SAPIENZA
UNIVERSITÀ DI ROMA

Faculty of Information Engineering
Computer Science and Statistics

Dottorato in
Automatica, Bioingegneria e Ricerca Operativa

Ph.D. Thesis

A Framework for Safe Human-Humanoid Coexistence

Academic Advisor
Prof. Giuseppe Oriolo

Candidate
Daniele De Simone

Ciclo XXXI

Abstract

This work is focused on the development of a safety framework for Human-Humanoid coexistence, with emphasis on humanoid locomotion. After a brief introduction to the fundamental concepts of humanoid locomotion, the two most common approaches for gait generation are presented, and are extended with the inclusion of a stability condition to guarantee the boundedness of the generated trajectories. Then the safety framework is presented, with the introduction of different *safety behaviors*. These behaviors are meant to enhance the overall level of safety during any robot operation. *Proactive* behaviors will enhance or adapt the current robot operations to reduce the risk of danger, while *override* behaviors will stop the current robot activity in order to take action against a particularly dangerous situation. A state machine is defined to control the transitions between the behaviors. The behaviors that are strictly related to locomotion are subsequently detailed, and an implementation is proposed and validated. A possible implementation of the remaining behaviors is proposed through the review of related works that can be found in literature.

Contents

Chapter 1

Introduction

1.1 Motivation

Since the introduction of robotic manipulators in factories, robots have started to substitute humans in performing dangerous, fatiguing or repetitive tasks. Robotics research has focused more and more on this aspect, to develop autonomous machineries able to speed-up production activities in factories, and to provide a better quality of life to human workers. However, those robots were heavy, bulky and it was not safe to let them operate in the presence of humans.

Fig. 1.1: Industrial robots in a factory

Therefore, in recent days, the research focused on the development of softer, lighter robots, with the clear idea of robots and human beings working together in the same environment. This started a new trend, usually referred to as Safe Human-Robot coexistence, trying to bring robots outside of their security cages, and finally outside the factories towards domestic environments.

Fig. 1.2: Lightweight manipulator

Unlike before, allowing robots in the same environment of humans to cooperate, required an higher level of mobility, achieved with the development of mobile manipulators (e.g. KUKA youBot, Robotnik RB1, PAL TIAGo). This new category of robots introduced a new set of problems for researchers to be solved, like perception of the environment, localization, and clearly safe interactions with humans. With the passing years, the design of mobile manipulators started evolving, with the goal to resemble humans in their appearance and trying to obtain capabilities comparable to those of a human being (e.g. Softbank Pepper, PAL Reem).

However a wheeled robot is necessarily limited from a locomotion point of view. There is no doubt that the capability of a person to take steps to overcome obstacles or to climb stairs cannot be replicated with wheels. The creation of legged humanoid robots is one of the most interesting and difficult challenges that researchers have ever faced. Ideally, a humanoid robot is extremely versatile, being able to walk in complex environments that include stairs, uneven terrain and obstacles exactly like humans do. Their complex, highly redundant structure allows humanoid robots to complete articulated tasks, like dual arm manipulation and multi-contact motions.

The state of the art is, unfortunately, quite far from the ideal situation. Humanoid robots are extremely complex, unstable, systems that require a huge effort to be controlled. If the goal is the deployment of humanoid robots in industrial or domestic environments to coexist with human beings, research must focus on how to make safe such coexistence.

Researchers are currently focusing on the main issue of humanoid robotics: locomotion. Moving in the environment by taking steps requires a constant control of the robot balance, given the unstable nature of the system and the complexity of the robot structure.

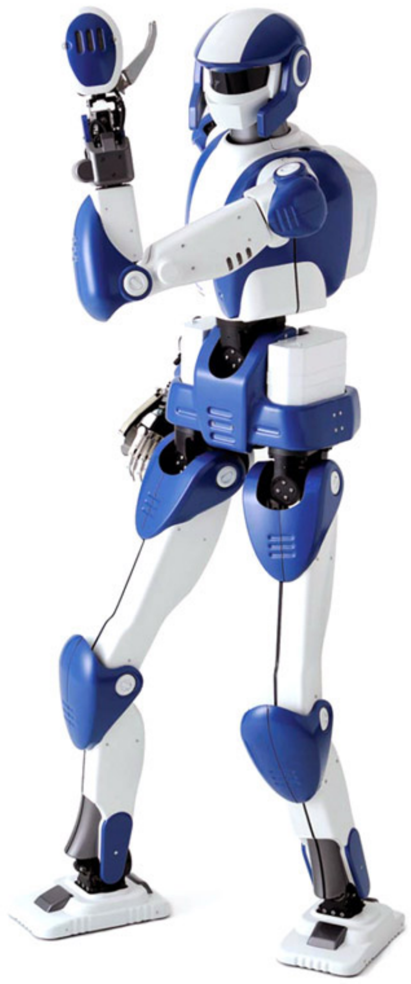


Fig. 1.3: HRP-4 Humanoid robot

This thesis is set in between this two problems: the issue of humanoid locomotion and balance control, and the guarantee of a safe shared environment for humans and robots.

