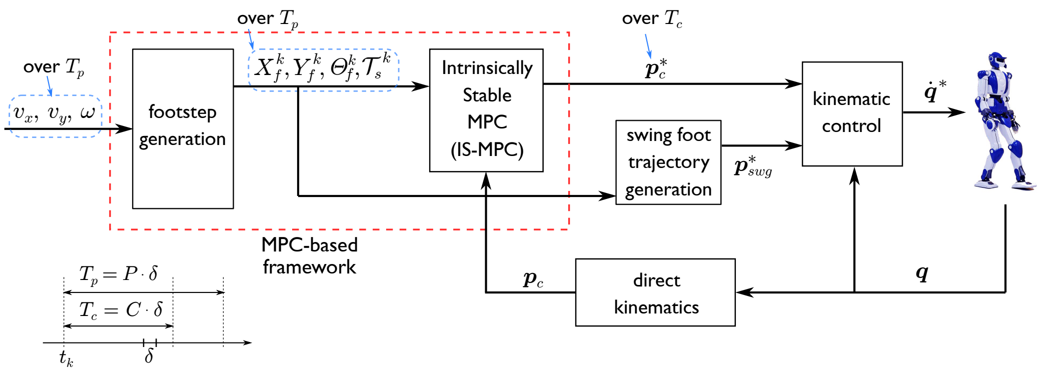


Figure 5.1. The basic version of IS-MPC.

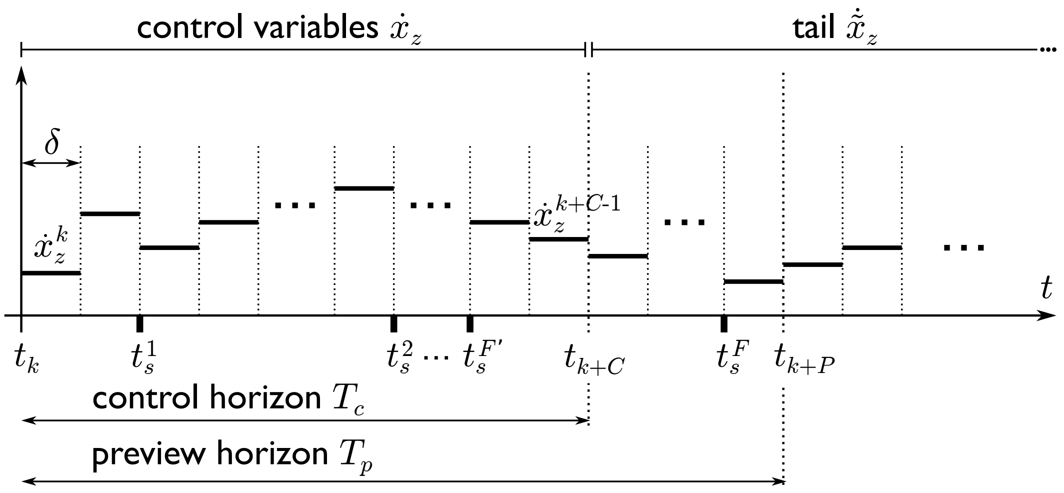


Figure 5.2. At time t_k, the control variables determined by IS-MPC are the piecewise-constant ZMP velocities over the control horizon. The ZMP velocities after the control horizon are instead conjectured in order to build the tail (see Sect. 5.4.1). Also shown are the F footstep timestamps placed by the footstep generation module in the preview horizon; F' of them fall in the control horizon.

determined using a simple rule expressing the fact that a change in the reference velocity should affect both the step duration and length. The footstep locations and orientations are then chosen through quadratic optimization.

At time t_k the reference velocities are provided over the preview horizon, i.e. from t_k to $t_k + T_p = t_{k+P}$ (see Fig. 5.2). The output of the planner is the footstep sequence $(X_f^k, Y_f^k, \Theta_f^k)$ over the same interval with the associated timing $\mathcal{T}_s^k$. In particular, these quantities are defined[1] as

$$\begin{aligned} X_f^k &= (x_f^1 \; \dots \; x_f^F)^T \\ Y_f^k &= (y_f^1 \; \dots \; y_f^F)^T \\ \Theta_f^k &= (\theta_f^1 \; \dots \; \theta_f^F)^T \end{aligned}$$

and

$$\mathcal{T}_s^k = \{T_s^1, \dots, T_s^F\},$$

where $(x_f^j, y_f^j, \theta_f^j)$ is the pose of the j-th footstep in the preview horizon and T_s^j is the duration of the step between the $(j-1)$-th and the j-th footstep, taken from the start of the single support phase to the next. F is the number of footsteps falling within the preview horizon, which may change as the footstep timing itself is variable. Note that these footstep poses and timing alternatively identify left and right footsteps.

5.1.1 Footstep timing

As a rule for determining the duration of each step, we employ the magnitude $v = (v_x^2 + v_y^2)^{1/2}$ of the reference Cartesian velocity at the beginning of that step.

Assume that a triplet of *cruise parameters* $(\bar{v}, \overline{T}_s, \bar{L}_s)$ has been chosen, where $\bar{v}$ is a central value of v and $\overline{T}_s$, $\bar{L}_s$ are the corresponding values of the step duration and length, respectively, with $\bar{v} = \bar{L}_s/\overline{T}_s$. These values clearly should be chosen respecting the capabilities of the robot.

The idea is that a deviation from $\bar{v}$ should reflect on a change in *both* T_s and L_s. This can be expressed as

$$v = \bar{v} + \Delta v = \frac{\bar{L}_s + \Delta L_s}{\overline{T}_s - \Delta T_s},$$

with $\Delta L_s = \alpha \Delta T_s$. One easily obtains

$$T_s = \overline{T}_s \, \frac{\alpha + \bar{v}}{\alpha + v}. \tag{5.1}$$

The resulting rule can be used for determining T_s as a function of v. The function is shown in Fig. 5.3 and it is compared to other possible simpler rules. For illustration, we have set $\bar{v} = 0.15$ m/s, $\overline{T}_s = 0.8$ s, $\bar{L}_s = 0.12$ m and $\alpha = 0.1$ m/s. It can easily be verified that an increase in v, for example, corresponds to both a decrease of T_s and an increase in L_s.

[1]To keep a light notation, the k symbol identifying the current sampling instant is used for the sequence vectors but not for their individual elements.

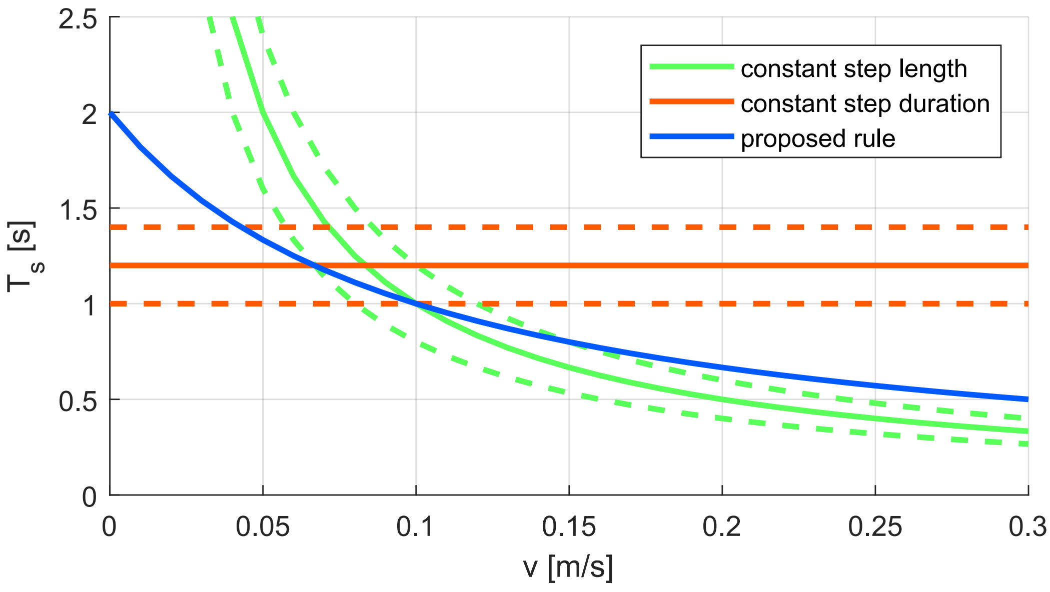


Figure 5.3. The proposed rule for determining the step duration T_s as a function of the magnitude v of the reference Cartesian velocity. For comparison, also shown are the rules yielding constant step duration and constant step length.

The reference angular velocity ω does not influence the rule (5.1). In fact, step length and timing do not differ significantly if the Cartesian velocity v is the same. This also means that if the robot is required to rotate in place, it will do so with the maximum allowed T_s.

Rule (5.1) is employed by iterating along the preview horizon $[t_k, t_k + T_p]$ in order to obtain the footstep timestamps:

$$t_s^j = t_s^{j-1} + \overline{T}_s \, \frac{\alpha + \bar{v}}{\alpha + v(t_s^{j-1})},$$

with t_s^0 equal to the timestamp of the last footstep before t_k. As soon as the end of the preview is reached, i.e., $t_s^j > t_{k+P}$, the iterations are stopped and the last timestamp is discarded as it is outside the preview horizon. The resulting step timings are collected as $\mathcal{T}_s^k = \{T_s^1, \dots, T_s^F\}$, with $T_s^j = t_s^{j+1} - t_s^j$.

5.1.2 Footstep placement

The footstep poses are chosen after their timing has been determined over the preview horizon. The idea is that the footsteps should follow the trajectory obtained by integrating the following template model under the action of the high-level reference velocities over T_p:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} v_x \\ v_y \\ \omega \end{pmatrix}. \tag{5.2}$$

This omnidirectional motion model can be considered as an extension of a unicycle, which is often employed as a template model for motion generation. Compared

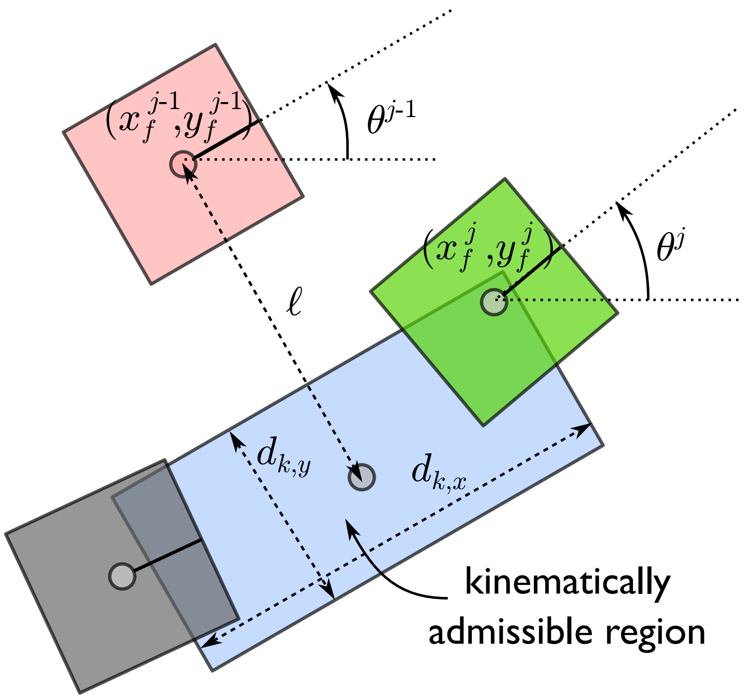


Figure 5.4. The kinematic constraint on footstep placement.

to the unicycle, it can also generate lateral motions, which the humanoid robot is allowed to perform.

The footstep plan should be distributed along the trajectory traced by the template model, but it should also respect the robot kinematic limits. For this reason, a *kinematic constraint* will be enforced on the position of the footsteps. Consider the j-th step in T_c, with the support foot centered at (x_f^{j-1}, y_f^{j-1}) and oriented as θ^{j-1}. The kinematically admissible region for placing the footstep is defined as a rectangle having the same orientation θ^{j-1} and whose center is displaced from the support foot center by a distance ℓ in the coronal direction (see Fig. 5.4). Denoting by $d_{k,x}$ and $d_{k,y}$ the dimensions of the kinematically admissible region, the constraint is written as

$$\pm\begin{pmatrix} 0 \\ \ell \end{pmatrix} - \frac{1}{2}\begin{pmatrix} d_{k,x} \\ d_{k,y} \end{pmatrix} \leq R_{j-1}^T \begin{pmatrix} x_f^j - x_f^{j-1} \\ y_f^j - y_f^{j-1} \end{pmatrix} \leq \pm\begin{pmatrix} 0 \\ \ell \end{pmatrix} + \frac{1}{2}\begin{pmatrix} d_{k,x} \\ d_{k,y} \end{pmatrix}, \qquad (5.3)$$

with the sign alternating for the two feet. This constraint should be satisfied by all footsteps in the preview horizon ($j = 1, \ldots, F$).

The footsteps positions and orientations are determined by a sequence of two QP problems. These need to be solved sequentially, as the second uses the orientations resulting from the first to construct the rotation matrices R_{j-1}^T. The first is

$$\left\{ \begin{array}{c} \displaystyle\min_{\Theta_f^k} \sum_{j=1}^{F} \left(\theta_f^j - \theta_f^{j-1} - \int_{t_s^{j-1}}^{t_s^j} \omega(\tau) d\tau \right)^2 \\ \text{subject to} \quad |\theta_f^j - \theta_f^{j-1}| \leq \theta_{\max} \end{array} \right.$$

Here, $\theta_{\max}$ is the maximum allowed rotation between two consecutive footsteps. The second QP problem is

$$\begin{cases} \displaystyle\min_{X_f^k, Y_f^k} \sum_{j=1}^{F} (x_f^j - x_f^{j-1} - \Delta x^j)^2 + (y_f^j - y_f^{j-1} - \Delta y^j)^2 \\ \text{subject to kinematic constraints (5.3)} \end{cases}$$

Here, (x_f^0, y_f^0) is the known position of the support foot at t_k and Δx^j, Δy^j are given by

$$\begin{pmatrix} \Delta x^j \\ \Delta y^j \end{pmatrix} = \int_{t_s^{j-1}}^{t_s^j} R_\theta \begin{pmatrix} v_x(\tau) \\ v_y(\tau) \end{pmatrix} d\tau \pm R_j \begin{pmatrix} 0 \\ \ell/2 \end{pmatrix},$$

where R_θ, R_j are the rotation matrices associated respectively to $\theta(\tau)$ (the orientation of the template robot at any given time τ) and to the footstep orientation θ_j, and ℓ is the reference coronal distance between consecutive footsteps. The sign of the second term alternates for left/right footsteps.

The footstep timings, positions and orientations are then sent to the IS-MPC stage. Recall that in the basic version of IS-MPC which is presented in this chapter, the MPC stage is not allowed to change the footstep positions, but the extension presented in the next chapter will be able to perform automatic footstep placement for improved robustness to perturbations.

Some examples of footstep generation are shown in Fig. 5.5.

5.2 Prediction model

The IS-MPC stage performs a prediction over the control horizon T_c, to determine ZMP and CoM trajectories that are compatible with the chosen footstep plan. For this it uses the LIP as a prediction model (see Sect. 3.2). In order to produce smoother trajectories, a *dynamic extension* is employed, in which the input is the ZMP derivative $\dot{x}_z$, instead of the ZMP itself. The dynamically extended LIP can be written as

$$\begin{pmatrix} \dot{x}_c \\ \ddot{x}_c \\ \dot{x}_z \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ \eta^2 & 0 & -\eta^2 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_c \\ \dot{x}_c \\ x_z \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \dot{x}_z \tag{5.4}$$

for the x component, with the y component being analogous and decoupled.

The scheme operates in a digital fashion, with time-steps of duration δ. The input $\dot{x}_z$ is assumed to be constant over the discrete time-steps (see Fig. 5.2)

$$\dot{x}_z(t) = \dot{x}_z^i, \quad t \in [t_i, t_{i+1}), \tag{5.5}$$

which makes the ZMP *piecewise linear* over the time-steps

$$x_z(t) = x_z^i + \dot{x}_z^i (t - t_i), \quad t \in [t_i, t_{i+1}), \tag{5.6}$$

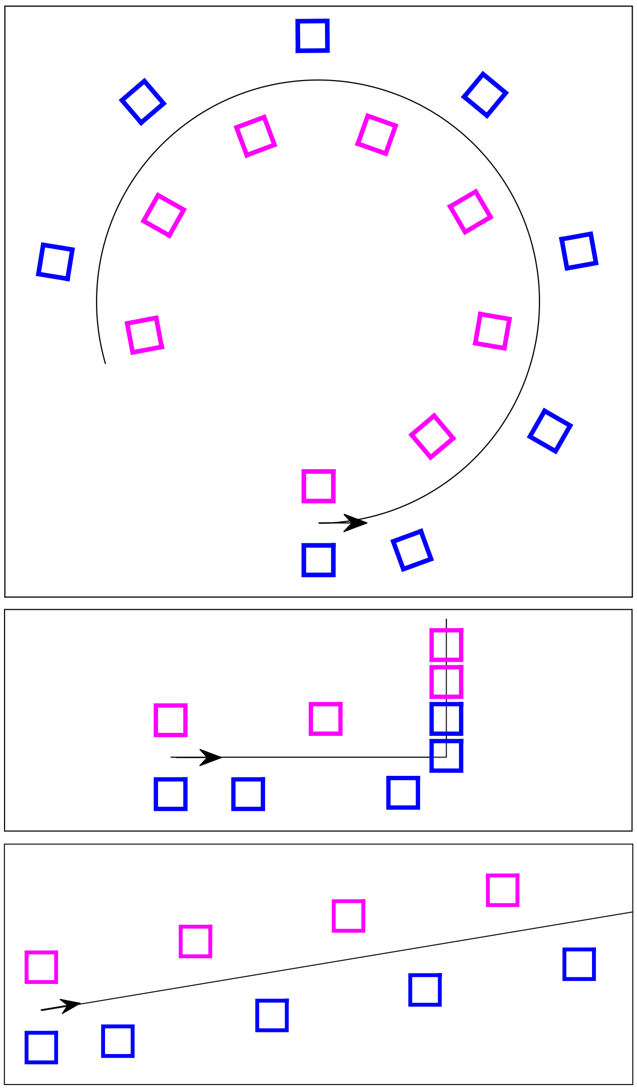


Figure 5.5. Footsteps generated by the proposed method for different high-level reference velocities corresponding to a circular walk (top), L-walk (center), diagonal walk (bottom). The paths in black are obtained by integrating model (5.2) under the reference velocities. Footstep in magenta and cyan refer respectively to the left and right foot.

The discrete time dynamic model is computed by integrating (5.4) over the time-step δ with a constant input $\dot{x}_z$ (see Appendix A for a complete derivation)

$$\begin{pmatrix} x_c^{k+1} \\ \dot{x}_c^{k+1} \\ x_z^{k+1} \end{pmatrix} = \begin{pmatrix} \cosh(\eta\delta) & \sinh(\eta\delta)/\eta & 1-\cosh(\eta\delta) \\ \eta\sinh(\eta\delta) & \cosh(\eta\delta) & -\eta\sinh(\eta\delta) \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_c^k \\ \dot{x}_c^k \\ x_z^k \end{pmatrix} + \begin{pmatrix} \delta - \sinh(\eta\delta)/\eta \\ 1-\cosh(\eta\delta) \\ \delta \end{pmatrix} \dot{x}_z^k. \tag{5.7}$$

Note that the discrete model is computed by exact integration of the continuous model over a constant input, with no approximation required. This ensures that the ZMP constraints (described in the next section) enforced at the discrete sampled timings, will also be satisfied in between the samples.

Recall that the IS-MPC stage works over the control horizon, which is in general *shorter* than the preview horizon of the footstep generation module. The number of samples in the preview and control horizon is denoted as P and C respectively. The predicted state and input variables are notated with a superscript, where k denotes the current instant corresponding to time t_k, and $k+i$ denotes the i-th predicted instant corresponding to time t_{k+i}. It is useful to collect them in vectors, for the input

$$\begin{aligned} \dot{X}_z^k &= \left(\dot{x}_z^k, \dot{x}_z^{k+1}, \dots, \dot{x}_z^{k+C-1}\right)^T \\ \dot{Y}_z^k &= \left(\dot{y}_z^k, \dot{y}_z^{k+1}, \dots, \dot{y}_z^{k+C-1}\right)^T \end{aligned} \tag{5.8}$$

and the state

$$\begin{aligned} X_c^{k+1} &= \left(x_c^{k+1}, x_c^{k+2}, \dots, x_c^{k+C}\right)^T \\ Y_c^{k+1} &= \left(y_c^{k+1}, y_c^{k+2}, \dots, y_c^{k+C}\right)^T \\ \dot{X}_c^{k+1} &= \left(\dot{x}_c^{k+1}, \dot{x}_c^{k+2}, \dots, \dot{x}_c^{k+C}\right)^T \\ \dot{Y}_c^{k+1} &= \left(\dot{y}_c^{k+1}, \dot{y}_c^{k+2}, \dots, \dot{y}_c^{k+C}\right)^T \\ X_z^{k+1} &= \left(x_z^{k+1}, x_z^{k+2}, \dots, x_z^{k+C}\right)^T \\ Y_z^{k+1} &= \left(y_z^{k+1}, y_z^{k+2}, \dots, y_z^{k+C}\right)^T. \end{aligned} \tag{5.9}$$

5.3 ZMP constraints

The ZMP constraint ensures dynamic balance of the humanoid. As mentioned earlier, there are two distinct phases during locomotion. The single-support phase is active when the robot has a single foot in contact with the ground, while during the double-support phase it has two. The ZMP constraint is expressed slightly differently in these two phases, which will be discussed separately.

Single-support ZMP constraints

The ZMP constraint during single-support is enforced by guaranteeing that the ZMP is in a convex region that is contained within the contact surface of the support foot with the ground. This is for simplicity approximated by a rectangle, but any convex polytope could in principle be used, as it can be written as a linear constraint.

$$-\frac{1}{2}\begin{pmatrix} d_{x,z} \\ d_{y,z} \end{pmatrix} \leq Rj^T \begin{pmatrix} x_z^{k+i} - x_f^j \\ y_z^{k+i} - y_f^j \end{pmatrix} \leq \frac{1}{2}\begin{pmatrix} d_{x,z} \\ d_{y,z} \end{pmatrix} \tag{5.10}$$

(x_f^j, y_f^j) is the position of the j-th support foot in the control horizon, and R_j is the rotation matrix associated with its orientation. Note that $j = 0, \ldots, F'$, where (x_f^0, y_f^0) represents the position of the current support foot. (x_z^{k+i}, y_z^{k+i}) is the predicted position of the ZMP at the i-th time instant into the future. The latter is easily expressed linearly in the decision variables as

$$\begin{pmatrix} x_z^{k+i} \\ y_z^{k+i} \end{pmatrix} = \begin{pmatrix} x_z^k \\ y_z^k \end{pmatrix} + \delta \sum_{j=k}^{i-1} \begin{pmatrix} \dot{x}_z^j \\ \dot{y}_z^j \end{pmatrix}. \tag{5.11}$$

This constraint is imposed for all i for which t_{k+i} is in a single-support phase.

Double-support ZMP constraint

The support polygon during double-support includes the area of the feet in contact with the ground, as well as the area in between them. If the footstep sequence is given, the whole area could be expressed as a linear constraint, but this would not extend well to the case of automatic footstep placement (see next chapter), and its complex expression would make it more difficult to prove stability and feasibility properties. For these reasons, during double support we enforce a *moving constraint*, which is an approximation that is always contained in the actual support polygon. We employed this in [5], but a similar idea was utilized in [61]. This constraint has the shape and size of the constraint during single support, but it gradually shifts from one footstep to the next during double support.

Consider the construction shown in Fig. 5.6, which considers the case of a single predicted footstep (i.e., $F' = 1$) for illustration. The support polygon during double support is outlined in yellow. The ZMP should move from the square in the bottom left to the one in the top right. Suppose that this transition takes D sampling intervals ($D = 3$ in the figure). This leads to define D equispaced support squares. During the n-th time-step ($n = 1, \ldots, 3$), the support square n is activated. By doing so, the ZMP is always contained inside the original double support polygon.

According to the above discussion, the ZMP constraint during double support can be defined by

$$-\frac{1}{2}\begin{pmatrix} d_{x,z} \\ d_{y,z} \end{pmatrix} \leq R_j^T \begin{pmatrix} x_z^{k+i} - \frac{n}{D+1} x_\mathrm{f}^{j-1} - \frac{D+1-n}{D+1} x_\mathrm{f}^j \\ y_z^{k+i} - \frac{n}{D+1} y_\mathrm{f}^{j-1} - \frac{D+1-n}{D+1} y_\mathrm{f}^j \end{pmatrix} \leq \frac{1}{2}\begin{pmatrix} d_{x,z} \\ d_{y,z} \end{pmatrix} \tag{5.12}$$

with $n = 1, \ldots, D$.