1.2 Contributions

- The primary contribution of this thesis is the formulation of a novel framework for safe human-humanoid coexistence. The framework provides general rules that should be followed when a human and a robot share their workspace to avoid any dangerous situation and keep the workspace safe. The proposed guidelines are taken into account in the motion planning and control phase, and translated into a set of behaviors that allow the robot to perform its task while keeping high the level of safety for its co-workers and the environment. In particular, the emphasis will be on those behaviors that are directly related to locomotion since while walking the humanoid is most likely to endanger the surroundings.

- The second contribution of this work concerns the humanoid walking pattern generation. We explore two gait generation techniques. The first approach is planning-based, and consist of planning in real time the reference trajectory for the robot Center of Mass based on a theoretical analysis of the simplified robot model, the Linear Inverted Pendulum Mode (LIPM). The second approach, control based, exploits the linearity of the LIPM to be used in a Model Predictive Control scheme to generate on-line the desired robot motion in order to adapt in real time to the changing conditions of the environment.

1.3 The COMANOID Project

The present work was carried out within the Horizon H2020 European project COMANOID, a EU-funded project focused on Multi-contact Collaborative Humanoids in Aircraft Manufacturing, that has started on January 1, 2015. COMANOID investigates the deployment of robotic solutions in well-defined Airbus airliner assembly operations that are laborious or tedious for human workers and for which access is impossible for wheeled or rail-ported robotic platforms. As a solution to these constraints a humanoid robot is proposed to achieve the described task in real-use cases provided by Airbus Group. At a first glance, a humanoid robotic solution appears extremely risky, since the operations to be conducted are in highly constrained aircraft cavities with non-uniform (cargo) structures. Furthermore, these tight spaces are to be shared with human workers. COMANOID aims at assessing clearly how far the state-of-the-art stands from such novel technologies. In particular the project focuses on implementing a real-world humanoid robotic solution using the best of research and innovation. The main challenge will be to integrate current scientific and technological advances including multi-contact planning and control, advanced visual-haptic servoing, perception and localization and human-robot safety.

1.4 List of publications

The following publications on peer-reviewed conferences have been produced during the work on this thesis:

- *Closed-loop MPC with Dense Visual SLAM - Stability through Reactive Stepping.* D. De Simone, A. Tanguy, A. I. Comport, G. Oriolo, A. Kheddar - 2019 IEEE International Conference on Robotics and Automation, Montreal, Canada

- *Humanoid gait generation for walk-to locomotion using single-stage MPC.* A. Aboudonia, N. Scianca, D. De Simone, L. Lanari, G. Oriolo - 2017 IEEE/RAS International Conference on Humanoid Robots, Birmingham, UK, pp. 178-183, 2017

- *MPC-based humanoid pursuit-evasion in the presence of obstacles.* D. De Simone, N. Scianca, P. Ferrari, L. Lanari, G. Oriolo - 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vancouver, Canada, pp. 5245-5250, 2017

- *Real-time pursuit-evasion with humanoid robots.* M. Cognetti, D. De Simone, F. Patota, N. Scianca, L. Lanari, G. Oriolo - 2017 IEEE International Conference on Robotics and Automation, Singapore, pp. 4090-4095, 2017

- *Intrinsically Stable MPC for Humanoid Gait Generation.* N. Scianca, M. Cognetti, D. De Simone, L. Lanari, G. Oriolo - 2016 IEEE-RAS International Conference on Humanoid Robots, Cancun, Mexico, pp.601-606, 2016

- *Real-Time Planning and Execution of Evasive Motions for a Humanoid Robot.* M. Cognetti, D. De Simone, L. Lanari, G. Oriolo - 2016 IEEE International Conference on Robotics and Automation, Stockholm, Sweden, pp. 4200-4206, 2016

The following journal paper is currently under review:

- *MPC for Humanoid Gait Generation: Stability and Feasibility.* N. Scianca, D. De Simone, L. Lanari, G. Oriolo - Submitted to IEEE Transaction On Robotics

Chapter 2

State of the art

The goal of robotics research is to introduce robots in our daily-life activities, both for service and industrial applications. For this reason, Human-Robot interaction (HRI) is getting more and more attention.

In the past, the robots in the factories were confined in industrial cages, and human workers were not allowed to enter this restricted workspace, without stopping the robot operations. In recent days, however, this situation is changing and robots are requested to collaborate with humans. This means that humans and robots have to share the workspace and safely coexist. To make safe Human-Robot interaction possible, we need to ensure *safety* in the environment, and this means that the robot should be able to perceive the environment and operate without colliding with objects and humans in the surroundings.

The problem of safety has been extensively studied for industrial manipulators, often approached as a collision-avoidance problem, and then successfully extended to mobile wheeled robots. However the problem of safety for humanoid robots did not receive much attention.

In the following we will briefly review the relevant approaches for safety with industrial manipulators and mobile robots. Then we will focus on the problem of safety with humanoid robots.

Due to the unstable nature of the system, the problem of safety must be tackled under two different points of view, the first concerning the stability and robustness of the robot locomotion, while the second focuses on how the humanoid should behave to safely coexist in a workspace shared with humans.

Thanks to the introduction of lightweight, compliant manipulators, robotics research has experienced an increasing interest to the development of new techniques for safe coexistence and interaction with humans leading to many interesting results. These works focused on the main issues for safe HRI, such as collision avoidance, detection and reaction [1] as well as the definition of frameworks for minimizing the risk in HRI [2, 3].

The problem of collision avoidance in cluttered environment has been studied widely, with the introduction of innovative concepts like the potential fields [4] or the danger fields [5, 6, 7].

In the first, the collision avoidance algorithm relies on the use of artificial potential fields. The robot is assumed to be moving in a field of forces. The goal of the robot generates an attractive field while each obstacle in the environment generates a repulsive force field, in order to push the robot away. However this approach, although it has been validated with manipulators and mobile robots, is subject to local minima.

In the second, the authors propose a dual approach, that takes into account the relative motion between the robot and the objects. The concept of danger fields is then extended with the introduction of the safety fields. The latter represent an index of safety rather than an index of danger.

Another approach of interest is the one in [8, 9]. Here the authors present an algorithm to avoid collisions through a correct evaluation of the distance between the robot and the obstacles in real time. The distance evaluation is performed in the so-called depth-space.

The collision avoidance problem, can be also considered as a motion planning problem. An RRT-based motion algorithm is introduced in [10] to plan the motion of a manipulator in the presence of moving obstacles. Another approach for dynamic environments is presented in [11] in a genetic algorithm fashion. Authors in [12] propose a real-time planning/replanning framework proved to be asymptotically optimal in static environments. The problem of safe HRI is also considered in [13, 14, 15], where the authors use the information about human intent to improve safety during human-robot collaboration activities.

In [16], the problem of robot navigation in the presence of humans is considered, and also in [17, 18] the problem of safe navigation is faced.

While the fundamental issues are the same, the design of safety layers for humanoids must take into account the distinctive peculiarities of these robotic systems, namely the fact that their base can only be displaced through stepping gaits and that equilibrium must be maintained at all times during motion. One of the first works that showcased a humanoid avoiding moving obstacles was [19], where real-time vision and replanning were used for autonomous navigation with ASIMO; more recent results are presented in [20] and, using Model Predictive Control (MPC) techniques, in [21, 22].

Model Predictive Control is now widely used to solve the problem of gait generation since in [23] the preview control of [24], was reformulated as an MPC problem. The main requirement is obviously that the robot maintains dynamic balance while walking. A celebrated sufficient condition that guarantees this property entails that the Zero Moment Point (ZMP, the point where the horizontal component of the moment of the ground reaction forces becomes zero) should remain at all times within the support polygon of the robot.

Many gait generation schemes enforce this ZMP condition by computing a suitable trajectory for the robot Center of Mass (CoM). Due to the complexity of humanoid dynamics, simplified models are invariably used to relate the evolution of the CoM to that of the ZMP. A second-order linear system is often adopted, known as the Linear Inverted Pendulum (LIP) or the Cart-Table (CT) depending on whether the ZMP is seen as an input or as an output [25]. Once a CoM trajectory is generated, kinematic control provides joint commands that drive the robot along it.