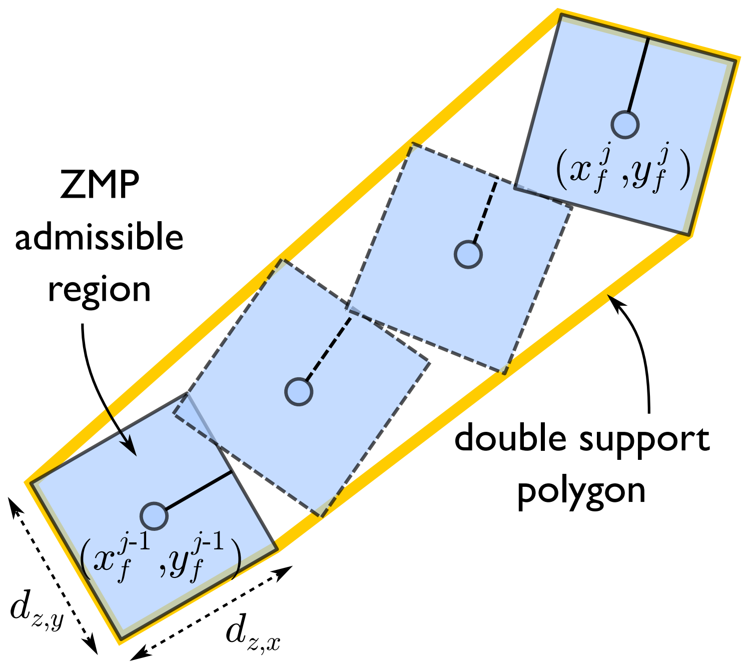


Figure 5.6. Redefining the ZMP constraint during double support.

5.4 The stability constraint

The role of the stability constraint is to ensure internal stability of the scheme. In the standard MPC practice a similar role is entrusted to a *terminal constraint*, which is a constraint on the state at the last instant of the prediction. In fact, for standard control problems such as regulation and tracking, terminal constraints can be found which are able to guarantee feasibility and stability of the scheme.

The case under consideration is different, as the goal is neither regulation nor tracking, but rather satisfying time varying constraints. For this reason a standard terminal constraint does not seem to be effective. The stability constraint fulfills the role of a terminal constraint in the IS-MPC scheme and, as we will discuss later, can be reformulated as a condition on the state at the end of the prediction.

To compute the stability constraint we utilize the stability condition (4.2) for the LIP system. The reason for this is that we are interested in guaranteeing boundedness of the CoM w.r.t. the ZMP, thus we can ignore the dynamic extension.

The stability condition (4.2) relates the initial condition to the future of the ZMP trajectory. However, since we will be enforcing this at every time-step, the initial time is the current time t_k and $x(t_0)$ is replaced by x_u^k

$$x_u^k = \eta \int_{t_k}^{\infty} e^{-\eta(\tau - t_k)} x_z(\tau) d\tau. \tag{5.13}$$

The stability condition, which involves x_u at the initial instant t_k of the control horizon, can be propagated to its final instant t_{k+C} by integrating (3.9) from x_u^k in (4.2):

$$x_u^{k+C} = \eta \int_{t_{k+C}}^{\infty} e^{-\eta(\tau - t_{k+C})} x_z(\tau) d\tau. \tag{5.14}$$

Condition (5.13) — or equivalently, (5.14) — can be used to set up the corresponding constraint for the MPC problem. To this end, we use the piecewise-linear profile (5.6) of x_z to obtain explicit forms.

Proposition 3 *For the piecewise-linear x_z in (5.6), condition (4.2) becomes*

$$x_u^k = x_z^k + \frac{1-e^{-\eta\delta}}{\eta}\sum_{i=0}^{\infty} e^{-i\eta\delta}\dot{x}_z^{k+i}, \tag{5.15}$$

while (5.14) takes the form

$$x_u^{k+C} = x_z^{k+C} + \frac{1-e^{-\eta\delta}}{\eta} e^{C\eta\delta}\sum_{i=C}^{\infty} e^{-i\eta\delta}\dot{x}_z^{k+i}. \tag{5.16}$$

Proof. The piecewise-linear ZMP can be directly plugged into the integral, but the computation is quite lengthy. For a much shorted derivation, rewrite eq. (5.6) as

$$x_z(t) = x_z^k + \sum_{i=0}^{\infty}(\rho(t-t_{k+i}) - \rho(t-t_{k+i+1}))\dot{x}_z^{k+i}, \tag{5.17}$$

where $\rho(t) = t\,\delta_{-1}(t)$ denotes the unit ramp and $\delta_{-1}(t)$ the unit step. Using Properties 1, 4 and 3 given in Sect. 4.2, we get

$$\int_{t_k}^{\infty} e^{-\eta(\tau-t_k)}(\rho(\tau-t_{k+i}) - \rho(\tau-t_{k+i+1}))d\tau = \frac{1-e^{-\eta\delta}}{\eta^2}e^{-i\eta\delta}.$$

Plugging this expression in condition (4.2) and using Property 2 of Sect. 4.2 one obtains (5.15).

To prove (5.16), rewrite (5.17) as

$$\begin{aligned} x_z(t) = x_z^k &+ \sum_{i=0}^{C-1}(\rho(t-t_{k+i}) - \rho(t-t_{k+i+1}))\dot{x}_z^{k+i} \\ &+ \sum_{i=C}^{\infty}(\rho(t-t_{k+i}) - \rho(t-t_{k+i+1}))\dot{x}_z^{k+i}. \end{aligned}$$

The contribution of the first two terms of x_z to the integral in (5.14) is x_z^{k+C}. Using Properties 1, 3 and 4 one verifies that the contribution of the third term is exactly the second term in the right hand side of (5.16). This completes the proof. ■

In (5.15), we should logically separate the values of $\dot{x}_z^i$ within the control horizon, i.e. the control variables $\dot{x}_z^i$ for $i = k, \dots, k+C-1$, from the remaining values, i.e., from $k+C$ on. The infinite summation is then split in two parts and (5.15) can be rearranged as[2]

$$\sum_{i=0}^{C-1} e^{-i\eta\delta}\dot{x}_z^{k+i} = -\sum_{i=C}^{\infty} e^{-i\eta\delta}\dot{x}_z^{k+i} + \frac{\eta}{1-e^{-\eta\delta}}(x_u^k - x_z^k). \tag{5.18}$$

[2]Constraint (5.18) can be written as a function of the actual state variables of our prediction model (x_c, $\dot{x}_c$ and x_z) using the coordinate transformation (3.11). The same is true for all subsequent forms of the stability constraint as well as of the terminal constraint.

Observe the inversion between (5.15), which expresses the stable initialization at t_k for a given $x_z(t)$, and (5.18), which constrains the control variables so that the associated stable initialization matches the current state at t_k. Therefore, we will refer to (5.18) as the *stability constraint.*

The control variables do not appear in condition (5.16), which involves only the value of the state variable x_u^{k+C} at the end of the control horizon. In other terms, this condition represents a terminal constraint.

Both the stability and the terminal constraint contain an infinite summation which depends on $\dot{x}_z^{k+C}, \dot{x}_z^{k+C+1}, \ldots$, i.e., the ZMP velocities *after* the control horizon. These are obviously unknown, because they will be determined by future iterations of the MPC algorithm; as a consequence, including either of the constraints in the MPC formulation would lead to a non-causal (unrealizable) controller. However, by exploiting the preview information on v_x, v_y, ω, we can make an *informed conjecture* at t_k about these ZMP velocities, which we will denote by $\dot{\tilde{x}}_z^{k+C}, \dot{\tilde{x}}_z^{k+C+1}, \ldots$ (see Fig. 5.2) and refer to collectively as the *tail* in the following. Correspondingly, the stability constraint (5.18) assumes the form

$$\sum_{i=0}^{C-1} e^{-i\eta\delta}\dot{x}_z^{k+i} = -\sum_{i=C}^{\infty} e^{-i\eta\delta}\dot{\tilde{x}}_z^{k+i} + \frac{\eta}{1-e^{-\eta\delta}}(x_u^k - x_z^k), \tag{5.19}$$

while the terminal constraint (5.16) becomes

$$x_u^{k+C} = x_z^{k+C} + \frac{1-e^{-\eta\delta}}{\eta} e^{C\eta\delta} \sum_{i=C}^{\infty} e^{-i\eta\delta}\dot{\tilde{x}}_z^{k+i}. \tag{5.20}$$

Using either of these in the MPC formulation will lead to a causal (realizable) controller.

5.4.1 The velocity tail

The ZMP velocities after the control horizon, which we have defined as tail in (5.18) and (5.16), can be conjectured in a number of different ways. These can be more or less appropriate in different circumstances, and have different effects on the recursive feasibility of the scheme.

We will consider three possibilities for the tail, which we denote as *truncated tail*, *periodic tail* and *anticipative tail.*

Truncated tail

The truncated tail corresponds to setting $\dot{x}_z^i = 0$ for $t > t_{k+C}$. The second summation is then simply zero and the stability constraint assumes the form

$$\sum_{i=0}^{C-1} e^{-i\eta\delta}\dot{x}_z^{k+i} = \frac{\eta}{1-e^{-\eta\delta}}(x_u^k - x_z^k). \tag{5.21}$$

This is the simplest possible choice for the velocity tail. It is clearly the most appropriate when the robot is about to stop.

By setting $\dot{x}_z^{k+i}$ for $i \geq C$ in (5.16) it is possible to see that the stability constraint using the truncated tail (5.21) is equivalent, expressed as a terminal constraint, to

$$x_u^{k+C} = x_z^{k+C}, \tag{5.22}$$

known as the *capturability constraint* [17].

Periodic tail

If the footstep sequence is regular enough, the generated gait will usually exhibit some kind of periodicity. The periodic tail exploits this periodicity to conjecture the velocities of the ZMP beyond the MPC control horizon.

Consider the stability constraint, and assume for simplicity that the gait period is equal to the control horizon. Split the infinite summation in groups of C terms each

$$\sum_{i=0}^{C-1} e^{-i\eta\delta} \dot{x}_z^{k+i} + \sum_{i=C}^{2C-1} e^{-i\eta\delta} \dot{x}_z^{k+i} + \sum_{i=2C}^{3C-1} e^{-i\eta\delta} \dot{x}_z^{k+i} + \cdots = \frac{\eta}{1-e^{-\eta\delta}}(x_u^k - x_z^k). \tag{5.23}$$

and with a simple change of indices

$$\sum_{i=0}^{C-1} e^{-i\eta\delta} \dot{x}_z^{k+i} + \sum_{i=0}^{C-1} e^{-(i+C)\eta\delta} \dot{x}_z^{k+i+C} + \sum_{i=0}^{C-1} e^{-(i+2C)\eta\delta} \dot{x}_z^{k+i+2C} + \cdots = \frac{\eta}{1-e^{-\eta\delta}}(x_u^k - x_z^k). \tag{5.24}$$

By assumption $x_z^{k+i} = x_z^{k+i+C}$, which simplifies to

$$\sum_{j=0}^{\infty} e^{-j\eta\delta C} \sum_{i=0}^{C-1} e^{-i\eta\delta} \dot{x}_z^{k+i} = \frac{\eta}{1-e^{-\eta\delta}}(x_u^k - x_z^k). \tag{5.25}$$

The first sum is a simple geometric series which converges to $1/(1-e^{-\eta\delta C})$

$$\sum_{i=0}^{C-1} e^{-i\eta\delta} \dot{x}_z^{k+i} = \eta \frac{1-e^{-\eta\delta C}}{1-e^{-\eta\delta}}(x_u^k - x_z^k). \tag{5.26}$$

It can be shown that the equivalent terminal constraint form when using the periodic tail becomes

$$\dot{x}_u^{t_{k+C}} = \dot{x}_u^{t_k}. \tag{5.27}$$

Anticipative tail

The anticipative tail uses the given footstep plan beyond the control horizon to approximate the ZMP trajectory up to the preview horizon, which we referred to as an informed conjecture. In principle any ZMP trajectory that is compatible with the given footstep plan can be used. In practice it is more convenient to use the trajectory generated by the midpoint between the ZMP constraints.

Whatever trajectory is chosen, the velocity sample obtained are denoted by $\dot{x}_{z,\text{ant}}^{k+i}$, for $i = C, \dots, P-1$. The *anticipative tail* is then obtained by:

- setting $\tilde{\dot{x}}_z^{k+i} = \dot{x}_{z,\text{ant}}^{k+i}$ for $i = C, \dots, P-1$;

- using a truncated or periodic expression for the residual part of the tail located *after* the preview horizon, i.e., for $\dot{\tilde{x}}_z^{k+i}$, $i = P, P+1, \dots$.

The stability constraint (5.19) can be written as

$$\sum_{i=0}^{C-1} e^{-i\eta\delta} \dot{x}_z^{k+i} = -\sum_{i=C}^{P-1} e^{-i\eta\delta} \dot{x}_{z,\mathrm{ant}}^{k+i} - \sum_{i=P}^{\infty} e^{-i\eta\delta} \dot{\tilde{x}}_z^{k+i} + \frac{\eta}{1-e^{-\eta\delta}}(x_u^k - x_z^k).$$

which is a simple sum of closed form expressions and does not constitute an additional load for the real-time implementation.

5.5 The IS-MPC algorithm

The generic step of the algorithm involves solving the following QP problem

$$\left\{ \begin{array}{l} \displaystyle\min_{\dot{X}_z^k, \dot{Y}_z^k} \; \|\dot{X}_z^k\|^2 + \|\dot{Y}_z^k\|^2 \\ \text{subject to:} \end{array} \right.$$

- ZMP constraints (5.10) and (5.12)
- stability constraints (5.18) for x and y

The generic iteration of the IS-MPC algorithm goes as follows. The input data are the sequence $(X_f^k, Y_f^k, \Theta_f^k)$ of planned footsteps, with the associated timing $\mathcal{T}_s^k$. As initialization, one needs x_c, $\dot{x}_c$ and x_z at the current sampling instant t_k. One may either use measured data or the current model prediction, depending on the available sensors.

The IS-MPC iteration at t_k goes as follows.

1. Solve the QP problem to obtain $\dot{X}_z^k, \dot{Y}_z^k$.
2. From the solutions, extract $\dot{x}_z^k$, $\dot{y}_z^k$, the first control samples.
3. Set $\dot{x}_z = \dot{x}_z^k$ in (5.4) and integrate from $(x_c^k, \dot{x}_c^k, x_z^k)$ to obtain $x_c(t)$, $\dot{x}_c(t)$, $x_z(t)$ for $t \in [t_k, t_{k+1}]$. Compute $y_c(t)$, $\dot{y}_c(t)$, $y_z(t)$ similarly.
4. Define the 3D trajectory of the CoM as $\boldsymbol{p}_c^* = (x_c, y_c, \bar{z}_c)$ in $[t_k, t_{k+1}]$ and return it.

The swing foot trajectory $\boldsymbol{p}_{swg}^*$ is computed using the planned footsteps. Both the CoM trajectory $\boldsymbol{p}_c^*$ and the swing foot trajectory $\boldsymbol{p}_{swg}^*$ are sent to the kinematic controller (see Fig. 5.1).

5.6 Stability and feasibility

This section will discuss some properties of the IS-MPC scheme. Recall that the footstep sequence is given up to the preview horizon T_p, and consider the following simplifying assumption

Assumption 4 *All footsteps have the same orientation $\theta^j = 0$.*

This assumption is useful in simplifying the computations, as it lets us decouple the x and y components of the MPC constraints. It is possible to remove the assumption by employing an appropriate coordinate change.

In the following we will introduce the concept of *feasibility region*, a region of the state space in which the problem is feasible at time t_k. Feasibility regions will be employed to prove recursive feasibility using the anticipative tail, which will lead to a proof of internal stability.

5.6.1 Feasibility regions

Consider the k-th step of the IS-MPC algorithm.

Definition 1 *The QP problem is feasible at t_k if there exists a ZMP trajectory $x_z(t)$ that for $t \in [t_k, t_{k+C}]$ satisfies both the ZMP constraint*

$$x_z^m(t) \leq x_z(t) \leq x_z^M(t), \tag{5.28}$$

and the stability constraint

$$\eta \int_{t_k}^{t_{k+C}} e^{-\eta(\tau - t_k)} x_z(\tau) d\tau = x_u^k - \eta \int_{t_{k+C}}^{\infty} e^{-\eta(\tau - t_k)} \tilde{x}_z(\tau) d\tau. \tag{5.29}$$

Note the following:

- $x_z^m(t)$ and $x_z^M(t)$ are respectively the lower and upper bound of the ZMP admissible region at time t, as derived from (8.1). Recall that we are employing the moving constraint (5.12) for the double support phase, so the difference between $x_z^m(t)$ and $x_z^M(t)$ is constant;
- $\tilde{x}_z$ is the ZMP position[3] corresponding (through integration) to the chosen velocity tail;
- both the ZMP and the stability constraint have been expressed in continuous time for later convenience (in particular, eq. (5.29) is obtained from (5.13) by splitting the integral in two and plugging the tail in the second integral);
- all constraints are decoupled under the current assumptions, so only the expressions for x are given, as the expressions for y are equivalent.

[3]In the rest of this section, we will for simplicity use the term ‘tail’ for both the ZMP velocity and the corresponding position.

Proposition 4 *At time t_k, IS-MPC is feasible if and only if*

$$x_u^{k,m} \leq x_u^k \leq x_u^{k,M} \tag{5.30}$$

where

$$\begin{aligned} x_u^{k,m} &= \eta \int_{t_k}^{t_{k+C}} e^{-\eta(\tau - t_k)} x_z^m d\tau + \eta \int_{t_{k+C}}^{\infty} e^{-\eta(\tau - t_k)} \tilde{x}_z d\tau \\ x_u^{k,M} &= \eta \int_{t_k}^{t_{k+C}} e^{-\eta(\tau - t_k)} x_z^M d\tau + \eta \int_{t_{k+C}}^{\infty} e^{-\eta(\tau - t_k)} \tilde{x}_z d\tau. \end{aligned}$$

Proof. To show the necessity of (5.30), multiply each side of the ZMP constraint (5.28) by $e^{-\eta(t-t_k)}$

$$e^{-\eta(t-t_k)} x_z^m(t) \leq e^{-\eta(t-t_k)} x_z(t) \leq e^{-\eta(t-t_k)} x_z^M(t), \tag{5.31}$$

and integrate over time from t_k to t_{k+C}.

$$\int_{t_k}^{t_{k+C}} e^{-\eta(\tau-t_k)} x_z^m(\tau) d\tau \leq \int_{t_k}^{t_{k+C}} e^{-\eta(\tau-t_k)} x_z(\tau) d\tau \leq \int_{t_k}^{t_{k+C}} e^{-\eta(\tau-t_k)} x_z^M(\tau) d\tau. \tag{5.32}$$

Adding to all sides the integral term in the right-hand side of (5.29), the middle side becomes exactly x_u^k, while the left- and right-hand sides become $x_u^{k,m}$ and $x_u^{k,M}$ as defined in the thesis.

The sufficiency can be proven by showing that if (5.30) holds then the ZMP trajectory

$$x_z(t) = x_z^M(t) - \frac{x_u^{k,M} - x_u^k}{1 - e^{-\eta T_c}} \tag{5.33}$$

satisfies both the ZMP constraint (5.28) and the stability constraint (5.29). To prove this we check if the following expression is an identity

$$x_u^k = \eta \int_{t_k}^{t_{k+C}} e^{-\eta(\tau-t_k)} \left(x_z^M(\tau) - \frac{x_u^{k,M} - x_u^k}{1 - e^{-\eta T_c}} \right) d\tau + \eta \int_{t_{k+C}}^{\infty} e^{-\eta(\tau-t_k)} \tilde{x}_z d\tau \tag{5.34}$$

which can be verified by splitting the first integral and rearranging the terms

$$\begin{aligned} & x_u^k + \eta \int_{t_k}^{t_{k+C}} e^{-\eta(\tau-t_k)} \frac{x_u^{k,M} - x_u^k}{1 - e^{-\eta T_c}} d\tau = \\ & \eta \int_{t_k}^{t_{k+C}} e^{-\eta(\tau-t_k)} x_z^M(\tau) d\tau + \eta \int_{t_{k+C}}^{\infty} e^{-\eta(\tau-t_k)} \tilde{x}_z d\tau \end{aligned} \tag{5.35}$$

the argument of the integral on the left side is constant, except for the exponential

$$\begin{aligned} & x_u^k + \eta \frac{x_u^{k,M} - x_u^k}{1 - e^{-\eta T_c}} \int_{t_k}^{t_{k+C}} e^{-\eta(\tau-t_k)} d\tau = \\ & \eta \int_{t_k}^{t_{k+C}} e^{-\eta(\tau-t_k)} x_z^M(\tau) d\tau + \eta \int_{t_{k+C}}^{\infty} e^{-\eta(\tau-t_k)} \tilde{x}_z d\tau \end{aligned} \tag{5.36}$$

The integral on the left side now evaluates to $(1 - e^{-\eta T_c})/\eta$, and the left side simplifies to $x_u^{k,M}$. The right side is the definition of $x_u^{k,M}$, which proves that the expression is and identity and thus the trajectory satisfies the stability constraint.

We will now prove that the trajectory satisfies the ZMP constraint $x_z^m(t) \leq x_z(t) \leq x_z^M(t)$

$$x_z^m(t) \leq x_z^M(t) - \frac{x_u^{k,M} - x_u^k}{1 - e^{-\eta T_c}} \leq x_z^M(t) \tag{5.37}$$

We can split this into two inequalities. The right side inequality is simply

$$x_z^M(t) - \frac{x_u^{k,M} - x_u^k}{1 - e^{-\eta T_c}} \leq x_z^M(t) \tag{5.38}$$

$$\frac{x_u^{k,M} - x_u^k}{1 - e^{-\eta T_c}} \geq 0 \tag{5.39}$$

note that $x_u^{k,M} - x_u^k$ is positive by hypothesis because x_u^k is inside the feasibility region, so the right side inequality is verified.

The left side inequality is

$$x_z^m(t) \leq x_z^M(t) - \frac{x_u^{k,M} - x_u^k}{1 - e^{-\eta T_c}} \tag{5.40}$$

$x_z^m(t)$ can be written as $x_z^M(t) - d_{z,x}$ because the ZMP constraint are at a constant distance equal to the size of the foot

$$x_z^M(t) - d_{z,x} \leq x_z^M(t) - \frac{x_u^{k,M} - x_u^k}{1 - e^{-\eta T_c}} \tag{5.41}$$

$$\frac{x_u^{k,M} - x_u^k}{1 - e^{-\eta T_c}} \leq d_{z,x} \tag{5.42}$$

The latter is verified because the size of the feasibility region is $d_{z,x}(1 - e^{-\eta T_c})$, which means that the following is verified

$$x_u^{k,M} - x_u^k \leq d_{z,x}(1 - e^{-\eta T_c}) \tag{5.43}$$

We have shown that trajectory (5.33) is feasible, which proves the thesis. ■

The interpretation of (5.30) is the following: it is the admissible range for x_u at time t_k to guarantee solvability of the QP problem associated to the current iteration of IS-MPC. Since x_u is related to the state variables of the prediction model through (3.11), eq. (5.30) actually identifies the feasibility region in state space.

Note that the extension of the feasibility region is

$$x_u^{k,M} - x_u^{k,m} = \eta \int_{t_k}^{t_{k+C}} e^{-\eta(\tau - t_k)}(x_z^M - x_z^m)d\tau = d_{z,x}(1 - e^{-\eta T_c}), \tag{5.44}$$

where we have used the fact that $x_z^M(t) - x_z^m(t) = d_{z,x}$ for all t, as implied by (8.1). This shows that the extension $x_u^{k,M} - x_u^{k,m}$ of the admissible range for x_u depends on the dimension $d_{z,x}$ of the ZMP admissible region, and tends to become exactly $d_{z,x}$ as the control horizon T_c is increased. On the other hand, the midpoint of

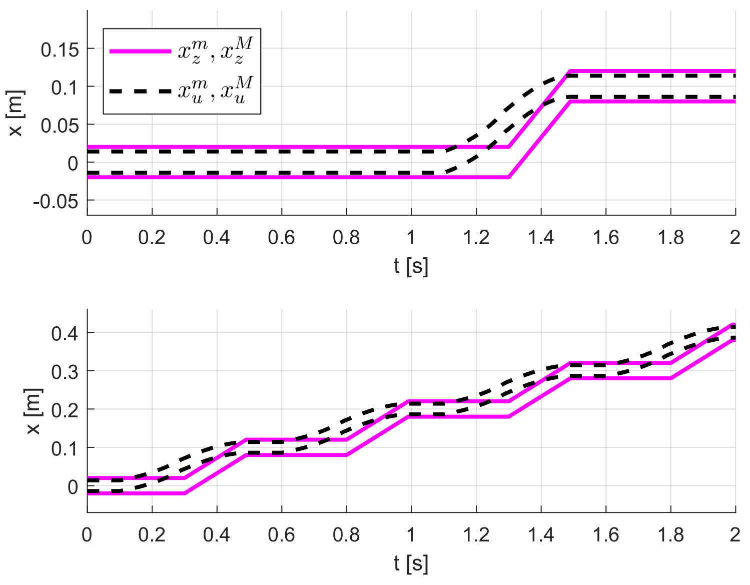


Figure 5.7. Feasibility regions. Top: The robot is taking a single step. Bottom: The robot is taking a sequence of steps. The anticipative tail is used in both cases. The solid magenta lines identify the ZMP constraint along the x-axis, while the dashed black lines show the feasibility region. It can be seen that, when the ZMP is not moving (e.g., top figure before and after the step) the feasibility region tends to identify with the ZMP constraint, except for being slightly smaller due to the finite control horizon. When the ZMP constraint moves, the feasibility region tends to move as well, in a way so as to *anticipate* the change.

this range depends on the tail chosen for the stability constraint (5.29), because $\eta \int_{t_{k+C}}^{\infty} e^{-\eta(\tau - t_k)} \tilde{x}_z d\tau$ acts as an offset in both the left- and right-hand sides of (5.30).