In MPC-based approaches, the ZMP trajectory is generated on-line by minimizing the control effort with the balance condition as a constraint. Improvements were obtained in [26, 27] by incorporating footstep generation in the MPC problem. These papers triggered a new line of works, e.g., [28, 29].

A related approach to gait generation is based on the notion of capture point, which was first introduced in [30, 31] for investigating the push recovery problem and then extended to the 3D case in [32]. Its use in an MPC framework on the LIP model has been discussed in [33] and [34].

In spite of their diversity, all the above approaches face a well-known structural problem when connecting CoM to ZMP trajectories. This can be understood, e.g., by considering that the LIP is unstable, while the CT — which may be seen as its inverse system — is non-minimum phase. As a consequence, an instability problem will arise in both cases, meaning that the CoM motion associated to any ZMP trajectory will in general include a divergent component. To achieve effective gait generation, it is therefore essential to select a CoM trajectory which does not diverge, i.e., it is *stable*. Indeed, in the CT-based MPC framework there is no guarantee of stability; however, it is argued that jerk minimization produces stable CoM trajectories provided that the prediction horizon is sufficiently long [35].

A recent study [36] of the above LIP instability issue has identified a necessary and sufficient condition for the CoM trajectory to be stable in response to any ZMP profile. This is a constraint on the initial system state, whose expression involves the future history of the input. In view of the duality of the LIP and CT, this result can be recast in the framework of stable inversion for non-minimum phase systems [37].

Chapter 3

Humanoid Locomotion

In this chapter we propose two different approaches to realize humanoid locomotion. The first, is based on the planning of the Zero-Moment Point (ZMP), while the second is based on Model Predictive Control (MPC). Both of the approaches rely on the well-known Linear Inverted Pendulum Mode (LIPM) [38], introduced to simplify the modeling of the humanoid robot dynamics. After analyzing the unstable dynamics of the LIPM, we derive a boundedness condition that must be included in both approaches to guarantee the stability of the system.

3.1 Linear Inverted Pendulum Mode

A humanoid robot is a system with a large number of degrees of freedom (DoF) and very complex dynamics. For this reason it is a common practice to make use of a simplified model to plan and control the locomotion of a humanoid robot.

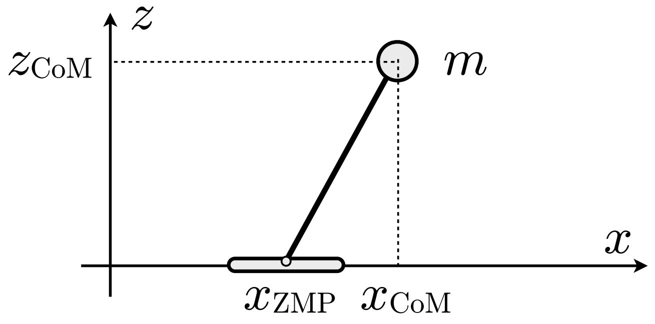


Fig. 3.1: The Linear Inverted Pendulum (LIP) in the sagittal plane.

The LIPM [25] is an approximate model that describes the motion of the humanoid Center of Mass when its height is kept constant and no rotational effects are taken into account. It is a popular choice due to the fact that the differential equations of the x and y components of motion are linear, decoupled and identical. Figure 3.1 shows an inverted pendulum, where x_{CoM} is the sagittal coordinate of the concentrated mass m, set at a constant height z_{CoM}, and x_{ZMP} is the position of the ZMP on the sagittal axis. The ZMP is the point with respect to which the sum of the reaction forces does not produce a moment in the horizontal direction.

The equation of motion governing the LIPM on the sagittal axis is

$$\ddot{x}_{\text{CoM}}(t) = \eta^2(x_{\text{CoM}}(t) - x_{\text{ZMP}}(t)), \tag{3.1}$$

where $\eta = \sqrt{g/z_{\text{CoM}}}$.