Figure 5.7 illustrates how the admissible range for x_u moves over time, for the case of a single step and of a sequence of steps. These results were obtained with $\bar{z}_c = 0.78$ m, $d_{z,x} = 0.04$ m and $T_c = 0.5$ s. In both cases, an anticipative tail was used, with the residual part truncated; the preview horizon is $T_p = 1$ s. Note that, as expected, the extension of the range is constant and smaller than $d_{z,x}$, and that the range itself gradually shifts toward the next ZMP admissible region as a step is approached.

5.6.2 Recursive feasibility

We prove next that the use of an anticipative tail provides recursive feasibility under a (sufficient) condition on the preview horizon T_p.

Proposition 5 *Assume that the anticipative tail is used in the stability constraint (5.29). Then, IS-MPC is recursively feasible if the preview horizon T_p is sufficiently large.*

Proof. To establish recursive feasibility, we must show that if the IS-MPC QP problem is feasible at t_k, it will be still feasible at time t_{k+1}.

Let us assume that (5.30) holds. This implies that the ZMP constraint (5.28) holds for $t \in [t_k, t_{k+C}]$, and that the stability constraint (5.29) is satisfied, i.e.,

$$x_u^k = \eta \int_{t_k}^{t_{k+C}} e^{-\eta(\tau - t_k)} x_z d\tau + \eta \int_{t_{k+C}}^{\infty} e^{-\eta(\tau - t_k)} \tilde{x}_z(\tau) d\tau,$$

with $\tilde{x}_z$ chosen as the anticipative tail at t_k.

Using (3.12), the value of x_u at t_{k+1} is written as

$$x_u^{k+1} = e^{\eta\delta} - \eta \int_{t_k}^{t_{k+1}} e^{\eta(t_{k+1} - \tau)} x_z(\tau) d\tau.$$

Plugging the above expression for x_u^k in this equation, simplifying, and considering that $x_z(t) \leq x_z^M(t)$ for $t \in [t_k, t_{k+C}]$ we obtain

$$x_u^{k+1} \leq \eta \int_{t_{k+1}}^{t_{k+C}} e^{\eta(t_{k+1} - \tau)} x_z^M(\tau) d\tau + \eta \int_{t_{k+C}}^{\infty} e^{\eta(t_{k+1} - \tau)} \tilde{x}_z(\tau) d\tau.$$

According to Proposition 4, feasibility at t_{k+1} requires[4]

$$x_u^{k+1} \leq \eta \int_{t_{k+1}}^{t_{k+C+1}} e^{\eta(t_{k+1} - \tau)} x_z^M(\tau) d\tau + \eta \int_{t_{k+C+1}}^{\infty} e^{\eta(t_{k+1} - \tau)} \tilde{x}'_z(\tau) d\tau,$$

with $\tilde{x}'_z(\tau)$ in the second integral denoting the anticipative tail at t_{k+1}. Recursive feasibility is then guaranteed if the right-hand side of the last equation is not larger than that of the penultimate. This condition can be rewritten as

$$\begin{aligned} &\int_{t_{k+C}}^{t_{k+C+1}} e^{\eta(t_{k+1} - \tau)} \tilde{x}_z(\tau) d\tau + \int_{t_{k+P}}^{\infty} e^{\eta(t_{k+1} - \tau)} \tilde{x}_z(\tau) d\tau \leq \\ &\int_{t_{k+C}}^{t_{k+C+1}} e^{\eta(t_{k+1} - \tau)} x_z^M(\tau) d\tau + \int_{t_{k+P}}^{\infty} e^{\eta(t_{k+1} - \tau)} \tilde{x}'_z(\tau) d\tau, \end{aligned}$$

where we have used the fact that the anticipative tails at t_k and t_{k+1} coincide over $[t_{k+C+1}, t_{k+P}]$. We introduce the *tail error*

$$\mathcal{T} = \int_{t_{k+P}}^{\infty} e^{\eta(t_{k+1} - \tau)} (\tilde{x}'_z(\tau) - \tilde{x}_z(\tau)) d\tau \tag{5.45}$$

which encodes the difference between the tail used at t_k and the tail used at t_{k+1}. We rewrite the inequality as

$$\int_{t_{k+C}}^{t_{k+C+1}} e^{\eta(t_{k+1} - \tau)} (x_z^M(\tau) - \tilde{x}_z(\tau)) d\tau \geq \mathcal{T}$$

At this point, exploiting the fact (see the end of Sect. 5.4.1) that *(i)* $x_z^M(t) - \tilde{x}_z(t) = d_{z,x}/2$ in the preview horizon, and *(ii)* the residual part of the anticipative tail is truncated, a tedious but simple calculation leads to the condition

$$\frac{d_{z,x}}{2}(1 - e^{-\eta\delta}) \geq \mathcal{T},$$

[4]From now on, we focus only on the right-hand side of the feasibility condition for compactness.

where $(\dot{\tilde{x}}'_z)^{k+P}$ is the last velocity sample in the preview horizon of the anticipative tail at t_{k+1}.

Recall that the tail after the end of the preview horizon is truncated, thus

$$\begin{aligned}\tilde{x}_z &= \tilde{x}_z^{k+P}\\ \tilde{x}'_z &= \tilde{x}_z^{k+P} + \dot{\tilde{x}}_z^{k+P}(\rho(t - t_{k+P}) - \rho(t - t_{k+P+1}))\end{aligned} \tag{5.46}$$

where $\rho(t)$ is the unit ramp. The tail error evaluates to

$$\mathcal{T} = \int_{t_{k+P}}^{\infty} e^{\eta(t_{k+1}-\tau)}(\tilde{x}'_z(\tau) - \tilde{x}_z(\tau)) = e^{-\eta T_p}\frac{\dot{\tilde{x}}_z^{k+P}}{\eta}(1 - e^{-\eta\delta}). \tag{5.47}$$

The tail error can be bounded using an upper bound on the absolute value of $(\dot{\tilde{x}}'_z)^{k+P}$, denoted as $v_{z,x}^{\max}$. We can claim that a sufficient condition for recursive feasibility is

$$T_p \geq T_c + \frac{1}{\eta}\log\frac{2\,v_{z,x}^{\max}}{\eta\,d_{z,x}}, \tag{5.48}$$

thus concluding the proof. ∎

Note the following points.

- An upper bound $v_{z,x}^{\max}$ to be used in (5.48) can be derived (and enforced in the tail) based on the dynamic capabilities of the specific robot or, even more directly, using the information embedded in the footstep sequence and timing.
- Equation (5.48) shows that a longer preview horizon T_p is needed to guarantee recursive feasibility for taller and/or faster robots (larger η and/or $v_{z,x}^{\max}$, respectively), or for robots with more compact feet (smaller $d_{z,x}$).
- Proposition 5 provides only a sufficient condition, and therefore does not exclude that recursive feasibility of IS-MPC can be achieved with a smaller preview horizon, or even with a different tail. For example, in the next subsection we will describe a case (Simulation 3) in which the periodic tail represents a sufficiently accurate conjecture and therefore recursive feasibility is achieved.

5.6.3 Stability

In Sect. 5.6.2 it has been shown that recursive feasibility can be guaranteed by using the anticipative tail, provided that the preview horizon T_p is sufficiently large (Proposition 5). Now we prove that recursive feasibility in turn implies internal stability (i.e., boundedness of the CoM trajectory with respect to the ZMP).

Proposition 6 *If IS-MPC is recursively feasible, then internal stability is guaranteed, i.e., the CoM trajectory is bounded w.r.t. to the ZMP.*

Proof. We have assumed the ZMP trajectory to be piecewise linear in the prediction model (see Sect. 5.2), and the ZMP is limited to the ZMP constraints. These constraints cannot accelerate indefinitely due to the maximum velocity allowed for the robot (recall that the kinematic constraint 5.3 is enforced on the footstep positions). This means that the derivative of the ZMP can be globally bounded by some constant D.

Consider the evolution of the stable component $x_s(t)$ from time t_0 to t_k

$$x_s^k = x_s(t_0)e^{-\eta(t_0-t_k)} + \eta \int_{t_0}^{t^k} e^{-\eta(t_k-\tau)} x_z(\tau) d\tau. \tag{5.49}$$

Let $t_0 \to -\infty$ and change the integration variable to $s = t_k - \tau$. The resulting integral is

$$x_s^k = \eta \int_0^\infty e^{-\eta s} x_z(t_k - s) ds, \tag{5.50}$$

which can be evaluated for a piecewise linear ZMP trajectory (follow the same procedure given in the proof of Proposition 3) to give

$$x_s^k = x_z^k + \frac{1 - e^{-\eta\delta}}{\eta} \sum_{i=0}^{\infty} e^{-i\eta\delta} \dot{x}_z^{k-i-1}. \tag{5.51}$$

This expression is analogous to the stability constraint, but the ZMP velocity samples are selected from past iterations, instead of being future samples.

Summing each member of (5.51) to the expression of the stability constraint (5.19), and recalling that $x_u^k + x_s^k = 2x_c^k$, gives

$$x_c^k - x_z^k = \frac{1 - e^{-\eta\delta}}{2\eta} \sum_{i=0}^{\infty} e^{-i\eta\delta} \left(\dot{x}_z^{k+i} + \dot{x}_z^{k-i-1} \right). \tag{5.52}$$

For this equation to be verified, the stability constraint needs to be verified, which it is because we assumed that IS-MPC is recursively feasible. The absolute value of the right-hand side of (5.52) can be bounded by D/η, which leads to

$$|x_c^k - x_z^k| \leq \frac{D}{\eta}, \tag{5.53}$$

proving the proposition. ■

5.6.4 Effect of the stability constraint

This section presents some MATLAB simulations aimed at showing the effect of the stability constraint on the controlled system. A CoM height $\bar{z}_c = 0.78$ m is used (an appropriate value for the HRP-4 humanoid robot). A sequence of evenly spaced footsteps is given with a constant step duration $T_s = 0.5$ s, split in $T_{ss} = 0.4$ s (single support) and $T_{ds} = 0.1$ s (double support). The dimensions of the ZMP admissible regions are $d_{z,x} = d_{z,y} = 0.04$ m and the sampling time is $\delta = 0.01$ s. The QP problem is solved with the `quadprog` function, which uses an interior-point algorithm.

IS-MPC is compared to a standard MPC for gait generation. In IS-MPC the stability constraint is active using the periodic tail (see Sect. 5.4.1), while in the standard MPC the stability constraint is not present and the cost function is changed to minimizing the norm of the CoM jerk. This corresponds to entrusting the boundedness of the CoM trajectory entirely to the cost function, by penalizing diverging behaviors.

Figure 5.8 shows the performance of IS-MPC and standard MPC for $T_c = 1.5$ s, i.e., 1.5 times the gait period. Both gaits are stable, with the IS-MPC gait more aggressively using the ZMP constraints in view of its cost function that penalizes ZMP variations.

Figure 5.9 compares the two schemes when the control horizon is reduced to $T_c = 1$ s. The standard MPC loses stability: the resulting ZMP trajectory is always feasible but the associated CoM trajectory diverges[5] with respect to it, because the control horizon is too short to allow sorting out the stable behavior via jerk minimization. With IS-MPC, instead, boundedness of the CoM trajectory with respect to the ZMP trajectory is preserved in spite of the shorter control horizon thanks to the embedded stability constraint.

Another interesting situation is that of Fig. 5.10, in which the CoM height is increased to $\bar{z}_c = 1.6$ m while keeping the 'long' control horizon $T_c = 1.5$ s of Simulation 1. Once again, standard MPC is unstable while IS-MPC guarantees boundedness of the CoM with respect to the ZMP. Since it is $\eta^2 = g/\bar{z}_c$, a similar situation can be met when g is decreased, as in gait generation for low-gravity environments (e.g., the moon).

It should be noted that adding to the cost function a term aimed at keeping the ZMP close to the foot center does not allow standard MPC to avoid instability; actually, this occurs even earlier, because the additional cost term has the effect of depenalizing the norm of the CoM jerk. Instead, IS-MPC remains stable also with this modified cost function, with the ZMP pushed well inside the admissible region. This is not surprising as the properties of stability and recursive feasibility are independent of the cost function, and only depend on the constraints.

5.6.5 Effect of the velocity tail on feasibility

Here will be reported some comparative MATLAB simulations showing how different choices for the tail lead to different results in terms of recursive feasibility. The same LIP model and parameters of Sect. 5.6.4 were used. The control horizon T_c is 0.8 s while the preview horizon T_p is 1.6 s.

Figure 5.11 shows a comparison between IS-MPC using the truncated and periodic tail for a regular footstep sequence. When using the truncated tail, gait generation fails because the system reaches an unfeasible state, due to the significant mismatch between the truncated tail and the persistent ZMP velocities required by the gait. Recursive feasibility is instead achieved by using the periodic tail, which coincides with an anticipative tail for this case.

[5]In particular, the divergence occurs in this case on the coronal coordinate y_c. However, it is also possible to find situations where divergence occurs on the sagittal coordinate x_c, or even on both coordinates.

Figure 5.12 refers to a situation in which the assigned footstep sequence is irregular: two forward steps are followed by two backward steps on the same footsteps. Use of the periodic tail leads now to a loss of feasibility, as IS-MPC is wrongly conjecturing that the ZMP trajectory will keep on moving forward. The anticipative tail, which is the recommended choice for this scenario, correctly anticipates the irregularity therefore achieving recursive feasibility.

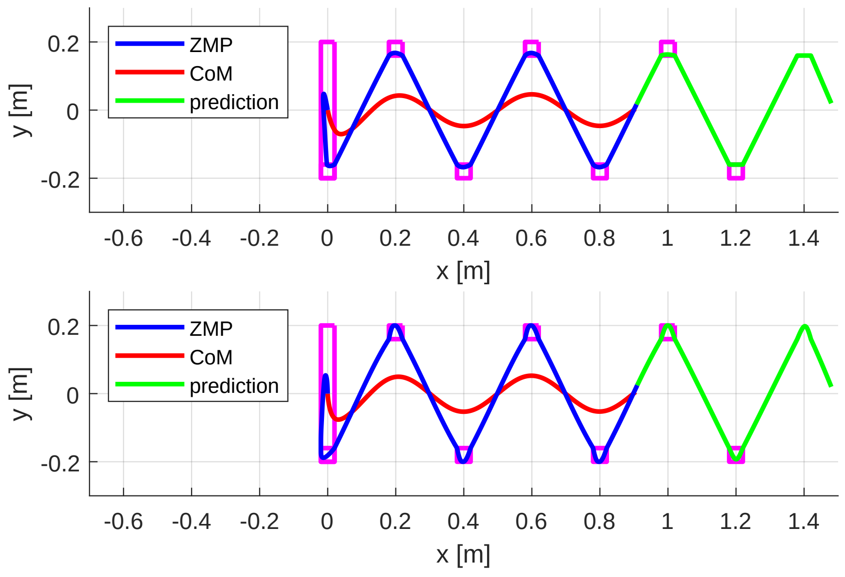


Figure 5.8. Simulation 1: Gaits generated by IS-MPC (top) and standard MPC (bottom) for $T_c = 1.5$ s. The given footstep sequence is shown in magenta. Note the larger region corresponding to the initial double support.

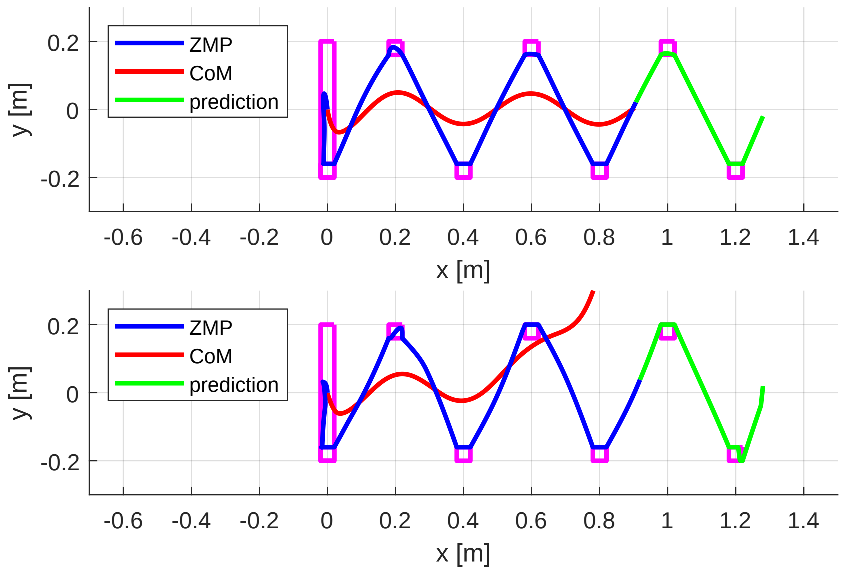


Figure 5.9. Simulation 2: Gaits generated by IS-MPC (top) and standard MPC (bottom) for $T_c = 1.0$ s. Note the instability in the standard MPC solution.

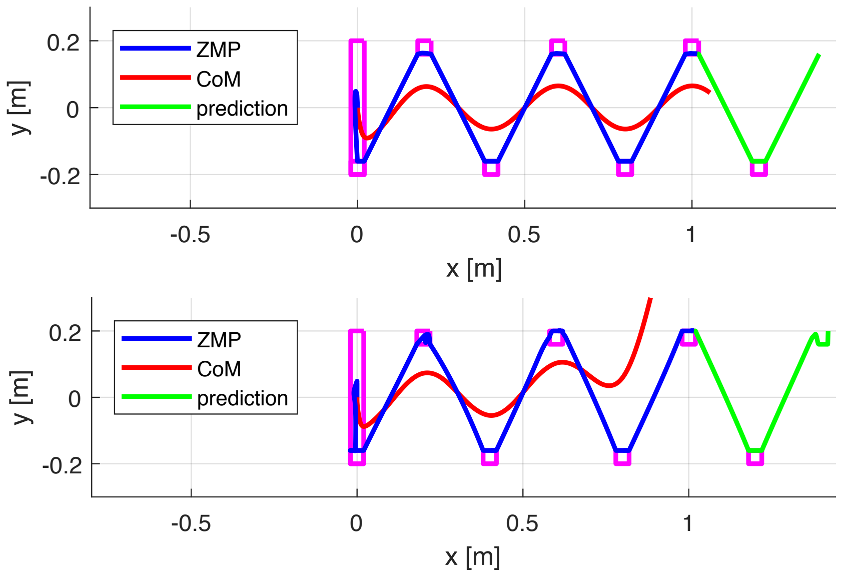


Figure 5.10. Simulation 2 bis: Gaits generated by IS-MPC (top) and standard MPC (bottom) for $T_c = 1.5$ s and a higher CoM. Note the instability in the standard MPC solution.

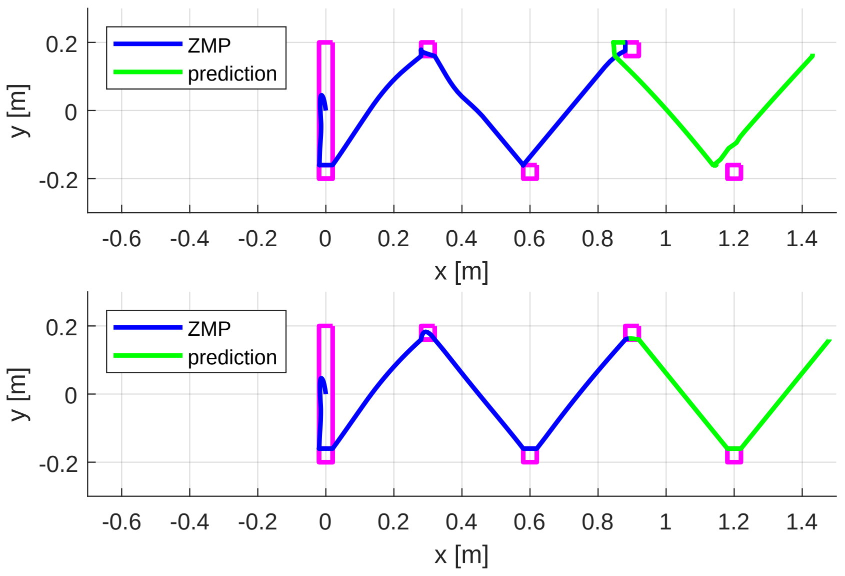


Figure 5.11. Simulation 3: Gaits generated by IS-MPC for a regular footstep sequence with different tails: truncated (top), periodic (bottom). Note the loss of feasibility when using the truncated tail.

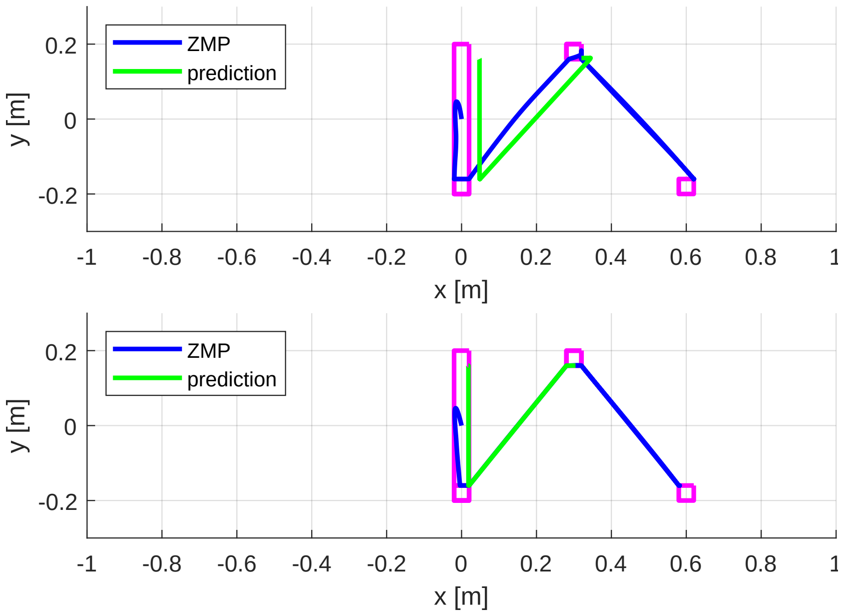


Figure 5.12. Simulation 4: Gaits generated by IS-MPC for an irregular footstep sequence with different tails: periodic (top), anticipative (bottom). The footstep sequence consists of two forward steps followed by two backwards steps on the same footsteps. Note the loss of feasibility when using the periodic tail.

Chapter 6

IS-MPC with automatic footstep placement

Footstep adaptation is an important aspect of legged locomotion. The location where the robot places its footsteps can be planned in advance, but sometimes it might be necessary to modify the planned steps in order to maintain balance. This is even more important if perturbations are present, such as the robot colliding with an obstacle, or an unforeseen unevenness of the ground.

The idea of automatic footstep placement is to let the optimization problem of the MPC scheme, also take care of placing the footsteps [29, 30, 31]. This adds a relatively low number of new variables to the problem, and it is not a heavy burdern on the computation. The advantage is that now the footsteps can be placed by taking into account the state of CoM and ZMP, thus allowing for reactive stepping to maintain balance.

IS-MPC can also be formulated as an automatic footstep placement scheme [3, 4] without needing to change the stability constraint, as this only depends on the future ZMP trajectory and the current state. As is shown in Fig. 6.1, the automatic footstep placement scheme differs very little from the basic scheme of Ch. 5, from an architectural standpoint. The key difference is that the footstep planner now only generates *candidate footsteps*, which the IS-MPC stage will attempt to reproduce. The objective of the optimization will be to place footsteps as close as possible to the candidate footsteps generated by the planner, but in the presence of a disturbance they can be moved so as to maintain balance. The second key difference is that the kinematic constraint is now also part of the IS-MPC stage. This was introduced in the footstep planner as a way to guarantee that the planned footsteps are compatible with the kinematic limitations of the robot, and it is also present in the candidate footstep planner. However, since IS-MPC is allowed to modify the position of the footsteps, we need to include the kinematic constraint in the MPC stage itself.

A proof of recursive feasibility and stability for the case of automatic footstep placement is not presented. A rigorous analysis of these aspects is among the possible future works.

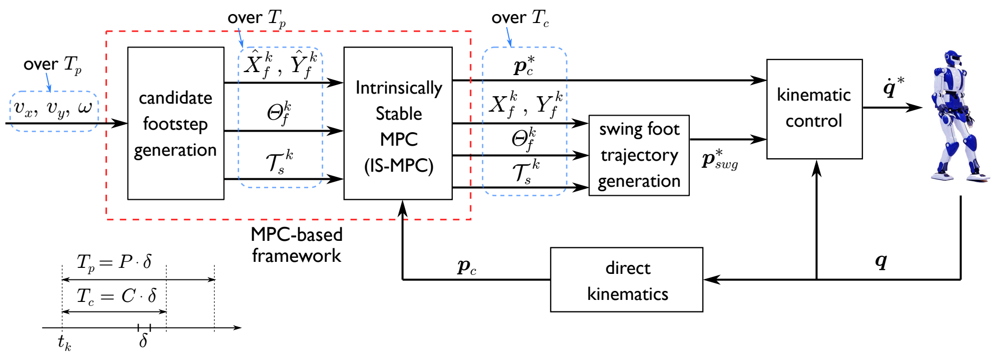


Figure 6.1. The block scheme of IS-MPC with automatic footstep placement.

6.1 Candidate footstep generation

The candidate footstep generator is identical to the footstep generation module described in Sect. 5.1. We employ a difference in notation to emphasize which of the variables computed by the footstep plan can be changed by the IS-MPC stage. These are now referred to as *candidate footstep positions* and denoted as $\hat{X}_f^k, \hat{Y}_f^k$, while the symbols X_f^k, Y_f^k now denote the actual positions, determined by the MPC (see Fig. 6.1).

The footstep timings $\mathcal{T}_f^k$ are not replanned by IS-MPC, as this would result in nonlinear constraints. As a consequence, there is no distinction between candidate and actual timings. The same is true for the footstep orientations Θ_f^k, although a later section will discuss a way to include orientations in the MPC stage, by appropriately linearizing the nonlinear constraints that derive.

6.2 Automatic footstep placement scheme

The expressions of the ZMP constraints (5.10) and (5.12), and the stability constraint (5.4) are identical to the ones introduced in Sect. 5 for the basic scheme. The difference in the ZMP constraints is that now some footstep positions (x_f^j, y_f^j) are variables of the optimization. More specifically, for $j = 1, \ldots, M$ they represent *predicted footsteps*, and are thus considered as variables, while (x_f^0, y_f^0) still represents a fixed value as it is the position of the current support foot.

As already mentioned, the footstep orientations θ_f^j are not considered as variables, because this would make the constraint nonlinear. For this reason, the planned values are maintained and simply plugged in the ZMP constraints. Sect. 6.3 will discuss approximations of the ZMP constraints which allow to include also the orientations as decision variables.

The kinematic feasibility constraint, introduced in Sect. 5.1.2 as a way to generate feasible footsteps in the planner, now needs to also be included as a constraint of the IS-MPC stage itself. This is necessary as, if the candidate footsteps are modified so to maintain balance, the altered footsteps must still respect the kinematic capabilities of the robot, i.e., they cannot stretch too far or cause self collisions between the legs.

The cost function is modified to include a term that penalizes deviation from the candidate footsteps. The resulting QP problems is

$$\left\{\begin{array}{l} \min\limits_{\substack{\dot{X}_z^k,\dot{Y}_z^k\\ X_f^k,Y_f^k}} \quad \|\dot{X}_z^k\|^2 + \|\dot{Y}_z^k\|^2 + \beta\left(\|X_f - \hat{X}_f\|^2 + \|Y_f - \hat{Y}_f\|^2\right) \\ \qquad\qquad\qquad \text{subject to:} \\ \bullet\ \text{ZMP constraints (5.10) and (5.12)} \\ \bullet\ \text{kinematic constraints (5.3)} \\ \bullet\ \text{stability constraints (5.19) for } x \text{ and } y \end{array}\right.$$

6.3 Including footstep orientations

The IS-MPC scheme with automatically placed footsteps of Sect. 6.2 did not include footstep orientations as a variable, because that would introduce a nonlinear constraint. However, we showed in [5] that by constructing approximations of the ZMP and kinematic constraints, it is possible to include the footstep orientations as variables of the IS-MPC stage.

ZMP constraint

The constraints need to be redefined so to become independent on the foot orientation.

In particular, consider the construction in Fig. 6.2 (left). For illustration, assume that the prediction horizon only includes one footstep ($F' = 1$) and $d_{x,z} = d_{y,z}$ (square footprint). The blue square represents the current footstep, while the red squares are two different placements of the predicted footstep (same location but different orientations). The green square, which has the same orientation as the current footstep but size reduced by a factor of $\sqrt{2}$, is always contained in the red squares, irrespective of their orientation. Thus, if the ZMP is located inside the green square, it is certainly contained in the actual footprint, whatever its orientation.

In conclusion, the ZMP constraint during single support can be redefined, for any value of the number of predicted footsteps F', as

$$-\frac{1}{2}\begin{pmatrix}\tilde{d}_{x,z}\\ \tilde{d}_{y,z}\end{pmatrix} \le \begin{pmatrix}x_z^{k+i} - x_f^j\\ y_z^{k+i} - y_f^j\end{pmatrix} \le \frac{1}{2}\begin{pmatrix}\tilde{d}_{x,z}\\ \tilde{d}_{y,z}\end{pmatrix},$$

where $\tilde{d}_{x,z} = d_{x,z}/\sqrt{2}$ and $\tilde{d}_{y,z} = d_{y,z}/\sqrt{2}$.

The above procedure for preserving linearity obviously implies a small reduction of the ZMP constraint area with respect to the actual footprint. However, this effect is more than balanced by the overall increase in the area that becomes feasible for stepping thanks to the inclusion of the footsteps orientation in the IS-MPC stage. The double support constraint is subject to a similar reduction, shown in Fig. 6.2 (right).

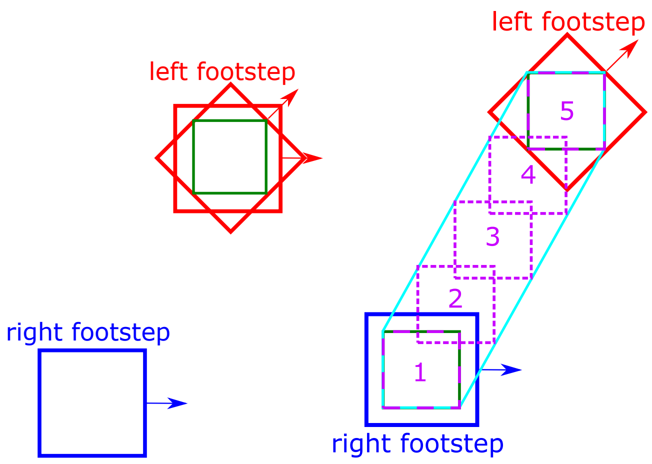


Figure 6.2. Constraint approximations in single-stage IS-MPC. Balance constraint during single support (left) and double support (right).

Kinematic feasibility constraint

The same kind of nonlinearity is manifested in the kinematic feasibility constraint 5.3.

Figure 6.3 shows two different predictions (same location, different orientations) for a right footstep and the corresponding feasible areas (solid line) for placing the next left footstep. Note that both the location and the orientation of these areas depends on the orientation of the right footstep. To remove this dependency, a reduced feasible area (dashed line) is defined in each original area. This reduced region, whose orientation is fixed, is then translated based on the orientation of the right footstep. By forcing the ZMP to be inside the union of all translated regions, we guarantee that it is also inside the union of the original feasibility areas. This is a linear constraint which can be written as

$$\begin{pmatrix} -\tilde{d}_{x,f}/2 \\ \ell - \tilde{d}_{y,f}/2 \end{pmatrix} \leq \begin{pmatrix} x_f^{j+1} - x_f^j - \ell\,\theta^j \\ y_f^{j+1} - y_f^j \end{pmatrix} \leq \begin{pmatrix} \tilde{d}_{x,f}/2 \\ \ell + \tilde{d}_{y,f}/2 \end{pmatrix},$$

with $\tilde{d}_{x,f}, \tilde{d}_{y,f}$ the dimensions of the reduced feasible area.

Note that the above construction is valid for the case in which two footsteps must to be placed within the control horizon ($F' = 2$). In principle, it can be extended to the case of multiple predicted footsteps, but the reduced feasible area shrinks quickly, so that it is only practical for small F'.

Maximum foot rotation constraint

Finally, we must directly add the linear constraint

$$|\theta^j - \theta^{j-1}| \leq \theta_{\max},$$

which was previously enforced only in the footstep generation stage (see Sect. 5.1.2).

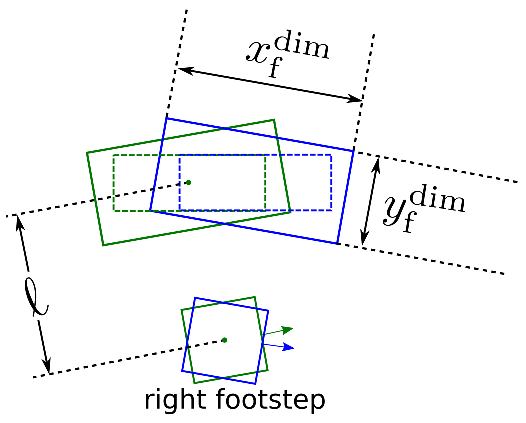


Figure 6.3. Constraint approximations in single-stage IS-MPC. Kinematic constraint on the footstep position approximated as a linear function of the footstep orientation.

6.4 Simulations

This section will show some simulations, performed in the V-REP environment, of IS-MPC with automatic footstep placement. The first two will feature the full-size humanoid HRP-4 which is executing high-level velocity commands. A third simulation will show the small-size humanoid NAO which is commanded to reach a position in the workspace, realizing a walk-to task.

The first two simulations perform automatic footstep placement without including the orientations as decision variables, and use the following parameters. The framework runs at 100 Hz ($\delta = 0.01$ s). Footstep timing is determined using rule (5.1) with $\bar{L}_s = 0.12$ m, $\bar{T}_s = 0.8$ s, $\bar{v} = 0.15$ m/s as cruise parameters, and $\alpha = 0.1$ m/s (as in Fig. 5.3). Each generated T_s is split into T_{ss} (single support) and T_{ds} (double support) using a 60%-40% distribution. Candidate footsteps are generated as explained in Sect. 5.1.2, with $\theta_{\max} = \pi/8$ rad and $\ell = 0.18$ m. In the IS-MPC module, which uses a control horizon T_c of 1.6 s, we have set $\bar{z}_c = 0.78$ m. The dimensions of the ZMP admissible region are $d_{x,z} = d_{y,z} = 0.04$ m, while those of the kinematically admissible region are $d_{x,f} = 0.3$ m, $d_{y,f} = 0.07$ m. The weight in the QP cost function is $\beta = 10^4$.

The first simulation is shown in Fig. 6.4. The robot is commanded a sagittal reference velocity v_x of 0.1 m/s which is then abruptly increased to 0.3 m/s. The preview horizon is $T_p = 3.2$ s and the anticipative tail is used.

In the second simulation, shown in Fig. 6.5, the reference velocities are aimed at producing a cusp trajectory. In particular, initially we have $v_x = 0.2$ m/s and $\omega = 0.2$ rad/s; after a quarter turn we change v_x to -0.2 m/s; after another quarter turn, ω is zeroed. As before, T_p is 3.2 s and the anticipative tail is used for the stability constraint.

In the third simulation the NAO robot, a smaller humanoid, is commanded to reach a goal in the workspace. This simulation is shown in Fig. 6.6 uses $\delta = 0.01$ s, a step duration $T_{\text{s}} = 0.3$ s (0.2 s of single support and 0.1 s of double support) and a prediction horizon $T_{\text{h}} = 0.6$ s, i.e. one gait period. Other parameters are $h_{\text{com}} = 0.26$ m, $d_{x,z} = d_{y,z} = 0.04$ m, $d_{x,f} = 0.1$ m, $d_{y,f} = 0.05$ m, $\ell = 0.125$ m, and $\theta_{\text{max}} = \pi/16$. This simulation is realized by setting the footstep orientations as decision variables of IS-MPC, as described in Sect. 6.3, and modifying the cost function to penalize the distance from the goal (see [5]). The reduced constraints have sizes $\tilde{d}_{x,z} = 0.08$ m and $\tilde{d}_{y,f} = 0.03$ m. Note that this simulation does not employ the candidate footstep generator, as the footsteps are placed directly by the IS-MPC stage.

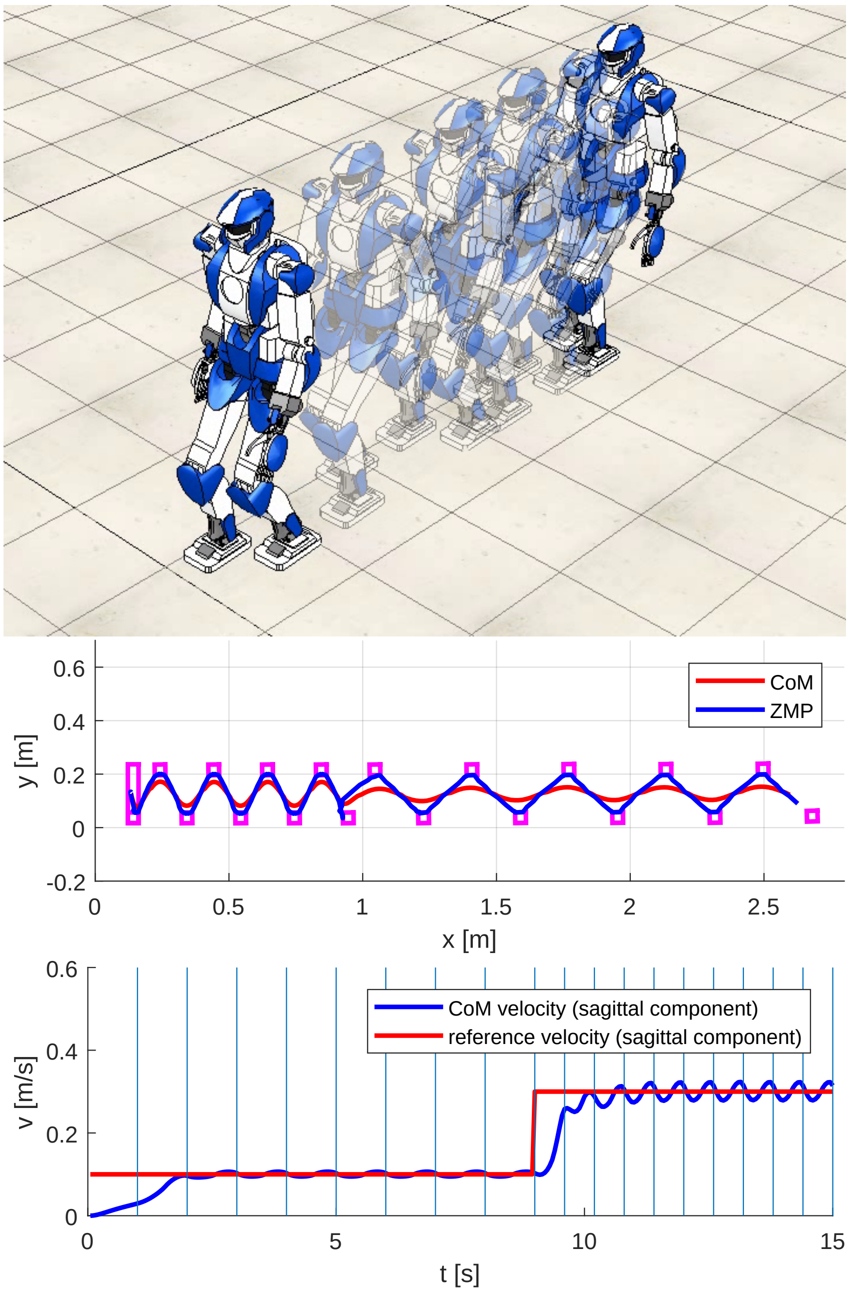


Figure 6.4. HRP-4 walking in a straight line.

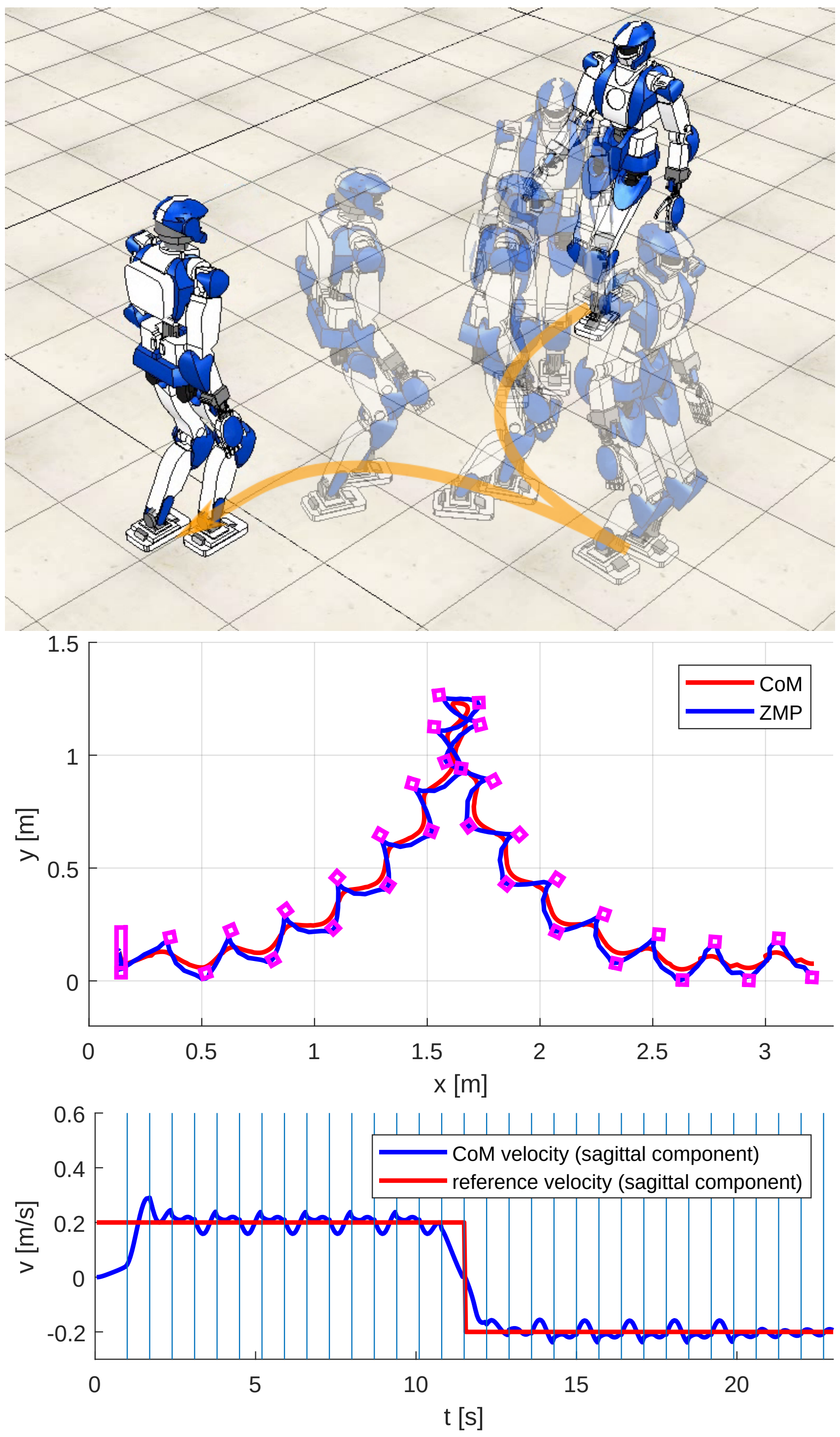
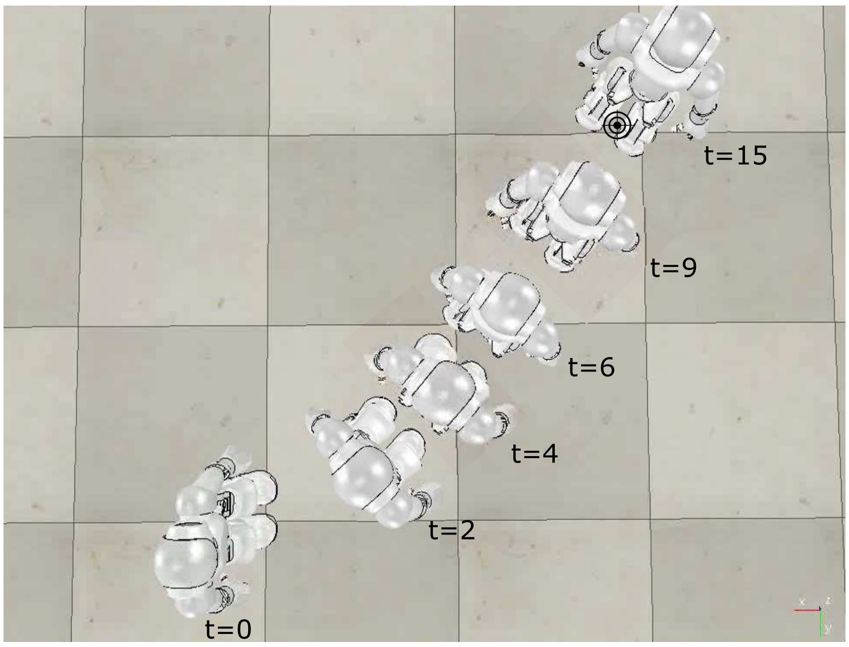


Figure 6.5. HRP-4 walking along a cusp.



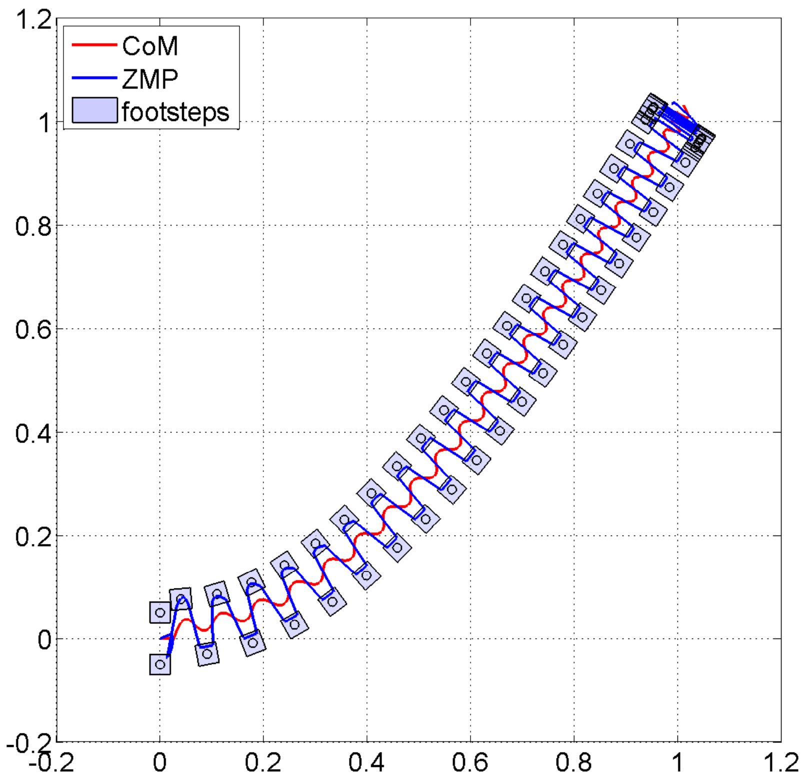


Figure 6.6. NAO successfully walks to the assigned goal (top); the robot CoM, ZMP and footsteps along the generated motion (bottom).

Chapter 7

Robust IS-MPC

There can be several sources of disturbance during locomotion. These can be external, like a push or an unanticipated slope, or internal, like model mismatch due to the adoption of simplifying assumptions in the dynamics. MPC can withstand moderate disturbances thanks to its constant replanning, but stronger perturbations might require appropriate handling, especially for the case of *persistent* disturbances, i.e., continuously applied to the robot. Examples of these are found when the robot is carrying a heavy object, or when it is walking on a slope.

The presence of disturbances on a MPC-controlled system can entail constraint violations, and possibly lead to instability. If a bound on the disturbance is available, a solution is to restrict the constraints using a robust positive invariant set [62, 32, 63, 64]. In order for this set to exist, the system needs to be stabilized.

Since, as it was shown in Sect. 5.6.3 the stability constraint already takes care of stabilizing the system, we wish to provide some guarantee of robustness without the need of an additional stabilization layer. The first technique described in this chapter is aimed at this, as it will employ constraint restriction on the unstable perturbed system directly and establish conditions which preserve recursive feasibility and stability of the IS-MPC scheme when a disturbance is present. Here, some bounds on the disturbance signal will be assumed known.

The second part of the chapter (Sect. 7.5) will discuss the inclusion of an observer in the scheme, which will provide an estimate of the disturbance based on available measures. This results in a less conservative approach against large disturbances. An explicit proof of feasibility and stability is not given for this case, although an analysis of the complete system including the observer would be an interesting development for future work.