If we look at the state-space representation of the system, considering as state of the pendulum the position and velocity of its CoM $(x_c, \dot{x}_c)$ and as input for the system the ZMP position x_z

$$\begin{pmatrix} \dot{x}_\mathrm{c} \\ \ddot{x}_\mathrm{c} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ \eta^2 & 0 \end{pmatrix} \begin{pmatrix} x_\mathrm{c} \\ \dot{x}_\mathrm{c} \end{pmatrix} + \begin{pmatrix} 0 \\ -\eta^2 \end{pmatrix} x_\mathrm{z} \tag{3.2}$$

it is evident that the system has an unstable dynamics associated to the positive eigenvalue η. As a consequence, an instability problem will arise when connecting the ZMP to the CoM, meaning that the CoM motion associated to any ZMP trajectory will in general include a divergent component. To achieve effective gait generation, it is therefore essential to select a CoM trajectory which does not diverge, i.e., it is *stable*. The above LIPM instability issue can be solved with the identification of a necessary and sufficient condition for the CoM trajectory to be stable in response to any ZMP profile [36]. This is a constraint on the initial state of the system, whose expression involves the future history of the input.

Considering the following change of coordinates

$$\begin{pmatrix} x_\mathrm{u} \\ x_\mathrm{s} \end{pmatrix} = \begin{pmatrix} 1 & 1/\eta \\ 1 & -1/\eta \end{pmatrix} \begin{pmatrix} x_\mathrm{c} \\ \dot{x}_\mathrm{c} \end{pmatrix}, \tag{3.3}$$

the system (3.2) can be decomposed into its unstable and stable component of motion $(x_\mathrm{u}, x_\mathrm{s})$

$$\begin{pmatrix} \dot{x}_\mathrm{u} \\ \dot{x}_\mathrm{s} \end{pmatrix} = \begin{pmatrix} \eta & 0 \\ 0 & -\eta \end{pmatrix} \begin{pmatrix} x_\mathrm{u} \\ x_\mathrm{s} \end{pmatrix} + \begin{pmatrix} -\eta \\ \eta \end{pmatrix} x_\mathrm{z}. \tag{3.4}$$

Focusing on the solution for the unstable subsystem for input $x_z(t)$

$$x_u(t) = e^{\eta(t-t_0)}x_u(t_0) - \eta \int_{t_0}^{t} e^{\eta(t-\tau)}x_z(\tau)d\tau \tag{3.5}$$

we note that it is, in general, diverging. However it exists an appropriate initial condition $x_{\mathrm{u}}^*(t_0)$ for the system, that allows to obtain a particular solution that will be bounded with respect to the input. This initial condition takes the form

$$x_{\mathrm{u}}^*(t_0) = \eta \int_{t_0}^{\infty} e^{-\eta(\tau-t_0)}x_{\mathrm{z}}(\tau)d\tau, \tag{3.6}$$

and leads to the particular bounded solution

$$x_{\mathrm{u}}^*(t) = \eta \int_{t}^{\infty} e^{-\eta(\tau)}x_{\mathrm{z}}(\tau+t)d\tau. \tag{3.7}$$

Note that this solution, as already mentioned, is anticausal, since it depends on the future values of the input x_z. However it has been noticed in [24] that the CoM trajectory must be noncausal with respect to the desired ZMP trajectory.

The condition expressed in (3.6) can be used to write a constraint on the initial conditions of the CoM

$$x_{\mathrm{c}}(t_0) + \dot{x}_{\mathrm{c}}(t_0)/\eta = \eta \int_{t_0}^{\infty} e^{-\eta(\tau-t_0)}x_{\mathrm{z}}(\tau)d\tau, \tag{3.8}$$

and we refer to this as the boundedness constraint [36]. Since this constraint is a function of the initial state and the input, it will be crucial in the process of gait generation, because it can be used to constraint the initial condition of the system if the reference ZMP input is already known, or constraint the design of the input given the initial conditions of the system.

In the remainder of this chapter we will make use of (3.8) in the design of the ZMP control input for the robot. First we will relate the ZMP to the CoM of the humanoid through a closed-form expression, allowing us to compute in real time a desired CoM trajectory to control the humanoid motion. Then we will enforce the boundedness constraint in a Model Predictive Control scheme.

3.2 ZMP-based Gait Generation

The elements that characterize the planning of a humanoid gait are the position of the footsteps and the trajectory of the ZMP and the CoM. An intuitive way of planning the gait consists of a preliminary phase of footstep planning [39, 40, 41], followed by the the definition of a suitable ZMP trajectory. The condition for maintaining dynamic balance during locomotion is that the ZMP must lie at any time instant inside the current supporting polygon. The support polygon is the robot footprint during the single support phase of the locomotion, or the convex hull of the two feet during the double support phase. So, given a footstep sequence, a ZMP trajectory can be easily computed (e.g. by an appropriate interpolation of the planned footstep positions).