7.1 Disturbance model

This section will discuss the characteristics of the disturbances that are taken into account in the rest of the chapter, as well as assumptions on them.

The disturbance w is assumed bounded for obvious physical reasons. An increase in the robustness is possible when, besides the knowledge of the bounds, additional information is known. Consider disturbances $w(t)$ defined as

$$w(t) = w_m^k + \Delta w(t) \quad \text{for } t \in [t_k, t_{k+1}] \tag{7.1}$$

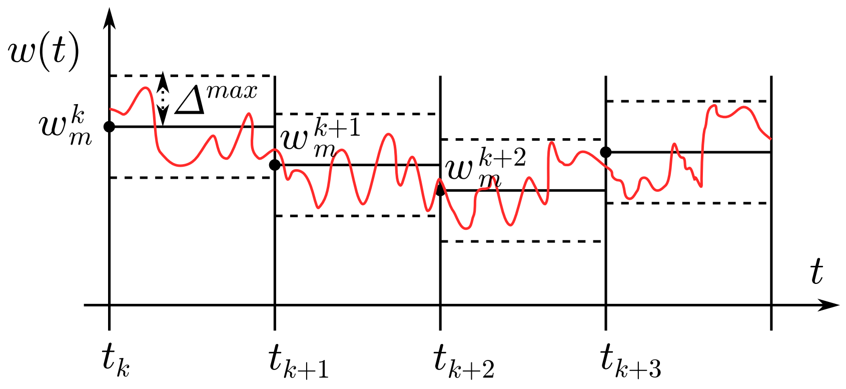


Figure 7.1. A typical disturbance $w(t)$ considered.

where $|\Delta w(t)| \leq \Delta^{max}$ and w_m^k satisfies the additional requirement

$$|w_m^{k+1} - w_m^k| \leq \Delta^{max}. \tag{7.2}$$

The value w_m^k is defined as the *mid range disturbance* in $[t_k, t_{k+1})$. We assume to know the maximum and minimum disturbance value, and thus w_m^k, in each interval and make the simplifying assumption that the range amplitude Δ^{max} is always the same. This can be always obtained in a conservative way by taking Δ^{max} as the maximum range amplitude.

A possible plot of $w(t)$ is shown in Fig. 7.1. This model can represent the apparent force on the CoM derived from the pushing of an object, say a cart. Such force will not be constant, but it could be kept bounded by an arm compliance controller as in [65].

A common special case arises when $w_m^i = w_m$ for all t_i, i.e., we have the same mid range value for every time interval. This situation could represent the disturbance on a humanoid walking on a not perfectly known slope. The disturbance will be in fact composed by the constant push/pull plus the unmodeled dynamics (e.g. variable CoM height).

If furthermore $w_m = 0$ and $w_{min} = -w_{max}$ the disturbance could represent the unmodeled dynamics during a regular gait [35].

7.2 Indirect disturbance compensation

To guarantee boundedness of the CoM with respect to the ZMP in the perturbed case, we employ the stability condition for the perturbed model (4.17) which requires the future knowledge of w. Therefore, in order to provide a causal implementation of (4.17) we should use only the available knowledge of the disturbance at time t_k. The simplest case here considered assumes that the disturbance remains constant, for $t \in [t_k, \infty)$, at the value w_m^k in t_k, and thus we ignore any further available information on values of the future mid range disturbance. The expression of the stability constraint becomes

$$x_u^k = \eta \int_{t_k}^{t_{k+C}} e^{-\eta(\tau - t_k)} x_z(\tau) d\tau + \eta \int_{t_{k+C}}^{\infty} e^{-\omega(\tau - t_k)} \tilde{x}_z(\tau) d\tau - \frac{w_m^k}{\eta^2} \tag{7.3}$$

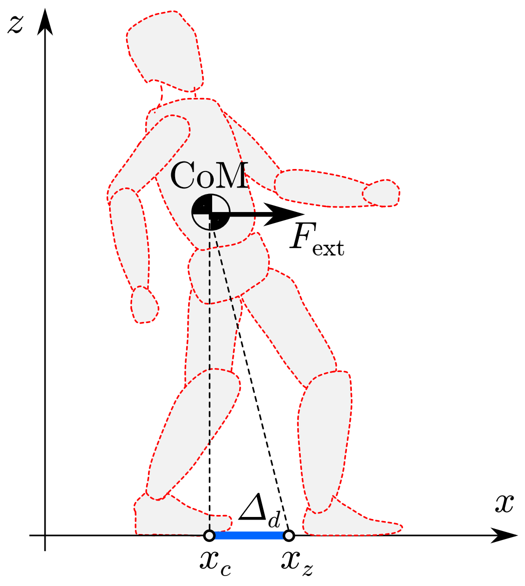


Figure 7.2. Balancing in the presence of a known constant force acting on the CoM: IS-MPC with disturbance compensation produces a steady-state displacement between the ZMP and the COM that can be interpreted as the humanoid "leaning against" the force. This displacement is exactly equal to the disturbance-related term in the stability constraint.

Including the known part w_m^k of the disturbance in the stability constraint leads to a robust IS-MPC scheme where the control inputs (the ZMP velocities within the control horizon) are directly modified by the known profile of the disturbance, realizing a form of *indirect disturbance compensation.*

To appreciate the effect of the compensation, consider the special case in which the humanoid must balance (i.e., footsteps are fixed) in the presence of a constant disturbance $\bar{w} = \bar{F}_{\text{ext}}/m$, arising from a constant force $\bar{F}_{\text{ext}}$ pushing on the CoM, with m the total mass of the robot. Under the action of IS-MPC with disturbance compensation, the robot converges to a steady state where — consistently with eq. (3.13) — the displacement of the ZMP with respect to the COM is

$$x_z - x_c = \frac{\bar{F}_{\text{ext}}}{m\,\eta^2}.$$

This can be interpreted as the humanoid "leaning against" the force in order to counteract it (see Fig.7.2).

The prediction model (5.4) is modified to include the disturbance w_m^k as

$$\begin{pmatrix} \dot{x}_c \\ \ddot{x}_c \\ \dot{x}_z \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ \eta^2 & 0 & -\eta^2 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_c \\ \dot{x}_c \\ x_z \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \dot{x}_z + \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} w_m^k \tag{7.4}$$

and is used to propagate at time t_{k+1} the effect of the first sample $\dot{x}_z^k$ of the QP solution.

7.3 ZMP Constraint restriction

The main tool introduced to guarantee robustness with respect to bounded disturbances is the restriction of the ZMP constraint. Intuitively, if the ZMP is subject to a tighter constraint we create a safety margin against perturbations and uncertainties.

We first define a *restriction function* $R(t)$ as a non-decreasing function[1], defined in $[0, T_c]$ s.t.

$$|R(t)| \leq d/2 \qquad \text{for} \quad t \in [0, T_c] \tag{7.5}$$

where d is the extension of the ZMP constraint in the x-coordinate. The simplest form of restriction function is the linear expression

$$R(t) = r\,t \qquad \text{for} \quad t \in [0, T_c] \tag{7.6}$$

where the slope r is a design parameter to be determined in order to formally guarantee robustness in the presence of bounded disturbances. With this choice, the design parameter r needs to be

$$r \leq \frac{d}{2T_c} \tag{7.7}$$

in order to guarantee (7.5).

The resulting restricted ZMP constraint that will be enforced in the QP at time t_k is then expressed as

$$x_z^m(t) + R(t - t_k) \leq x_z(t) \leq x_z^M(t) - R(t - t_k) \tag{7.8}$$

with $t \in [t_k, t_{k+C}]$. Figure 7.3 shows how the restriction function affects the ZMP constraint over time.

7.4 Robustness against bounded disturbances

The feasibility region discussed in Sect. 5.6.1 is modified by the new stability constraint (7.3) and the restricted ZMP constraint (7.8). The modified region is

$$\begin{aligned} x_u^{k,m} + \int_{t_k}^{t_{k+C}} e^{-\eta(\tau - t_k)} R(\tau - t_k) d\tau - \frac{w_m^k}{\eta^2} \leq x_u^k \leq \\ x_u^{k,M} - \int_{t_k}^{t_{k+C}} e^{-\eta(\tau - t_k)} R(\tau - t_k) d\tau - \frac{w_m^k}{\eta^2}. \end{aligned} \tag{7.9}$$

The following result shows how robust recursive feasibility can be achieved by properly choosing the slope r of the linear restriction function (7.6).

Proposition 7 *Consider the robust IS-MPC in the case of pre-assigned footsteps with anticipative tail. Let $w(t)$ be a disturbance of the form (7.1) acting on the system. Then the robust IS-MPC is recursively feasible if the following linear restriction function is adopted*

$$R(t - t_k) = r\,(t - t_k) \quad t \in [t_k, t_{k+C}] \tag{7.10}$$

[1]This choice is instrumental for simplifying our proof but is not strictly necessary.

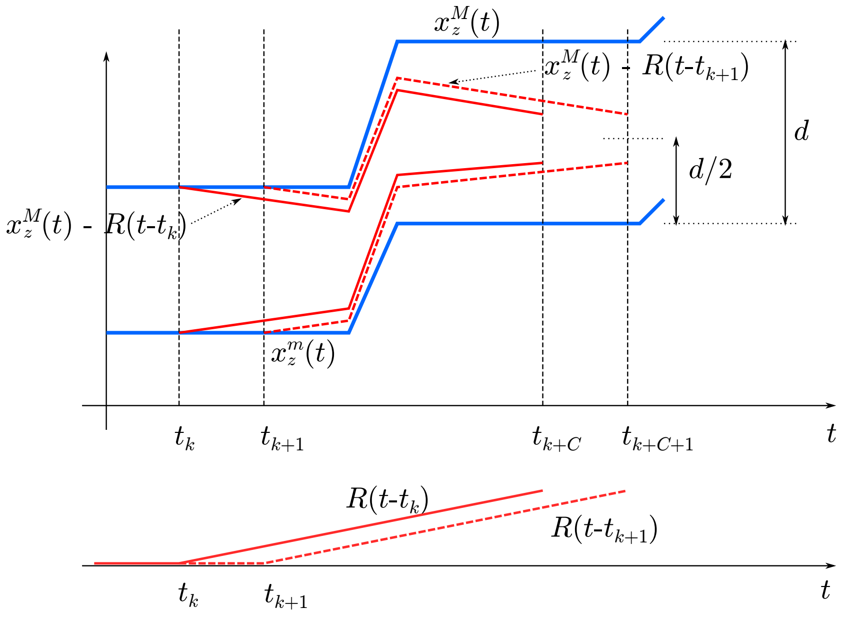


Figure 7.3. Use of the linear restriction function $R(t-t_k) = r(t-t_k)$ in the ZMP constraint.

with r subject to (7.7) and

$$\frac{1}{\delta(1-e^{-\eta T_c})}\left(\frac{\Delta^{max}e^{\eta\delta}}{\eta^2}+\mathcal{T}^{\max}\right)\leq r. \tag{7.11}$$

$\mathcal{T}^{\max}$ *is the upper bound of the tail error, defined in* (5.45).

Proof. The complete proof goes through a set of upper and lower inequalities for the sagittal motion (similarly for the coronal y motion). The lower inequality are omitted for compactness.

We start by assuming that the stability constraint (7.3) and restricted ZMP constraint (7.8) are satisfied in t_k. The first step is to derive a bound on x_u^{k+1}. Integrating (3.14) gives

$$x_u^{k+1} = e^{\eta\delta}x_u^k - \eta\int_{t_k}^{t_{k+1}} e^{\eta(t_{k+1}-\tau)}x_z(\tau)d\tau + \frac{1}{\eta}\int_{t_k}^{t_{k+1}} e^{\eta(t_{k+1}-\tau)}w(\tau)d\tau. \tag{7.12}$$

Plugging (7.3) into (7.12), and recalling that the restricted ZMP constraint (7.8) is satisfied, we can bound x_u^{k+1} as

$$\begin{aligned} x_u^{k+1} \leq & \eta\int_{t_{k+1}}^{t_{k+C}} e^{-\eta(\tau-t_{k+1})}(x_z^M - R(\tau-t_k))d\tau + \\ & \eta\int_{t_{k+C}}^{\infty} e^{-\eta(\tau-t_{k+1})}\tilde{x}_z(\tau)d\tau + \frac{1}{\eta}\int_{t_k}^{t_{k+1}} e^{\eta(t_{k+1}-\tau)}w(\tau)d\tau - \frac{w_m^k e^{\eta\delta}}{\eta^2}. \end{aligned} \tag{7.13}$$

We want to check if x_u is in the feasibility region (5.30) at time t_{k+1}, i.e.

$$x_u^{k+1} \leq \eta\int_{t_{k+1}}^{t_{k+C+1}} e^{-\eta(\tau-t_{k+1})}(x_z^M - R(\tau-t_{k+1}))d\tau + \eta\int_{t_{k+C+1}}^{\infty} e^{-\eta(\tau-t_{k+1})}\tilde{x}_z'(\tau)d\tau - \frac{w_m^{k+1}}{\eta^2} \tag{7.14}$$

where $\tilde{x}'_z(\tau)$ is the tail at t_{k+1}.

By comparing the right-hand side of (7.13) and (7.14) we can obtain conditions for which (7.14) is always satisfied. Rearranging leads to

$$\begin{aligned}
&\eta \int_{t_{k+1}}^{t_{k+C}} e^{-\eta(\tau-t_{k+1})}(R(\tau-t_k)-R(\tau-t_{k+1}))d\tau+\\
&\eta \int_{t_{k+C}}^{t_{k+C+1}} e^{-\eta(\tau-t_{k+1})}(x_z^M-R(\tau-t_{k+1})-\tilde{x}_z(\tau))d\tau+\\
&\eta \int_{t_{k+C+1}}^{\infty} e^{-\eta(\tau-t_{k+1})}(\tilde{x}'_z(\tau)-\tilde{x}_z(\tau))d\tau\\
&-\frac{w^{k+1}-w^k}{\eta^2}-\frac{1}{\eta}\int_{t_k}^{t_{k+1}} e^{\eta(t_{k+1}-\tau)}(w(\tau)-w^k)d\tau \geq 0.
\end{aligned} \tag{7.15}$$

We choose to neglect the second integral as it is always positive and much smaller than the other terms. This leads to a slightly conservative result, but considerably simplifies the computation. We rename the other terms as

$$\mathcal{R} \geq \mathcal{T} + \mathcal{W} \tag{7.16}$$

where

$$\begin{aligned}
\mathcal{R} =&\eta \int_{t_{k+1}}^{t_{k+C}} e^{-\eta(\tau-t_{k+1})}(R(\tau-t_k)-R(\tau-t_{k+1}))d\tau\\
\mathcal{T} =&\eta \int_{t_{k+C+1}}^{\infty} e^{-\eta(\tau-t_{k+1})}(\tilde{x}_z(\tau)-\tilde{x}'_z(\tau))d\tau\\
\mathcal{W} =&\frac{w^{k+1}-w^k}{\eta^2}+\frac{1}{\eta}\int_{t_k}^{t_{k+1}} e^{\eta(t_{k+1}-\tau)}(w(\tau)-w^k)d\tau
\end{aligned} \tag{7.17}$$

For the considered linear restriction $R(t-t_k) = r(t-t_k)$ the term $\mathcal{R}$ becomes

$$\mathcal{R} = r\delta(1-e^{-\eta T_c}). \tag{7.18}$$

$\mathcal{T}$ is the tail error, due to imperfect conjecture on the velocity tail. As showed in Sect. 5.6.2 it can be bounded as

$$\mathcal{T} \leq e^{-\eta T_p}\frac{1-e^{-\eta\delta}}{\eta}v_z^{\max} = \mathcal{T}^{max} \tag{7.19}$$

where $v_z^{\max}$ is the maximum velocity of the ZMP in the tail.

$\mathcal{W}$ depends on the disturbance and can be bounded using assumptions (7.2) and $|\Delta w(t)| \leq \Delta^{max}$.

$$\mathcal{W} \leq \frac{\Delta^{max}e^{\eta\delta}}{\eta^2}. \tag{7.20}$$

By plugging these bounds into (7.16), the result follows. ∎

For the special case $w_m^i = w_m$ for all t_i the previous proposition leads to a less conservative result.

Corollary 1 *Consider the robust IS-MPC in the case of pre-assigned footsteps with anticipative tail. Let the disturbance $w(t)$ acting on the system be of the form (7.1) with $w_m^i = w_m$ for all t_i. Then the robust IS-MPC is recursively feasible if the slope r of the linear restriction function (7.10) satisfying (7.7) is s.t.*

$$\frac{1}{\delta(1-e^{-\eta T_c})}\left(\frac{\Delta^{max}(e^{\eta\delta}-1)}{\eta^2}+\mathcal{T}^{max}\right)\leq r \tag{7.21}$$

where $\mathcal{T}^{\max}$ is the upper bound of the tail error, defined in (5.45).

Proof of Corollary 1. Repeating the proof of Proposition 7 but setting $w_m^{k+1} = w_m^k = w_m$ in (7.17) results in

$$\mathcal{W}\leq\frac{1}{\eta}\int_{t_k}^{t^{k+1}}e^{\eta(t_{k+1}-\tau)}\Delta w(\tau)d\tau\leq\frac{\Delta^{max}e^{\eta\delta}}{\eta^2}(e^{\eta\delta}-1) \tag{7.22}$$

which proves the inequality. ■

A few remarks are in order.

- Restriction (7.21) is less conservative than (7.11) because, when $w_m^k = w_m$, we are using all the available information in the stability constraint.
- Since (7.7) must hold, there is a maximum disturbance amplitude Δ^{max} which is tolerated in order to maintain feasibility of the MPC algorithm. This value is obtained by equating the left side of (7.11) (or (7.21)) to $d/2T_c$.
- If the whole footstep history is assigned, or equivalently if $T_p = \infty$, the tail error is zero since we can choose $\tilde{x}_z = \tilde{x}'_z$ for all $t \geq t_{k+C+1}$ (see proof of Proposition 5 or 7). Therefore $\mathcal{T}^{max}$ is also equal to zero.
- Note that, increasing the control horizon T_c, makes the lower bound on r smaller but also limits the maximum attainable slope. This is due to the particular linear choice of the restriction function $R(t)$. Other choices as linear/saturated or exponential are possible.

In (7.11) we can distinguish the effect on recursive feasibility of two sources of errors in the stability constraint: the presence of an unknown but bounded disturbance and the difference between the conjectured ZMP and its true unknown future history (tail error).

We can summarize the proposed idea by saying that the stability constraint indirectly compensates for most of the known part of the disturbance while the ZMP restriction takes care of the uncertain part in a preventive way based on the known bound.

We finally claim that having guaranteed robust recursive feasibility we also have robust internal stability, i.e., the boundedness of the CoM trajectory with respect to the ZMP trajectory. Due to the physical assumption of bounded disturbances and since we have designed the restriction function to be linear, we are in the same hypothesis of Sect. 5.6.3.

7.5 Observer-based IS-MPC

The results given in the previous section provide guarantees for bounded disturbances, by assuming that the minimum and maximum value of the disturbance are known at each time step. This is equivalent to having information on the mid-range disturbance, and a bound on the oscillation of the actual disturbance around the mid-range. A natural development consists in removing some assumptions on the disturbance and directly estimating it. The estimated term can then be used in the modified stability constraint (7.3).

The estimate can be provided by a disturbance observer employing information from the available measures. In the following we assume that the coordinates of the CoM and the ZMP, respectively x_c and x_z, are measured (details on how these measures can be obtained will be discussed in Sect. 7.6 when presenting the dynamic simulations). Since w is piecewise-linear, we can adopt the following disturbance model (exosystem):

$$\ddot{w} = 0,$$

and use it to extend the perturbed model (7.4), obtaining a system with state $x = (x_c, \dot{x}_c, x_z, w, \dot{w})$ and characterized by the following matrices

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ \eta^2 & 0 & -\eta^2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$
$$C = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}. \tag{7.23}$$

Since the system is easily found to be observable, we can build an asymptotic observer

$$\dot{\hat{x}} = A\hat{x} + Bu + G(C\hat{x} - y) \tag{7.24}$$

where $\hat{x}$ is the observer state and y are the available measurements. The gain matrix G can be computed by simple pole placement. This observer is guaranteed to reconstruct asymptotically any piecewise-linear disturbance signal.

To perform IS-MPC with disturbance compensation in the general case when w is unknown, the estimate $\hat{w}$ provided by the asymptotic observer (7.24) can be used in the stability constraint (7.3). The next section will show some simulations with persistent disturbances, and show that observer-based IS-MPC can provide compensation for constant, as well as for slowly-varying signals.

7.6 Simulations

This section will present a few dynamic simulations obtained in DART (Dynamic Animation Robotics Toolkit). Simulation are performed on the NAO robot, with the CoM height $\bar{z}_c = 0.33$ m, and a square ZMP constraint $d = 0.05$ m. The gait parameters for the single and double support phases are 0.3 s and 0.2 s, respectively; the MPC uses a sampling time $\delta = 0.05$ s, although it is recomputed every 0.01 s

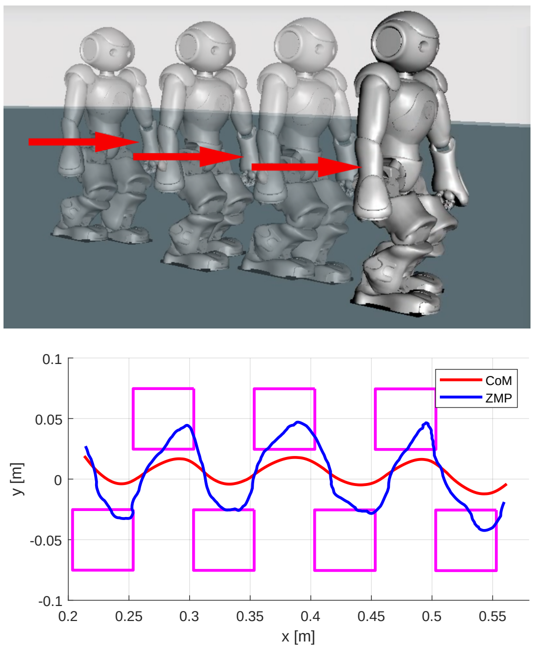


Figure 7.4. NAO walking on flat ground with a constant lateral push from $t = 2.5$ s (top), CoM and ZMP trajectories during the push (bottom).

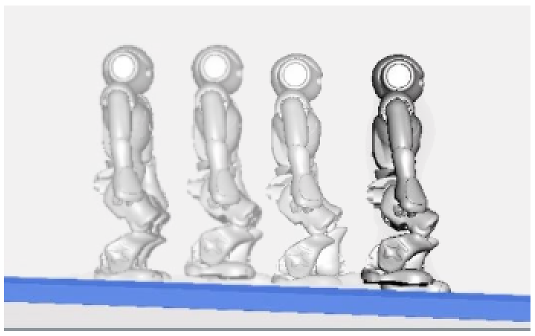
which is the sampling time of the kinematic controller. The control horizon is $T_c = 1$ s and the preview horizon is $T_p = 2$ s.

It is important to notice that, since we are testing our controller in a dynamic simulator, the humanoids unmodeled dynamics represent another source of persistent disturbance that has been ignored. The resulting stable gait in dynamic simulation proves the further degree of robustness achieved by IS-MPC.

In the first simulation the robot walks over flat ground and a constant push of 3.2 N is applied along the positive y axis corresponding to a disturbance $w = 0.71$ m/s^2. The push is applied continuously from $t = 2.5$ s on. In this case we have directly derived the value of w_m^k using a ZMP-based disturbance measure [66], while we assumed a limited variation range of $\Delta^{max} = 0.08$ m/s^2. The value of $r = 0.071$ m/s is given by Proposition 7.

In Fig. 7.4 it is possible to notice that the robot CoM and ZMP trajectories are skewed towards the positive values of the y-axis. This is due to an underestimate of the w_m. The robot is however able to walk without falling, whereas if the constraint restriction is not applied the MPC becomes infeasible, resulting in a failure.

In the second simulation the robot is walking on a descending ramp with an assumed slope in the range $1° \div 2.6°$ (the real value is $2.3°$). The slope generates a disturbance on the robot CoM along the sagittal axis because the gravity force is not completely compensated by the ground reaction. Considering the maximal and minimal disturbance $w_{min} = g \sin(1°) = 0.17$ m/s^2 and $w_{max} = g \sin(2.6°) = 0.45$ m/s^2 leads to $w_m = 0.31$ m/s^2 and $\Delta^{max} = 0.14$ m/s^2. From Corollary 1, a constraint restriction with $r = 0.074$ m/s, is able to guarantee a stable gait that is robust against the disturbance during locomotion, as shown in Fig. 7.5. Also in this case, the MPC fails without constraint restriction. Note that, in this simulation



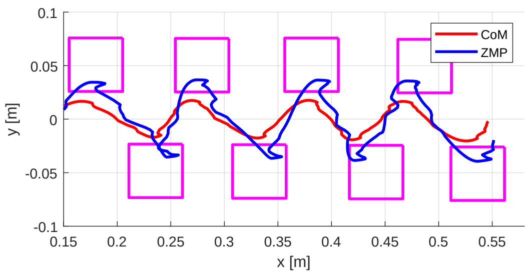


Figure 7.5. NAO walking on a ramp with imperfectly known slope. Simulation snapshots (top), CoM-ZMP trajectories (bottom).

we designed the swing foot trajectory to be compatible with the inclined ground using the true slope value. This is a slight simplification, which might be removed by designing a robust swing foot trajectory.

Note that Fig. 7.4 and 7.5 show the unrestricted ZMP constraints as the restriction is only applied in the prediction. The actual ZMP is allowed to use the full unrestricted area.

In the third dynamic simulation, a constant external force of 3.8 N along the sagittal axis is applied to the CoM, and it is estimated using the observer described in Sect. 7.5, and no ZMP constraint restriction is employed. As shown in Fig. 7.6, the robot falls when nominal IS-MPC is used, whereas observer-based IS-MPC allows to counteract the disturbance successfully, producing the expected effect of leaning against the force (see Sect. 3.3). An interesting aspect of this simulation, clearly shown in the bottom plot, is that the observer does not estimate only the constant force, as it also reacts to dynamic effects that are not modeled in the LIP.

In the fourth simulation, a force $F_{\text{ext}} = 2 + 3.8 \sin 0.45\pi t$ N, which includes a sinusoidal component, acts on both x and y. The disturbance is estimated by the observer, without using ZMP constraint restriction. Figure 7.7 shows a comparison between the CoM trajectories generated by nominal vs. observer-based IS-MPC. Once again, the first fails while the second is able to maintain balance while walking.

The fifth simulation also uses the observer with no ZMP constraint restriction. Here the disturbance is not directly applied to the CoM. A 0.2 kg pendulum is attached to the humanoid arm (this could represent, e.g., an oscillating shopping bag), as in Fig. 7.8. Thanks to the use of observer-based IS-MPC, the robot successfully counteracts the disturbance.

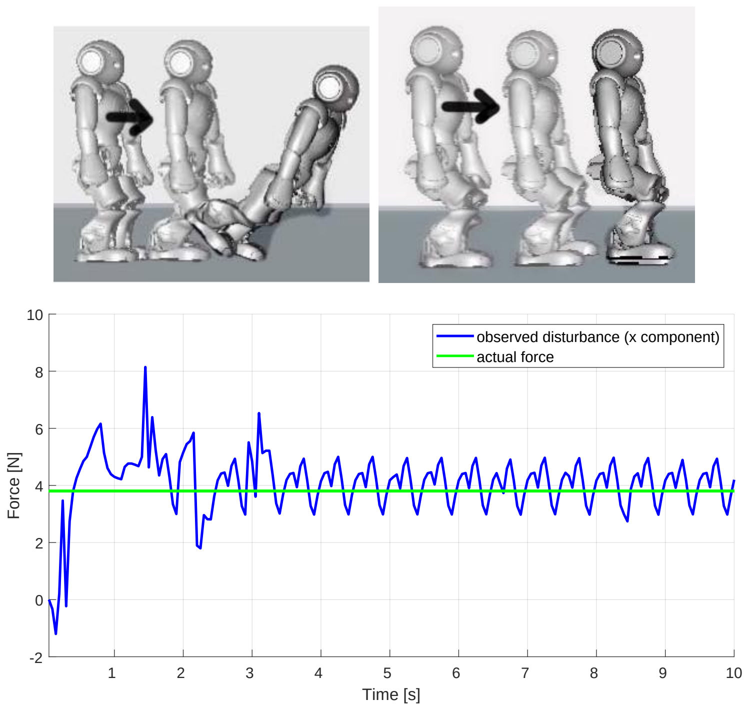


Figure 7.6. NAO dynamic simulation in the presence of an unknown constant force acting on the CoM. With IS-MPC, the robot is unable to maintain balance (top left). With observer-based IS-MPC, the robot successfully counteracts the disturbance (top right). Also shown is the observed force against the actual force (bottom).

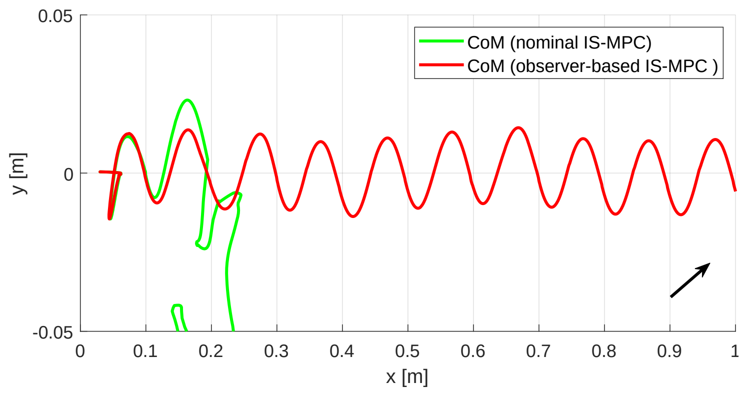


Figure 7.7. NAO dynamic simulation in the presence of a slowly-varying force acting on the CoM. With nominal IS-MPC, the robot is unable to maintain balance, whereas with observer-based IS-MPC a stable gait is achieved.

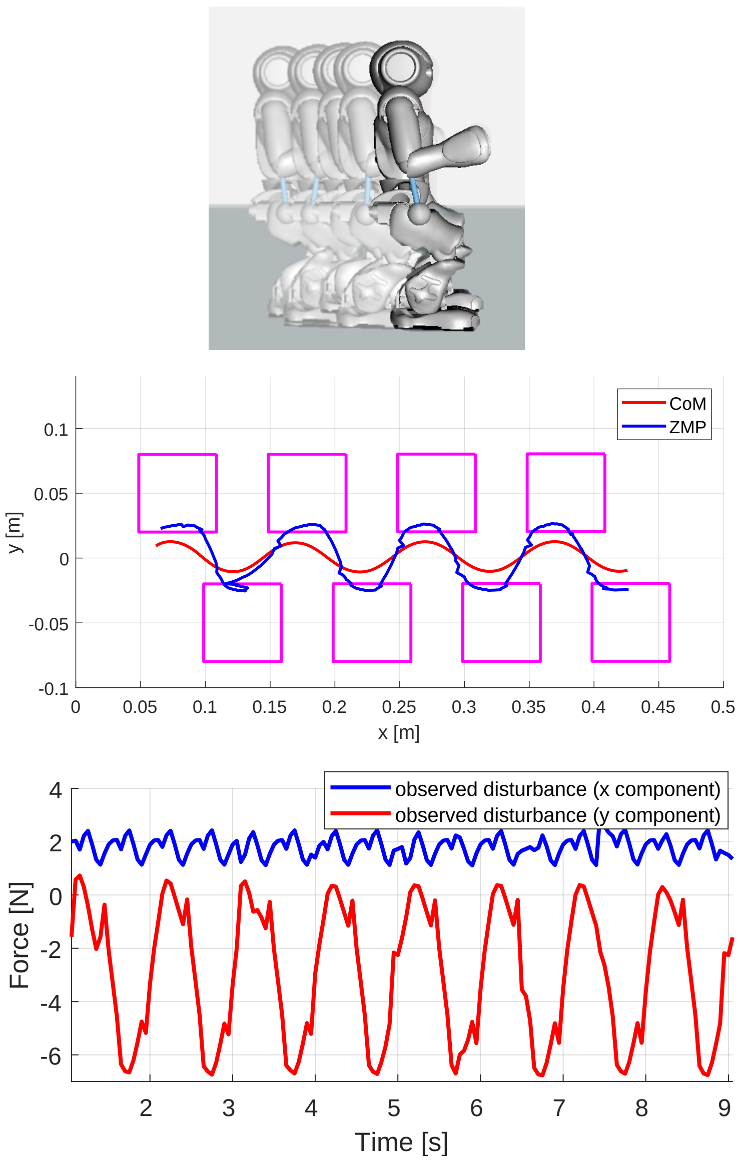


Figure 7.8. NAO dynamic simulation in the presence of a pendulum attached to the arm. Using observer-based IS-MPC, a stable gait is achieved (top). Also shown are the observed disturbances along the two axes (bottom).

Chapter 8

IS-MPC on uneven ground

In principle, one of the advantages of humanoids is the possibility of moving through complex environments. These might include stairs and surfaces at different levels, or cluttered environments for which it might be necessary to step over or onto obstacles. However, while there are a large number of locomotion techniques for flat ground, walking on uneven surfaces poses additional challenges which make it largely an open field of research.

In the basic version of IS-MPC presented in Ch. 5, as well in the extension of Ch. 6, the ground was assumed to be flat. The main reason for this assumption is that it allows the use of the LIP model, which requires a constant CoM height (see Sect. 3.2). The robust version of Ch. 7 is able to tolerate some unevenness, but in practice moving on uneven ground requires generating vertical CoM trajectories using a suitable 3D model of the dynamics.

In this chapter we wish to extend the basic scheme to the case where the environment is a *world of stairs*, i.e., the ground is constituted by horizontal patches at different heights. We will employ the 3D extension of the LIP, which was discussed in Sect. 3.4.

There will be two main difference with the basic IS-MPC scheme: the footstep generator and the ZMP constraint. The footstep generator of Sect. 5.1.2 was based on convex optimization, which cannot be used on uneven ground, as most constraint become nonlinear. As an example, consider a simple staircase: the foot has to be placed inside any of the steps, but not in between two different steps, in order for the foot to not overhang. This makes the constraint on the footstep position non-convex. The planner of choice will be based on random exploration, and will build a tree of feasible footstep positions in the scene in order to find the path to a given goal.

The ZMP constraint needs to be modified according to the new balance condition (see Sect. 3.4), which requires the 3D version of the ZMP to remain inside a convex polyhedron, whose base is determined by the contact points, and whose vertex is the CoM. This constraint is nonlinear, but if the height difference between consecutive steps is small enough it can be well approximated as a linear constraint.

Algorithm 1: Footstep Planner

```
1  root the tree $\mathcal{T}$ at $v_{\text{ini}} \leftarrow (\boldsymbol{f}_L, \boldsymbol{f}_R)$;
2  $i \leftarrow 0$;
3  repeat
4      $i \leftarrow i + 1$;
5      generate a random point $\boldsymbol{p}_{\text{rand}}$ on the ground;
6      select the closest vertex $v_{\text{near}}$ in $\mathcal{T}$ to $\boldsymbol{p}_{\text{rand}}$ according to $\gamma(\cdot, \boldsymbol{p}_{\text{rand}})$;
7      randomly select from the primitive catalogue $U$ a candidate footstep $\boldsymbol{f}^{\text{cand}}$;
8      if $\boldsymbol{f}^{\text{cand}}$ is feasible w.r.t. R1–R2 then
9          $h \leftarrow h_{\text{min}}$;
10         $\boldsymbol{p}^{\text{cand}}_{\text{swg}} \leftarrow$ BuildTrajectory$(\boldsymbol{f}^{\text{near}}_{\text{swg}}, \boldsymbol{f}^{\text{cand}}, h)$;
11         while $h \leq h_{\text{max}}$ and Collision$(\boldsymbol{p}^{\text{cand}}_{\text{swg}})$ do
12             $h \leftarrow h + \Delta h$;
13             $\boldsymbol{p}^{\text{cand}}_{\text{swg}} \leftarrow$ BuildTrajectory$(\boldsymbol{f}^{\text{near}}_{\text{swg}}, \boldsymbol{f}^{\text{cand}}, h)$;
14         end
15         if $h \leq h_{\text{max}}$ then
16             $v_{\text{new}} \leftarrow (\boldsymbol{f}^{\text{cand}}, \boldsymbol{f}^{\text{near}}_{\text{sup}})$;
17             add vertex $v_{\text{new}}$ to $\mathcal{T}$ as a child of $v_{\text{near}}$;
18             compute midpoint $\boldsymbol{m}$ between the feet at $v_{\text{new}}$;
19         end
20     end
21 until $\boldsymbol{m} \in \mathcal{G}$ or $i = i_{max}$;
```

8.1 Footstep planning with random exploration tree

A humanoid robot is assigned a *walk-to* locomotion task to a desired goal region $\mathcal{G}$. The environment is a *world of stairs*, i.e., the ground is uneven and composed of horizontal patches located at different heights. Depending on its elevation with respect to the neighboring areas, a patch may be accessible for the humanoid to climb on from an appropriate direction, or else represent an obstacle to be avoided. Some low-height patches may be shaped in such a way that the humanoid may decide to go over (rather then stepping on) them.

A natural choice for representing the considered kind of ground is a 2.5D grid map of equally-sized cells, also called *elevation map* [67]. This will be denoted as $\mathcal{M}_z$, and assumed known in advance so that whenever needed it can be queried as $z = \mathcal{M}_z(x, y)$, to provide the height of the ground at the cell having coordinates (x, y).

The planner is in charge of finding a sequence of footsteps $\{\boldsymbol{f}^j\}$ leading to the desired location $\mathcal{G}$, together with the associated swing foot trajectories $\{\boldsymbol{p}^j_{\text{swg}}\}$ between consecutive footsteps. Here, $\boldsymbol{f}^j = (x^j_f, y^j_f, z^j_f, \theta^j_f)^T$ is the pose of the j-th footstep, with x^j_f, y^j_f, and z^j_f representing its position and θ^j_f its orientation.

The footstep sequence needs to be feasible, i.e., each element $\{\boldsymbol{f}^j\}$ and $\{\boldsymbol{p}^j_{\text{swg}}\}$ must satisfy the following requirements:

R1 The height variation w.r.t. the previous footstep is within a maximum range, i.e., $|z^j_f - z^{j-1}_f| \leq \Delta z_{\text{max}}$.

R2 The footstep is fully in contact within a single horizontal patch, i.e., each cell of $\mathcal{M}_z$ belonging to the footprint has the same height z_f^j.

R3 Apart from the ground contact at the start and at the end, the swing foot trajectory $\boldsymbol{p}_{\text{swg}}^j$ is collision-free.

Note that there is no requirement on the distance between consecutive footsteps, as this would be automatically satisfied as each new footstep is generated using a primitive catalogue U (see Fig. 8.1). Elements in the catalogue are chosen so as to be kinematically acceptable for the robot in use.

A pseudocode of the footstep planner is given in Algorithm 1. The idea is to iteratively build a tree $\mathcal{T}$ in the search space, where a vertex $v = (\boldsymbol{f}_{\text{sup}}, \boldsymbol{f}_{\text{swg}})$ specifies the poses $\boldsymbol{f}_{\text{sup}}$ and $\boldsymbol{f}_{\text{swg}}$ of the two feet during a double support phase. In all steps originating from v, the support foot will remain at $\boldsymbol{f}_{\text{sup}}$ while the swinging foot will move from $\boldsymbol{f}_{\text{swg}}$. An edge exists between two vertices when there exists a collision-free trajectory of the swing foot connecting the two.

The algorithm starts by rooting $\mathcal{T}$ at $v_{\text{ini}} = (\boldsymbol{f}_L, \boldsymbol{f}_R)$, where $\boldsymbol{f}_L$ and $\boldsymbol{f}_R$ are the foot poses at the starting configuration. The initial support foot can be chosen arbitrarily.

The generic iteration of the algorithm begins by selecting a sample point $\boldsymbol{p}_{\text{rand}}$ on the ground. We allow the planner to randomly choose between exploration and exploitation, to bias the growth of the tree towards, respectively, unexplored regions and the goal. In the first case, $\boldsymbol{p}_{\text{rand}}$ is generated by randomly choosing its x, y coordinates and retrieving the corresponding z coordinate from $\mathcal{M}_z$. In the second case, $\boldsymbol{p}_{\text{rand}}$ is sampled from the goal region $\mathcal{G}$.

Then, the vertex v_{near} of $\mathcal{T}$ that is closest to $\boldsymbol{p}_{\text{rand}}$ is selected for an expansion attempt. The following metric is used for evaluating the distance between a certain vertex v and a point $\boldsymbol{p} \in I\!R^3$:

$$\gamma(v, \boldsymbol{p}) = d(\boldsymbol{m}, \boldsymbol{p}) + \alpha|\theta_p|.$$

Here, $d(\boldsymbol{m}, \boldsymbol{p})$ is the Euclidean distance of the midpoint $\boldsymbol{m}$ between the feet (at the poses $\boldsymbol{f}_{\text{sup}}$ and $\boldsymbol{f}_{\text{swg}}$ specified in v) from $\boldsymbol{p}$; θ_p is the angle between the robot sagittal axis (defined as the axis passing through $\boldsymbol{m}$ with orientation equal to the average of the orientations of the two footsteps) and the line joining $\boldsymbol{m}$ to $\boldsymbol{p}$; α is a positive scalar.

Once v_{near} has been identified, the foot poses $\boldsymbol{f}_{\text{sup}}^{\text{near}}$ and $\boldsymbol{f}_{\text{swg}}^{\text{near}}$ are extracted from it. A candidate footstep $\boldsymbol{f}^{\text{cand}}$ is generated by randomly selecting a final pose of the swinging foot from the catalogue U of primitives, which is defined with respect to $\boldsymbol{f}_{\text{sup}}^{\text{near}}$ (see Fig. 8.1). The z coordinate of $\boldsymbol{f}^{\text{cand}}$ is retrieved from the elevation map $\mathcal{M}_z$.

At this point, the candidate footstep is checked for feasibility with respect to requirements R1–R2. If the outcome is positive, the algorithm verifies whether a collision-free trajectory $\boldsymbol{p}_{\text{swg}}^{\text{cand}}$ exists that brings the swinging foot from $\boldsymbol{f}_{\text{swg}}^{\text{near}}$ to $\boldsymbol{f}^{\text{cand}}$ (requirement R3). This is done by using a parametrized trajectory which, once the endpoints are selected, can be deformed[1] by changing the maximum height h along

[1]As a deformable trajectory we used a polynomial, but other choices (e.g., B-splines and Bezier curves) are also possible.

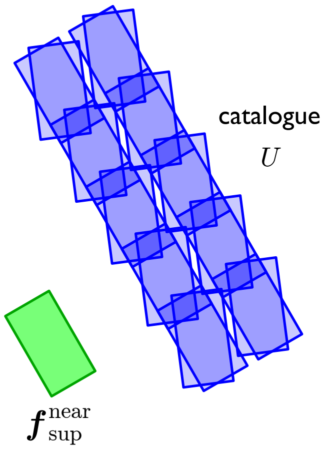


Figure 8.1. The catalogue U of primitives specifies the possible planar poses (in blue) of the candidate footstep $\boldsymbol{f}^{\text{cand}}$ with respect to the pose $\boldsymbol{f}_{\text{sup}}^{\text{near}}$ of the current support foot (in green). Here we have considered the case of a swinging right foot; the catalogue for a swinging left foot is specular. Once a primitive is chosen, the z coordinate of the candidate footstep will be retrieved from the elevation map $\mathcal{M}_z$.

the trajectory. Starting from a lower bound h_{min}, the algorithm iteratively checks increasing values for h up to an upper bound h_{max}. If a collision-free trajectory is found, a new vertex $v_{\text{new}} = (\boldsymbol{f}^{\text{cand}}, \boldsymbol{f}_{\text{sup}}^{\text{near}})$ is added to the tree, connected to its parent v_{near} by $\boldsymbol{p}_{\text{swg}}^{\text{cand}}$. Note that in v_{new} the roles of the feet have been swapped: in future steps originating from v_{new}, the support foot will stay at $\boldsymbol{f}^{\text{cand}}$ while the swinging foot will move from $\boldsymbol{f}_{\text{sup}}^{\text{near}}$.

Planning terminates when v_{new} completes the walk-to task (i.e., the corresponding midpoint $\boldsymbol{m}$ belongs to $\mathcal{G}$) or a maximum number of iterations has been reached. In the first case, the path joining the root to v_{new} is extracted from the tree, and the footstep sequence $\{\boldsymbol{f}^j\}$ is reconstructed with the associated swing foot trajectories $\{\boldsymbol{p}_{\text{swg}}^j\}$.

8.2 IS-MPC on uneven ground

This section will describe the constraints that are used in IS-MPC for uneven ground. These includes the 3D ZMP constraint (which in its general form is nonlinear) and its linear approximation, and the 3D stability constraint, which is derived from the model in Sect. 3.4. There is no constraint on the footstep positions in the IS-MPC stage, as in the 3D case we assume the planned footsteps to be non-modifiable. The modified cost function will be discussed at the end of the section.

3D ZMP constraint

As discussed in Sect. 3.4, in the 3D case the ZMP is allowed to leave the ground in order to generate vertical CoM motions, and correspondingly $\dot{z}_z^{k+i}$ becomes a control variable. As illustrated in Fig. 8.2 (left), the balance condition now requires the ZMP to remain inside the polyhedral cone defined by the support polygon and the CoM. However, when the ZMP is allowed to move vertically the cone defines a

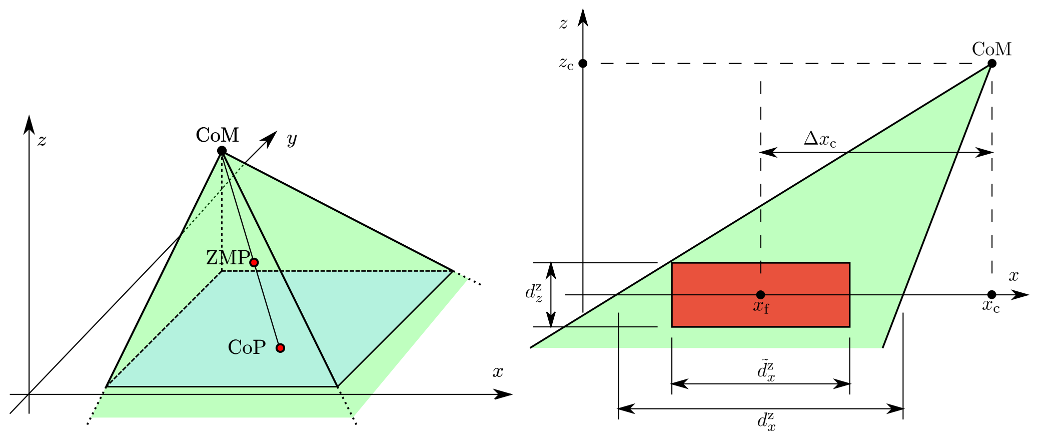


Figure 8.2. Left: the balance condition in the 3D case is satisfied if the ZMP is inside the green polyhedral cone. Right: side view of the polyhedral cone (green) representing the actual ZMP constraint and the box (red) used for defining approximate linear constraint. Here, the CoM is at its maximum allowed displacement Δx_c with respect to the center of the support foot.

nonlinear constraint. In order to remove this nonlinearity, a box constraint is used instead (see Fig. 8.2):

$$-\frac{1}{2}\begin{pmatrix} \tilde{d}_{x,z} \\ \tilde{d}_{y,z} \\ d_{z,z} \end{pmatrix} \leq R_{k+i}^T \begin{pmatrix} x_z^{k+i} - x_f^{k+i} \\ y_z^{k+i} - y_f^{k+i} \\ z_z^{k+i} - z_f^{k+i} \end{pmatrix} \leq \frac{1}{2}\begin{pmatrix} \tilde{d}_{x,z} \\ \tilde{d}_{y,z} \\ d_{z,z} \end{pmatrix}, \tag{8.1}$$

where $d_{z,z}$ is a design parameter that defines the maximum allowed vertical ZMP displacement w.r.t. the horizontal patch. To guarantee that the box is contained in the cone, its x and y dimensions are reduced to $\tilde{d}_{x,z}$ and $\tilde{d}_{y,z}$. Suitable values for these parameters are

$$\tilde{d}_{x,z} = d_{x,z}\left(1 - \frac{d_{z,z}}{2z_c^{\min}}\right) - \frac{d_{z,z}}{z_c^{\min}}\Delta x_c,$$

where Δx_c is the maximum expected displacement of the CoM with respect to the center of the support foot and $z_c^{\min}$ is the minimum expected value for the CoM height. An analogous formula can be written for $\tilde{d}_{y,z}$.

Similarly to the 2D case, the box constraint is kept fixed during single support. During double support, the box slides linearly from its position around the previous support foot to its position around the next support foot (see the corresponding 2D case in Sect. 5.12), thus always remaining within the polyhedral cone which defines the ZMP balance constraint for this situation (see Fig. 8.3).

3D stability constraint

The stability constraint along x and y is unchanged from the basic case presented in Sect. 5.4, and whichever tail is most appropriate for the situation can be used. The z equation requires its own stability constraint, which assumes a form very similar

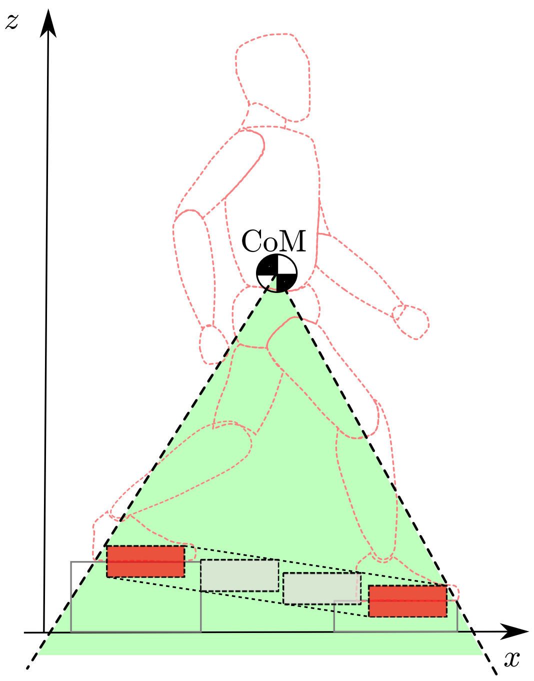


Figure 8.3. During double support, the box constraint slides from the previous to the next support foot.

to the other two, but with an additional term

$$\sum_{i=0}^{C-1} e^{-i\eta\delta}\dot{z}_z^{k+i} = -\sum_{i=C}^{\infty} e^{-i\eta\delta}\dot{z}_z^{k+i} + \frac{\eta}{1-e^{-\eta\delta}}\left(z_u^k - z_z^k - \frac{g}{\eta^2}\right). \tag{8.2}$$

Here too, any of the tails discussed in Sect. 5.4 could be used to approximate the contribution beyond the control horizon. Clearly, the anticipative tail will be the preferred choice for feasibility reasons. In the paper where this 3D formulation was first proposed [7], we argued however that the periodic tail is not a good choice here, because the vertical motion of the CoM, unlike the horizontal components, cannot be expected to exhibit a cyclic behavior.

QP problem

The QP problem is modified to include the 3D version of the constraints, as well as a modified cost function.

$$\left\{\begin{array}{l} \displaystyle\min_{\dot{X}_z^k,\dot{Y}_z^k,\dot{Z}_z^k} \sum_{i=0}^{N-1}\left((\dot{x}_z^{k+i})^2 + (\dot{y}_z^{k+i})^2 + (\dot{z}_z^{k+i})^2 + \beta(z_z^{k+i} - z_{\mathrm{f}}^{k+i})^2\right) \\ \text{subject to:} \\ \bullet\ \text{ZMP constraints (8.1)} \\ \bullet\ \text{stability constraints for } x,\ y \text{ and } z \text{ (5.18) and (8.2)} \end{array}\right.$$

Here the IS-MPC stage does not perform any kind of footstep adaptation, as it only employs the planned footsteps as for the case of the basic scheme of Ch. 5. The

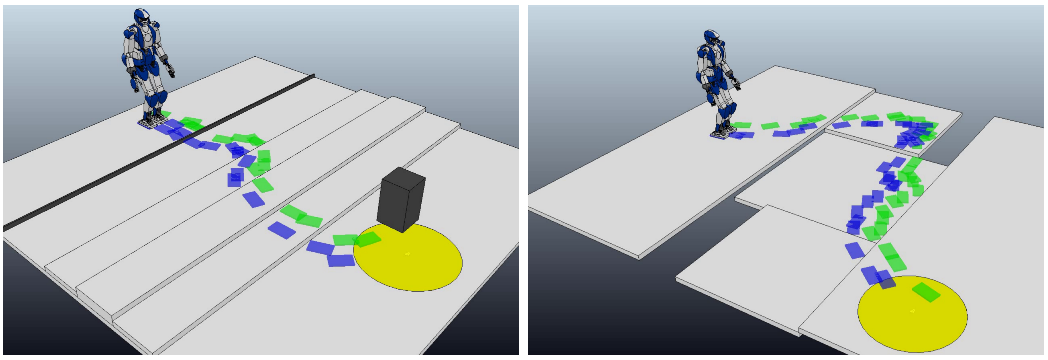


Figure 8.4. The footstep plans for both simulations.

cost function thus only contains the decision variables (ZMP derivatives) as well as a term which is charge of bringing the vertical component of the ZMP at ground level, whenever this is possible.

8.3 Simulations

This section shows two simulations which utilize the footstep planner of Sect. 8.1 along with the 3D gait generation scheme of Sect. The simulations are shown in Fig. 8.4 and 8.5. The environment is assumed known and the robot has to reach a circular goal region of radius 0.5 m. The robot is HRP-4 and the simulations environment is V-REP.

The parameter settings are the same for both these simulations. In the footstep planning module we have set $\alpha = 1$, $\Delta z_{\max} = 0.08$ m, $h_{\min} = 0.02$, $h_{\max} = 0.12$ and $\Delta h = 0.02$. For the gait generation module, we used $\omega = 3.6$ s$^-1$, the step duration is $T_{\mathrm{s}} = 0.8$ s of which 0.5 s of single support and 0.3 s of double support, the size of the constraints is $\tilde{d}_{x,z} = \tilde{d}_{y,z} = 0.08$ m and $d_{z,z} = 0.03$ m. Finally, the elevation map $\mathcal{M}_z$ has a resolution of 0.02 m.

In the first scenario, the robot and the goal are placed at opposite sides of the environment. The goal can be reached by walking a mostly straight path, but footsteps need to be placed appropriately to satisfy all the requirements. Figure 8.5 (top) shows one of the solutions found, with the underlying footstep sequence in Fig. 8.4 (top). The robot starts walking forward and reaches the proximity of the bar obstacle, which does not provide a large enough surface to step on. Then, the robot goes over it by taking a longer step, with a swing foot trajectory sufficiently high to avoid collisions. After that, the staircase is ascended and descended. Finally, the robot avoids the black box and reaches the goal region.

The second scenario is traversed by a ditch which can only be entered from the left and exited from the right, because the platform in the middle of it is too low to be accessed directly. One solution is shown in Fig. 8.5 (bottom), with the underlying footstep sequence in Fig. 8.4 (bottom). The robot moves appropriately among the different levels, first going down in the ditch and then up towards the goal.

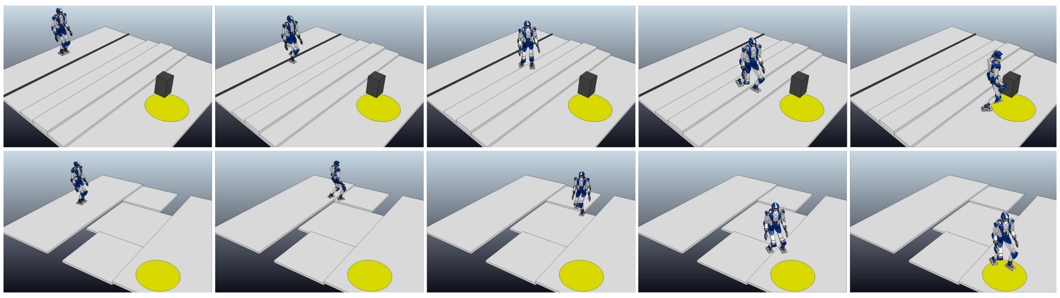


Figure 8.5. Execution of the plans for both simulations.

Chapter 9

IS-MPC on a multi-mass model

In Sect. 3.5 we introduced a multi-mass model, with particular reference to the case in which the masses are two. They are labeled *primary mass*, which accounts for the contribution of the main body of the robot to the total ZMP of the system, and *secondary mass*, which accounts for the contribution of the swinging leg. As already stated, this model gives a more accurate representation of the humanoid robot dynamics than the LIP because it models, at least approximately, the contribution from the rate of change of angular momentum around the CoM, which is normally neglected (see Assumption 2).

In this chapter we describe a variation of the basic IS-MPC scheme which uses the two-mass model in place of the LIP. This will require correctly reformulating the constraints, as well as the automatic footstep placement mechanism (see Ch. 6). The modified scheme will be discussed, starting from the prediction model, the constraints and cost function, as well as the parametrized foot trajectory which is used to compute the ZMP contribution of the secondary mass. Some comparative simulations with the basic scheme will be given in the last section of the chapter.

9.1 Foot trajectory and automatic footstep placement

To compute the contribution of the secondary mass it is necessary to specify its trajectory. Since it models the swinging leg contribution, we assume the trajectory to be that of the swinging foot. There are multiple possibilities for the swinging foot

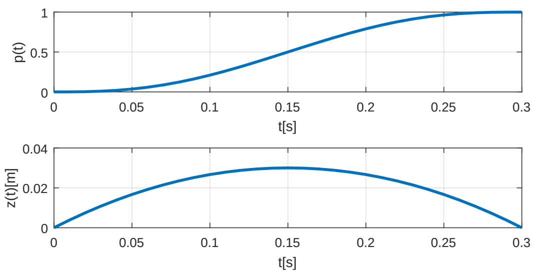


Figure 9.1. The trajectory of the swing foot used in the 2-mass model MPC. $p(t)$ denotes the polynomial shape, the actual trajectory is defined by the starting and ending position of the foot.

trajectory, and we choose to use polynomial expressions. In particular, we adopt a fifth degree polynomial $p(t)$ for the x and y components of the trajectory (the shape of $p(t)$ is shown in Fig 9.1). The swing foot trajectory along x (equivalently along y) is thus expressed parametrically in the footstep positions

$$x_m = x_f^{j-2} + (x_f^j - x_f^{j-2})p(t) \tag{9.1}$$

The z component of the swing foot trajectory is a second degree polynomial (i.e., a parabola)

$$z_m(t) = -\frac{4z_m^{\max}}{t_{ss}^2} t(t - t_{ss}). \tag{9.2}$$

The total ZMP $x_{z,\mathrm{tot}}$ is computed from (3.32), using the above trajectory (9.1) and (9.2)

$$x_{z,m} = x_f^j \left(1 - p + \frac{z_m}{g + \ddot{z}_m}\ddot{p}\right) + x_f^{j-2}\left(p - \frac{z_m}{g + \ddot{z}_m}\ddot{p}\right) \tag{9.3}$$

If the footstep sequence is fixed, this can be computed before the MPC stage as it does not depend on any MPC decision variable. If we incorporate automatic footstep placement (see 6), and make the predicted position of the footsteps a decision variable of the MPC, this is included as part of the prediction model, as it is linear in the footstep positions.

9.2 Multi-mass IS-MPC scheme

This section will discuss the prediction model, constraints, and QP problem employed in IS-MPC for the multi-mass case.

In Sect. 3.5 we showed how the two mass model can be represented as an equivalent LIP (3.5). The variables $x_{z,M}$ and $x_{z,m}$, defined in Sect. 3.5, can be interpreted as the contribution to the ZMP given by each individual mass, with the total ZMP given by a weighted average of the two. If the position of the footsteps is fixed, the ZMP contribution of the secondary mass can be computed, and the ZMP contribution of the primary mass is our degree of freedom. As usual, we extend the system to include the derivative of the ZMP as the input, which makes $\dot{x}_{z,m}$ the decision variable in this formulation.

$$\begin{pmatrix} \dot{x}_M \\ \ddot{x}_M \\ \dot{x}_{z,M} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ \omega_M^2 & 0 & -\omega_M^2 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_M \\ \dot{x}_M \\ x_{z,M} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \dot{x}_{z,M}, \tag{9.4}$$

where, as for the basic scheme, the input is assumed constant over time-steps δ, giving a piecewise linear ZMP for the primary mass.

$$x_{z,M}(t) = x_{z,M}^i + (t - t_i)\dot{x}_{z,M}^i, \quad t \in [t_i; t_{i+1}). \tag{9.5}$$

In the following, the constraints will be discussed. The kinematic constraint is identical to the original (5.3), so it will not be discussed further.

ZMP constraint for multi-mass IS-MPC

The balance condition for the two-mass model requires that the total ZMP of the two-mass systems must be at all times in the support polygon of the robot.

$$-\frac{1}{2}\begin{pmatrix} d_{x,z} \\ d_{y,z} \end{pmatrix} \leq R\left(\theta^j\right)^T \begin{pmatrix} x_{z,\text{tot}}^{k+i} - x_f^j \\ y_{z,\text{tot}}^{k+i} - y_f^j \end{pmatrix} \leq \frac{1}{2}\begin{pmatrix} d_{x,z} \\ d_{y,z} \end{pmatrix} \tag{9.6}$$

where $x_{z,\text{tot}}^{k+i}$ is the predicted total ZMP at the i-th predicted time-step, and it is computed using the prediction model.

Stability constraint for multi-mass IS-MPC

The stability constraint in the case of the multi-mass model is obtained by integrating (4.19) for a piecewise linear $x_{z,M}$ trajectory.

$$\sum_{i=0}^{C-1} e^{-i\eta_M\delta}\dot{x}_{z,M}^{k+i} = -\sum_{i=C}^{\infty} e^{-i\eta_M\delta}\dot{x}_{z,M}^{k+i} + \frac{\eta_M}{1-e^{-\eta_M\delta}}(x_u^k - x_{z,M}^k). \tag{9.7}$$

This constraint has the same form and plays the same role of the stability constraint in standard IS-MPC, and the infinite sum on the right-hand side can be substituted using any of the tail presented in Sect. 5.4.1.

QP problem

Having defined the constraints, the IS-MPC scheme with multi-mass model is formulated as

$$\left\{\begin{array}{l} \displaystyle\sum_{i=0}^{C-1}\left(\left(\dot{x}_{z,M}^{k+i}\right)^2 + k_\text{v}^x\left(\dot{x}_M^{k+i+1} - v_x\right)^2 + \left(\dot{y}_{z,M}^{k+i}\right)^2 + k_\text{v}^y\left(\dot{y}_M^{k+i+1} - v_y\right)^2\right) \\ \qquad\qquad\qquad\text{subject to:} \\ \bullet\ \text{the ZMP constraint (9.6) on } x_{z,\text{tot}}, \\ \bullet\ \text{the stability constraint (9.7),} \\ \bullet\ \text{the kinematic constraint (5.3).} \end{array}\right.$$

This particular formulation includes a term in the cost function that penalizes deviations from a reference velocity (v_x, v_y), but it might be reformulated as in Ch. 6 to include a candidate footstep planner and penalize deviations from candidate footsteps.

9.3 Simulations

Simulations using this technique show an improvement in the accuracy of the ZMP predicted by the MPC. The simulations are performed using the Dynamic Animations and Robotic Toolkit (DART), on a NAO humanoid robot. In all simulations every step lasts 0.5 s with single and double support phases respectively of 0.3 s and 0.2 s.

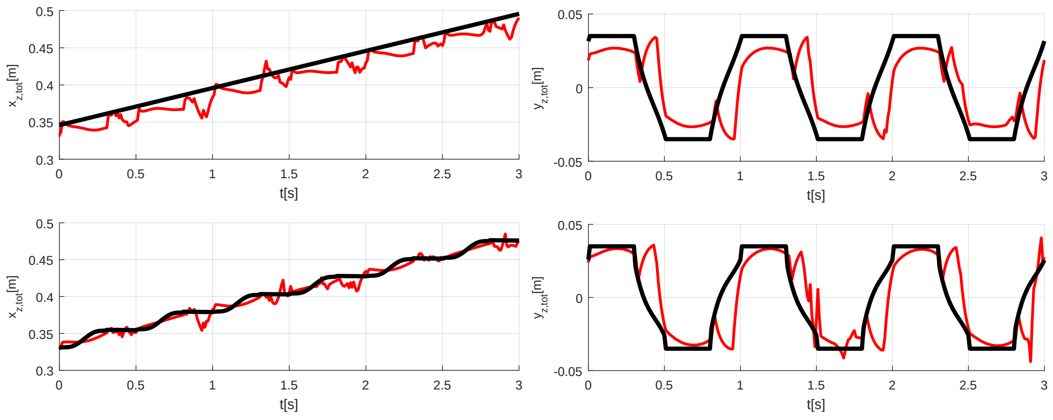


Figure 9.2. A comparison between standard IS-MPC (top) and IS-MPC using a 2-mass model (bottom). The black trajectory is the nominal ZMP as computed by the MPC, and the red trajectory is the ZMP measured in a dynamic simulator.

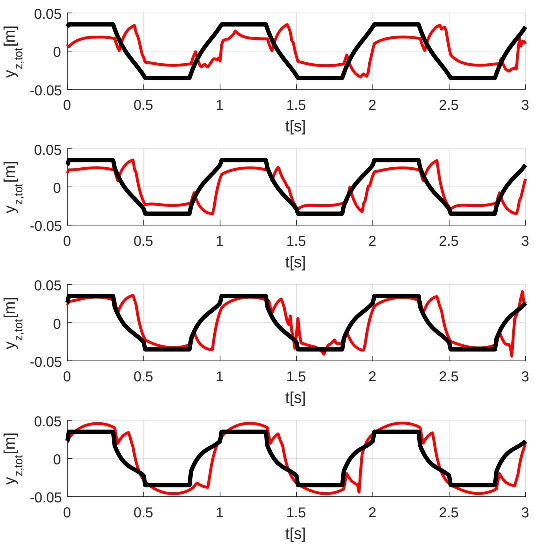


Figure 9.3. IS-MPC using a 2-mass model (y-component only). Simulations using different values for the secondary mass. From the top: 0%, 15%, 30%, 45% of the total mass.

The balance constraints are squares with $d_{x,z} = d_{y,z} = 0.03$ m, while the size of the feasibility constraints is $d_{x,f} = 0.05$ m, $d_{y,f} = 0.025$ m, $\ell = 0.125$ m. The control horizon is $T_c = 1$ s (i.e. 2 steps), and the sampling time δ is 0.01 s. The gains for the cost function are $k_{\mathrm{v}}^{x} = k_{\mathrm{v}}^{y} = 10$. The periodic tail is used in the stability constraint (see Sect. 5.4.1).

In Fig. 9.2 IS-MPC using the 2-mass model is compared to the basic IS-MPC with automatic footstep placement, which employs the LIP model. The secondary mass m accounts for 30% of the total mass. In the first case the measured ZMP is visibly different from the predicted one. The 2-mass MPC provides a better result, as the measured ZMP is closer to the predicted one during the single support phase, and the error is reduced by 35%. We do not see a significant improvement in the double support phase, which is also due to the fact that the dynamic simulator is less reliable for measuring the ZMP in a situation with multiple contact surfaces, as it produces several false values.

Figure. 9.3 shows a comparison between four different values of the mass m, from 0% to 45% with an increase of 15% at each plot. The best fitting during the single support occurs with a value of 30% which has also been used for the previous results. Moreover it is also evident that as we continue to increase the mass m, the performance get worse as clearly shown in the 45% plot.

It is important to notice that the 0% case considers the humanoid CoM coinciding with the mass M which roughly represents the torso while the first plots in Fig. 9.2 represent the LIP-based MPC case where the reference is generated for the true humanoid CoM, not the torso. Moreover, by controlling M and requiring that its height remains constant, it is also evident that the true humanoid CoM will have a variable height.

Chapter 10

Experiments

Experimental validation of IS-MPC was performed on two platforms, i.e., the NAO and HRP-4 humanoid robots.

On NAO, the scheme is implemented as a custom module in the B-Human RoboCup SPL team framework. It runs in real-time on the on-board CPU at a control frequency of 100 Hz ($\delta = 0.01$ s). Footstep timing is determined using rule (5.1) with $\bar{L}_s = 0.075$ m, $\bar{T}_s = 0.5$ s, $\bar{v} = 0.15$ m/s as cruise parameters, and $\alpha = 0.1$ m/s (as in Fig. 5.3). The scheme uses the automatic footstep placement of Ch. 6, and candidate footsteps are generated as explained in Sect. 5.1.2, with $\theta_{\max} = \pi/8$ rad and $\ell = 0.1$ m. In the IS-MPC module we have set $T_c = 1.0$ s and $\bar{z}_c = 0.23$ m. The dimensions of the ZMP admissible region are $d_{z,x} = d_{z,y} = 0.03$ m, while those of the kinematically admissible region are $d_{k,x} = 0.1$ m, $d_{k,y} = 0.05$ m. The weight in the QP cost function is $\beta = 10^4$. The anticipative tail is used with a preview horizon $T_p = 2.0$ s.

The software architecture of HRP-4 requires control commands to be generated at a frequency of 200 Hz ($\delta = 0.005$ s). Gait generation runs on an external laptop PC and joint motion commands are sent to the robot via Ethernet using TCP/IP. Automatic footstep placement is in use, and the footstep timing is determined using rule (5.1) with $\bar{L}_s = 0.12$ m, $\bar{T}_s = 0.8$ s, $\bar{v} = 0.15$ m/s as cruise parameters, and $\alpha = 0.1$ m/s (as in Fig. 5.3). Each generated T_s is split into T_{ss} (single support) and T_{ds} (double support) using a 60%-40% distribution. Candidate footsteps are generated as explained in Sect. 5.1.2, with $\theta_{\max} = \pi/8$ rad and $\ell = 0.18$ m. In the IS-MPC module, which uses a control horizon T_c of 1.6 s, we have set $\bar{z}_c = 0.78$ m. The dimensions of the ZMP admissible region are $d_{z,x} = d_{z,y} = 0.01$ m, while those of the kinematically admissible region are $d_{k,x} = 0.3$ m, $d_{k,y} = 0.07$ m. The weight in the QP cost function is $\beta = 10^4$. The anticipative tail is used in the stability constraint with a preview $T_p = 3.2$ s.

Before presenting complete locomotion experiments, Fig. 10.1 reports some data from a typical forward gait of HRP-4. In particular, the plot shows the nominal ZMP trajectory, as generated by IS-MPC, together with the ZMP measurements reconstructed from the force-torque sensors at the robot ankles [68]. Note that these experiments do not employ any form of robustness like in the scheme described in Ch. 7, but for increased safety the size of the ZMP admissible region was reduced (by a constant value, not progressively along the prediction) from that of the corresponding

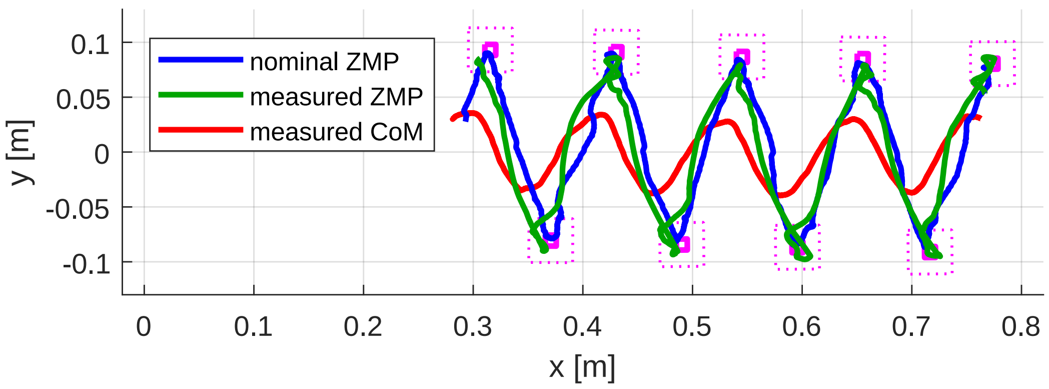


Figure 10.1. Nominal ZMP, measured ZMP and measured CoM along a forward gait of HRP-4. Note the restricted ZMP regions (magenta, solid) and the original ZMP regions used in the simulations (magenta, dotted).

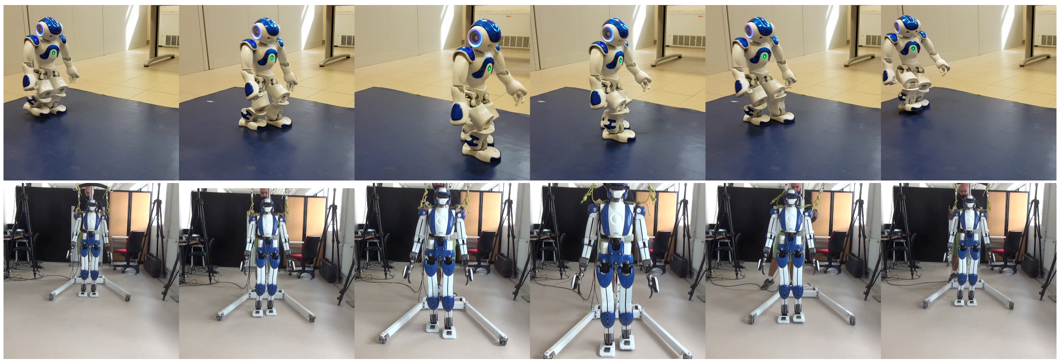


Figure 10.2. Experiments on NAO and HRP-4 with automatic footstep placement. The robots are required to walk forward and then backwards.

simulations (see Sect.6.4). It is possible to notice in Fig. 10.1 how the restriction is effective, in the sense that while the measured ZMP violates the constraints, it stays well within the original ZMP admissible region used in the simulation.

In the first experiment, the robots are required to perform a forward-backward motion as shown in Fig. 10.2. The reference velocities are $v_x = \pm 0.15$ m/s for the NAO and $v_x = \pm 0.2$ m/s for the HRP-4.

In the second experiment, which is shown in Fig. 10.3, the robots are given reference velocities aimed at performing an L-shaped motion. In particular, we have $v_x = 0.15$ m/s followed by $v_y = 0.05$ m/s for the NAO, and $v_x = 0.2$ m/s followed by $v_y = 0.2$ for the HRP-4.

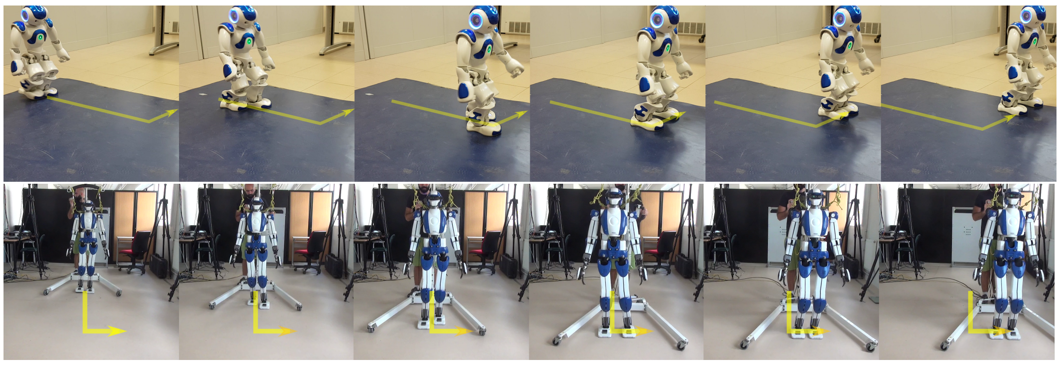


Figure 10.3. Experiments on NAO and HRP-4 with automatic footstep placement. The robots are required to walk in an L-shape.

Chapter 11

Conclusions

This thesis has presented IS-MPC, an MPC scheme for humanoid gait generation with an embedded stability constraint. After introducing the dynamic considerations leading to the LIP model in Ch. 3, the stability condition was presented in Ch. 4 as a relation between the current state of the LIP and the future ZMP trajectory. In Ch. 5 IS-MPC was described along with the stability constraint, and the properties of recursive feasibility and stability that derive from it were proved and discussed.

Several extensions and additions to the basic scheme were presented: Ch. 6 introduced automatic footstep placement, which allows for reactive stepping whenever this is necessary to maintain balance; Ch. 7 discussed robust gait generation, proving conditions for stability and feasibility under the presence of perturbations; Ch. 8 was concerned walking on non-flat ground, using a 3D linear model related to the LIP and discussing the linearization of the nonlinear constraint; Ch. 9 presented a version of IS-MPC which uses a multi-mass model for increased accuracy in the prediction of the ZMP.

The dynamic simulations presented throughout the work, along with the experiments discussed in Ch. 10, have demonstrated the effectiveness and applicability of the proposed techniques. Two different robotic platforms were used: the small-scale humanoid robot NAO by SoftBank Robotics, and the human-size robot HRP-4 by Kawada Robotics.

The proposed techniques open up the way to further research. Among the possible future developments are:

- experimental validation of the developments beyond the IS-MPC basic scheme
- an analysis of the effect on stability and feasibility of the introduction of the disturbance observer, in the robust IS-MPC
- an improvement of the footstep planner on uneven ground to allow for online replanning

Appendix A

Integration of the LIP model

The prediction model of MPC, as discussed in Sect. 5.2 is a dynamic extension of the LIP, that has the ZMP derivative as input. The associated discrete-time model is derived by integrating the LIP equation over the timestep δ, using the piecewise linear trajectory (5.6), which reduces to a linear trajectory over a single timestep. Throughout the following, we will make use of the hyperbolic functions $\sinh x = (e^x - e^{-x})/2$ and $\cosh x = (e^x + e^{-x})/2$.

First, recall the general solution at time t_{k+1} of the decoupled system (3.12) for a generic input $x_z(t)$, starting from time t_k

$$\begin{aligned} x_u^{k+1} &= x_u^k e^{\eta\delta} - \eta \int_{t_k}^{t_{k+1}} e^{\eta(t_{k+1}-\tau)} x_z(\tau) d\tau \\ x_s^{k+1} &= x_s^k e^{-\eta\delta} + \eta \int_{t_k}^{t_{k+1}} e^{-\eta(t_{k+1}-\tau)} x_z(\tau) d\tau. \end{aligned}$$

From the change of coordinates (3.11) we recall that $x_u + x_s = 2x_c$. We can then write

$$2x_c^{k+1} = x_u^k e^{\eta\delta} - \eta \int_{t_k}^{t_{k+1}} e^{\eta(t_{k+1}-\tau)} x_z(\tau) d\tau + x_s^k e^{-\eta\delta} + \eta \int_{t_k}^{t_{k+1}} e^{-\eta(t_{k+1}-\tau)} x_z(\tau) d\tau =$$
$$\left(x_c^k + \frac{\dot{x}_c^k}{\eta} \right) e^{\eta\delta} + \left(x_c^k - \frac{\dot{x}_c^k}{\eta} \right) e^{-\eta\delta} - \eta \int_{t_k}^{t_{k+1}} \left(e^{\eta(t_{k+1}-\tau)} - e^{-\eta(t_{k+1}-\tau)} \right) x_z(\tau) d\tau$$

which leads the CoM position at time t_k, when the system is subject to a generic input $x_z(t)$

$$x_c^{k+1} = x_c^k \cosh(\eta\delta) + \frac{\dot{x}_c^k}{\eta} \sinh(\eta\delta) - \eta \int_{t_k}^{t_{k+1}} \sinh\left(\eta(t_{k+1}-\tau)\right) x_z(\tau) d\tau$$

In our case we are interested in the linear trajectory $x_z(\tau) = x_z^k + \dot{x}_z^k(\tau - t_k)$

$$x_c^{k+1} = x_c^k \cosh(\eta\delta) + \frac{\dot{x}_c^k}{\eta} \sinh(\eta\delta) - \eta \int_{t_k}^{t_{k+1}} \sinh\left(\eta(t_{k+1}-\tau)\right) (x_z^k + \dot{x}_z^k(\tau - t_k)) d\tau$$

The last integral can be split in two and solved by substituting $s = \tau - t_k$. The first part is readily integrated

$$-x_z^k \eta \int_0^\delta \sinh\left(\eta(\delta - s)\right) ds = x_z^k \left[\cosh\left(\eta(\delta - s)\right)\right]_0^\delta = (1 - \cosh(\eta\delta)) x_z^k$$

for the remaining half, integrate by parts

$$-\dot{x}_z^k\eta\int_0^\delta \sinh\left(\eta(\delta-s)\right)sds = \dot{x}_z^k\left(\left[\cosh\left(\eta(\delta-s)\right)s\right]_0^\delta - \int_0^\delta \cosh\left(\eta(\delta-s)\right)ds\right) =$$

$$\dot{x}_z^k\left(\left[\cosh\left(\eta(\delta-s)\right)s\right]_0^\delta + \frac{1}{\eta}\left[\sinh\left(\eta(\delta-s)\right)\right]_0^\delta\right) = \left(\delta - \frac{1}{\eta}\sinh(\eta\delta)\right)\dot{x}_z^k$$

Substituting back in the initial expression gives the expression for the CoM position at time t_{k+1}, in terms of the state and input at time t_k

$$x_c^{k+1} = x_c^k\cosh(\eta\delta) + \frac{\dot{x}_c^k}{\eta}\sinh(\eta\delta) + (1-\cosh(\eta\delta))x_z^k + \left(\delta + \frac{1}{\eta}\sinh(\eta\delta)\right)\dot{x}_z^k. \quad \text{(A.1)}$$

The next step is to compute the expression of the CoM velocity at time t_{k+1}. The procedure is similar but now we use $2\dot{x}_c/\eta = x_u - x_s$

$$2\frac{\dot{x}_c^{k+1}}{\eta} = x_u^k e^{\eta\delta} - \eta\int_{t_k}^{t_{k+1}} e^{\eta(t_{k+1}-\tau)}x_z(\tau)d\tau - x_s^k e^{-\eta\delta} - \eta\int_{t_k}^{t_{k+1}} e^{-\eta(t_{k+1}-\tau)}x_z(\tau)d\tau$$

$$= \left(x_c^k + \frac{\dot{x}_c^k}{\eta}\right)e^{\eta\delta} - \left(x_c^k - \frac{\dot{x}_c^k}{\eta}\right)e^{-\eta\delta} - \eta\int_{t_k}^{t_{k+1}}\left(e^{\eta(t_{k+1}-\tau)} + e^{-\eta(t_{k+1}-\tau)}\right)x_z(\tau)d\tau$$

which leads to

$$\dot{x}_c^{k+1} = x_c^k\eta\sinh(\eta\delta) + \dot{x}_c^k\cosh(\eta\delta) - \eta^2\int_{t_k}^{t_{k+1}}\cosh\left(\eta(t_{k+1}-\tau)\right)x_z(\tau)d\tau.$$

Specializing this equation for a linear ZMP trajectory gives

$$\dot{x}_c^{k+1} = x_c^k\eta\cosh(\eta\delta) + \dot{x}_c^k\sinh(\eta\delta) - \eta^2\int_{t_k}^{t_{k+1}}\cosh\left(\eta(t_{k+1}-\tau)\right)(x_z^k + \dot{x}_z^k(\tau-t_k))d\tau$$

where the last integral is computed by splitting the two terms and substituting $s = \tau - t_k$. For the first half

$$-x_z^k\eta^2\int_0^\delta \cosh\left(\eta(\delta-s)\right)ds = x_z^k\eta\left[\sinh\left(\eta(\delta-s)\right)\right]_0^\delta = -\eta\sinh(\eta\delta))x_z^k$$

and for the second half, integrate by parts

$$-\dot{x}_z^k\eta^2\int_0^\delta \cosh\left(\eta(\delta-s)\right)sds = \dot{x}_z^k\left(\eta\left[\sinh\left(\eta(\delta-s)\right)s\right]_0^\delta - \eta\int_0^\delta \sinh\left(\eta(\delta-s)\right)ds\right)$$

$$= \dot{x}_z^k\eta\left(\left[\sinh\left(\eta(\delta-s)\right)s\right]_0^\delta + \left[\cosh\left(\eta(\delta-s)\right)\right]_0^\delta\right) = \left(1-\cosh(\eta\delta)\right)\dot{x}_z^k.$$

Substituting back in the initial expression gives the expression for the CoM velocity at time t_{k+1}, in terms of the state and input at time t_k

$$\dot{x}_c^{k+1} = x_c^k\eta\sin(\eta\delta) + \dot{x}_c^k\cosh(\eta\delta) - \eta\sinh(\eta\delta))x_z^k + \left(1-\cosh(\eta\delta)\right)\dot{x}_z^k. \quad \text{(A.2)}$$

Finally, the ZMP position at time t_{k+1} is given by simple integration of the linear trajectory

$$x_z^{k+1} = x_z^k + \delta\dot{x}_z^k. \quad \text{(A.3)}$$

Grouping (A.1), (A.2) and (A.3) into a single equation results in the discrete-time model (5.7).

Bibliography

[1] S. Behnke, “Humanoid robots-from fiction to reality?” *KI*, vol. 22, no. 4, pp. 5–9, 2008.

[2] A. Kheddar, S. Caron, P. Gergondet, A. Comport, A. Tanguy, C. Ott, B. Henze, G. Mesesan, J. Englsberger, M. Roa *et al.*, “Humanoid robots in aircraft manufacturing,” *Robotics and Automation Magazine*, 2019.

[3] N. Scianca, M. Cognetti, D. De Simone, L. Lanari, and G. Oriolo, “Intrinsically stable MPC for humanoid gait generation,” in *16th IEEE-RAS Int. Conf. on Humanoid Robots*, 2016, pp. 101–108.

[4] N. Scianca, D. De Simone, L. Lanari, and G. Oriolo, “MPC for humanoid gait generation: Stability and feasibility,” 2019. [Online]. Available: https://arxiv.org/abs/1901.08505

[5] A. Aboudonia, N. Scianca, D. De Simone, L. Lanari, and G. Oriolo, “Humanoid gait generation for walk-to locomotion using single-stage MPC,” in *17th IEEE-RAS Int. Conf. on Humanoid Robots*, 2017, pp. 178–183.

[6] F. M. Smaldone, N. Scianca, V. Modugno, L. Lanari, and G. Oriolo, “Gait generation using intrinsically stable MPC in the presence of persistent disturbances,” in *2019 IEEE-RAS Int. Conf. on Humanoid Robots*, 2019.

[7] A. Zamparelli, N. Scianca, L. Lanari, and G. Oriolo, “Humanoid gait generation on uneven ground using intrinsically stable MPC,” *IFAC-PapersOnLine*, vol. 51, pp. 393–398, 2018.

[8] P. Ferrari, N. Scianca, L. Lanari, and G. Oriolo, “An integrated motion planner/controller for humanoid robots on uneven ground,” in *2019 European Control Conf.*, 2019.

[9] N. Scianca, V. Modugno, L. Lanari, and G. Oriolo, “Gait generation via intrinsically stable MPC for a multi-mass humanoid model,” in *2017 IEEE-RAS Int. Conf. on Humanoid Robots*, 2017, pp. 547–552.

[10] D. De Simone, N. Scianca, P. Ferrari, L. Lanari, and G. Oriolo, “MPC-based humanoid pursuit-evasion in the presence of obstacles,” in *2017 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2017.

[11] M. Vukobratović and D. Juričić, “Contribution to the synthesis of biped gait,” *IEEE Trans. on Biomedical Engineering*, no. 1, pp. 1–6, 1969.

[12] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3D linear inverted pendulum mode: A simple modeling for a biped walking pattern generation," in *2001 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 1. IEEE, 2001, pp. 239–246.

[13] J. Pratt, J. Carff, S. Drakunov, and A. Goswami, "Capture point: A step toward humanoid push recovery," in *2006 6th IEEE-RAS international conference on humanoid robots*, 2006, pp. 200–207.

[14] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," in *2003 IEEE Int. Conf. on Robotics and Automation*, 2003, pp. 1620–1626.

[15] P.-B. Wieber, "Trajectory free linear model predictive control for stable walking in the presence of strong perturbations," in *6th IEEE-RAS Int. Conf. on Humanoid Robots*, 2006, pp. 137–142.

[16] B. Henze, C. Ott, and M. A. Roa, "Posture and balance control for humanoid robots in multi-contact scenarios based on model predictive control," in *2014 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. IEEE, 2014, pp. 3253–3258.

[17] A. Sherikov, D. Dimitrov, and P. B. Wieber, "Whole body motion controller with long-term balance constraints," in *14th IEEE-RAS Int. Conf. on Humanoid Robots*, 2014, pp. 444–450.

[18] T. Koolen, T. De Boer, J. Rebula, A. Goswami, and J. Pratt, "Capturability-based analysis and control of legged locomotion, part 1: Theory and application to three simple gait models," *The International Journal of Robotics Research*, vol. 31, no. 9, pp. 1094–1113, 2012.

[19] T. Sugihara and T. Yamamoto, "Foot-guided agile control of a biped robot through zmp manipulation," in *2017 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2017, pp. 4546–4551.

[20] J. Carpentier, R. Budhiraja, and N. Mansard, "Learning feasibility constraints for multi-contact locomotion of legged robots," in *Robotics: Science and Systems*, 2017, p. 9p.

[21] E. C. Kerrigan, "Robust constraint satisfaction: Invariant sets and predictive control," Ph.D. dissertation, University of Cambridge, 2001.

[22] M. Ciocca, P.-B. Wieber, and T. Fraichard, "Strong recursive feasibility in model predictive control of biped walking," in *2017 IEEE-RAS Int. Conf. on Humanoid Robots*. IEEE, 2017, pp. 730–735.

[23] A. Sherikov, "Balance preservation and task prioritization in whole body motion control of humanoid robots," Ph.D. dissertation, Grenoble Alpes, 2016.

[24] M. Krause, J. Englsberger, P.-B. Wieber, and C. Ott, "Stabilization of the capture point dynamics for bipedal walking based on model predictive control," *IFAC Proceedings Volumes*, vol. 45, no. 22, pp. 165–171, 2012.

[25] M. Naveau, M. Kudruss, O. Stasse, C. Kirches, K. Mombaur, and P. Souères, "A reactive walking pattern generator based on nonlinear model predictive control," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 10–17, 2016.

[26] R. J. Griffin and A. Leonessa, "Model predictive control for dynamic footstep adjustment using the divergent component of motion," in *2016 IEEE Int. Conf. on Robotics and Automation.* IEEE, 2016, pp. 1763–1768.

[27] S. Faraji, S. Pouya, C. G. Atkeson, and A. J. Ijspeert, "Versatile and robust 3D walking with a simulated humanoid robot (Atlas): A model predictive control approach," in *2014 IEEE Int. Conf. on Robotics and Automation*, 2014, pp. 1943–1950.

[28] J. Alcaraz-Jiménez, D. Herrero-Pérez, and H. Martínez-Barberá, "Robust feedback control of ZMP-based gait for the humanoid robot Nao," *The Int. Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1074–1088, 2013.

[29] H. Diedam, D. Dimitrov, P.-B. Wieber, K. Mombaur, and M. Diehl, "Online walking gait generation with adaptive foot positioning through linear model predictive control," in *2008 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems.* IEEE, 2008, pp. 1121–1126.

[30] A. Herdt, H. Diedam, P.-B. Wieber, D. Dimitrov, K. Mombaur, and M. Diehl, "Online walking motion generation with automatic footstep placement," *Advanced Robotics*, vol. 24, no. 5-6, pp. 719–737, 2010.

[31] A. Herdt, N. Perrin, and P. B. Wieber, "Walking without thinking about it," in *2010 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2010, pp. 190–195.

[32] L. Chisci, J. A. Rossiter, and G. Zappa, "Systems with persistent disturbances: Predictive control with restricted constraints," *Automatica*, vol. 37, no. 7, pp. 1019–1028, Jul. 2001.

[33] D. Mayne, M. Seron, and S. V. Raković, "Robust model predictive control of constrained linear system with bounded disturbances," *Automatica*, vol. 41, pp. 219–224, 2005.

[34] G. Pannocchia, J. B. Rawlings, and S. J. Wright, "Conditions under which suboptimal nonlinear MPC is inherently robust," *Systems & Control Letters*, vol. 60, pp. 747–755, 2011.

[35] N. A. Villa and P. Wieber, "Model predictive control of biped walking with bounded uncertainties," in *17th IEEE-RAS Int. Conf. on Humanoid Robots*, 2017, pp. 836–841.

[36] K. Kaneko, F. Kanehiro, M. Morisawa, E. Yoshida, and J. Laumond, “Disturbance observer that estimates external force acting on humanoid robots,” in *12th IEEE Int. Work. on Advanced Motion Control*, 2012, pp. 1–6.

[37] S. Czarnetzki, S. Kerner, and O. Urbann, “Observer-based dynamic walking control for biped robots,” *Robotics and Autonomous Systems*, vol. 57, no. 8, pp. 839–845, 2009.

[38] R. J. Griffin, A. Leonessa, and A. Asbeck, “Disturbance compensation and step optimization for push recovery,” in *2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2016, pp. 5385–5390.

[39] J. Englsberger, G. Mesesan, and C. Ott, “Smooth trajectory generation and push-recovery based on divergent component of motion,” in *2017 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2017, pp. 4560–4567.

[40] A. Pajon and P.-B. Wieber, “Safe 3d bipedal walking through linear mpc with 3d capturability,” in *2019 IEEE Int. Conf. on Robotics and Automation*, 2019, pp. 1404–1409.

[41] S. Caron and A. Kheddar, “Dynamic walking over rough terrains by nonlinear predictive control of the floating-base inverted pendulum,” in *2017 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems.* IEEE, 2017, pp. 5017–5024.

[42] S. Caron, A. Escande, L. Lanari, and B. Mallein, “Capturabilitybased analysis, optimization and control of 3d bipedal walking,” in *2018IEEE Int. Conf. on Robotics and Automation*, 2018.

[43] A. Herdt, “Model predictive control of a humanoid robot,” Ph.D. dissertation, Ecole Nationale Supérieure des Mines de Paris, 2012.

[44] M. A. Hopkins, D. W. Hong, and A. Leonessa, “Humanoid locomotion on uneven terrain using the time-varying divergent component of motion,” in *14th IEEE-RAS Int. Conf. on Humanoid Robots*, 2014, pp. 266–272.

[45] T. Kamioka, T. Watabe, M. Kanazawa, H. Kaneko, and T. Yoshike, “Dynamic gait transition between bipedal and quadrupedal locomotion,” in *2015 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2015, pp. 2195–2201.

[46] K. Terada and Y. Kuniyoshi, “Online gait planning with dynamical 3d-symmetrization method,” in *7th IEEE-RAS Int. Conf. on Humanoid Robots*, 2007, pp. 222–227.

[47] S. C. Luo, P. H. Chang, J. Sheng, S. C. Gu, and C. H. Chen, “Arbitrary biped robot foot gaiting based on variate com height,” in *13th IEEE-RAS Int. Conf. on Humanoid Robots*, 2013, pp. 534–539.

[48] J. Englsberger, C. Ott, and A. Albu-Schäffer, “Three-dimensional bipedal walking control based on divergent component of motion,” *IEEE Transactions on Robotics*, vol. 31, no. 2, pp. 355–368, 2015.

[49] P. Sardain and G. Bessonnet, “Forces acting on a biped robot. center of pressure-zero moment point,” *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 34, no. 5, pp. 630–637, 2004.

[50] D. E. Orin, A. Goswami, and S.-H. Lee, “Centroidal dynamics of a humanoid robot,” *Autonomous Robots*, vol. 35, no. 2-3, pp. 161–176, 2013.

[51] L. Lanari, S. Hutchinson, and L. Marchionni, “Boundedness issues in planning of locomotion trajectories for biped robots,” in *2014 IEEE-RAS Int. Conf. on Humanoid Robots*, 2014, pp. 951–958.

[52] A. L. Hof, “The ‘extrapolated center of mass’ concept suggests a simple control of balance in walking,” *Human movement science*, vol. 27, no. 1, pp. 112–125, 2008.

[53] T. Takenaka, T. Matsumoto, and T. Yoshiike, “Real time motion generation and control for biped robot-1 st report: Walking gait pattern generation,” in *2009 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems.* IEEE, 2009, pp. 1084–1091.

[54] S. Caron and A. Kheddar, “Dynamic walking over rough terrains by nonlinear predictive control of the floating-base inverted pendulum,” in *2017 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems.* IEEE, 2017, pp. 5017–5024.

[55] S. Kajita, H. Hirukawa, K. Harada, and K. Yokoi, *Introduction to Humanoid Robotics.* Springer Publishing Company Inc., 2014.

[56] L. Lanari and S. Hutchinson, “Inversion-based gait generation for humanoid robots,” in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2015, pp. 1592–1598.

[57] ——, “Planning desired center of mass and zero moment point trajectories for bipedal locomotion,” in *15th IEEE-RAS Int. Conf. on Humanoid Robots*, 2015, pp. 637–642.

[58] L. Lanari and J. T. Wen, “Feedforward calculation in tracking control of flexible robots,” in *[1991] Proceedings of the 30th IEEE Conference on Decision and Control*, 1991, pp. 1403–1408.

[59] S. Devasia, D. Chen, and B. Paden, “Nonlinear inversion-based output tracking,” *IEEE Transactions on Automatic Control*, vol. 41, no. 7, pp. 930–942, 1996.

[60] M. Cognetti, D. De Simone, F. Patota, N. Scianca, L. Lanari, and G. Oriolo, “Real-time pursuit-evasion with humanoid robots,” in *2017 IEEE Int. Conf. on Robotics and Automation.* IEEE, 2017, pp. 4090–4095.

[61] D. Dimitrov, A. Paolillo, and P.-B. Wieber, “Walking motion generation with online foot position adaptation based on ℓ_1- and ℓ_∞-norm penalty formulations,” in *2011 IEEE Int. Conf. on Robotics and Automation*, 2011, pp. 3523–3529.

[62] E. C. Kerrigan, “Robust constraint satisfaction: Invariant sets and predictive control,” Ph.D. dissertation, Department of Engineering - University of Cambridge, 2000.

[63] S. Raković and D. Mayne, “A simple tube controller for efficient robust model predictive control of constrained linear discrete time systems subject to bounded disturbances,” *IFAC-PapersOnLine*, vol. 16, 01 2005.

[64] D. Limon, I. Alvarado, and T. Alamo, “Robust tube-based mpc for tracking of constrained linear systems with additive disturbances,” *Journal of Process Control*, vol. 20, pp. 248–260, 03 2010.

[65] N. Motoi, M. Ikebe, and K. Ohnishi, “Real-time gait planning for pushing motion of humanoid robot,” *IEEE Transactions on Industrial Informatics*, vol. 3, no. 2, pp. 154–163, May 2007.

[66] L. Hawley and W. Suleiman, “External force observer for medium-sized humanoid robots,” in *2016 IEEE-RAS Int. Conf. on Humanoid Robots*, Nov 2016, pp. 366–371.

[67] W. Burgard, M. Hebert, and M. Bennewitz, “World modeling,” in *Springer handbook of robotics.* Springer, 2016, pp. 1135–1152.

[68] A. Tanguy, D. De Simone, A. I. Comport, G. Oriolo, and A. Kheddar, “Closed-loop mpc with dense visual slam-stability through reactive stepping,” in *2019 IEEE Int. Conf. on Robotics and Automation*, 2019, pp. 1397–1403.