Once the ZMP trajectory is planned, using the LIPM dynamics (3.1) and imposing the boundedness constraint (3.8), it is possible to obtain a closed-form expression that relates the reference ZMP to a bounded CoM trajectory

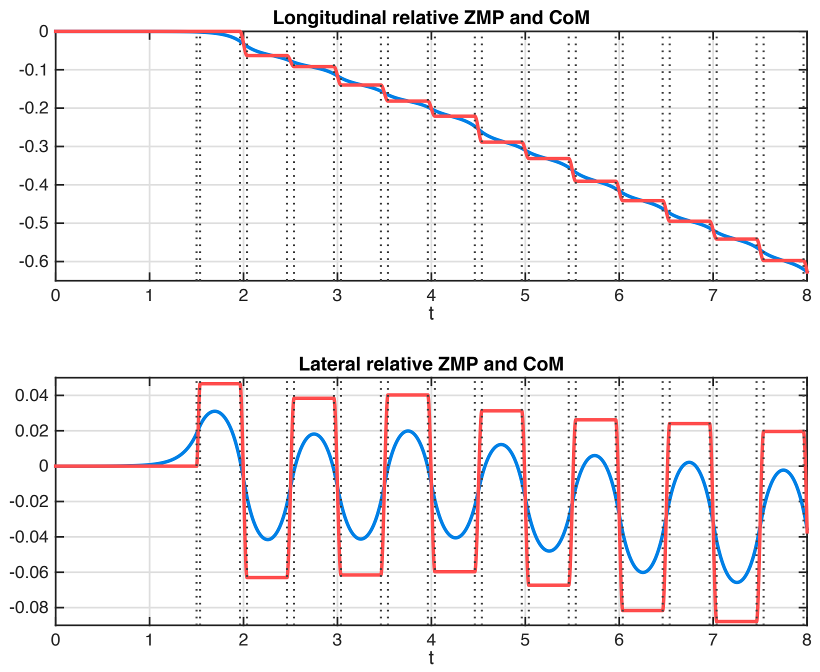


Fig. 3.2: Evolution in time of the reference ZMP trajectory (red) and the associated bounded CoM trajectory (blue).

$x_c^*(t)$:

$$x_c^*(t) = e^{-\eta t} x_c(0) + \frac{x_s(t) - e^{-\eta t} x_u(0) + x_u(t)}{2}. \tag{3.9}$$

The use of a closed-form relation to compute the desired trajectory for the robot Center of Mass allows to plan and execute the desired motion in real time. Figure 3.2 shows an example of CoM trajectory computed from a reference ZMP trajectory using eq. (3.9).

3.3 MPC-based Gait Generation

Model Predictive Control (MPC) is a widely used control technique based on the optimization of a certain cost function, over a finite prediction horizon while satisfying a set of constraints. The key idea behind MPC is to optimize

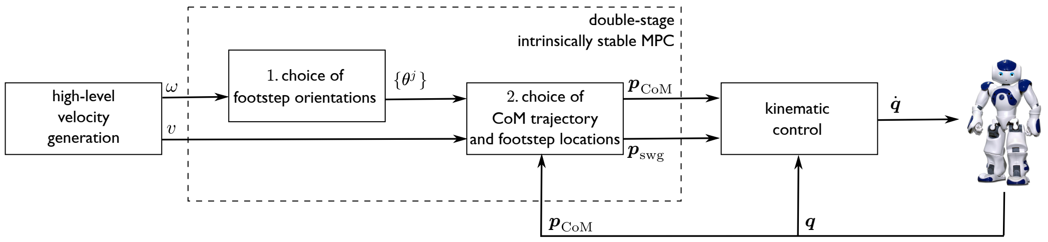

double-stage
intrinsically stable MPC
high-level velocity generation
ω
v
1. choice of footstep orientations
$\{\theta^j\}$
2. choice of CoM trajectory and footstep locations
$\boldsymbol{p}_{\mathrm{CoM}}$
$\boldsymbol{p}_{\mathrm{swg}}$
kinematic control
$\dot{\boldsymbol{q}}$
$\boldsymbol{p}_{\mathrm{CoM}}$
$\boldsymbol{q}$

Fig. 3.3: Intrinsically stable MPC gait generation framework.

the current time instant, while keeping future time instants into account. This is achieved by optimizing over a finite-time horizon, but only implementing the current time instant and then optimizing again. Also, MPC has the ability to anticipate future events and take control actions accordingly.

Due to its simple but robust formulation and its versatility, Model Predictive Control became a powerful and commonly used tool for generating walking gaits. The possibility of enforcing constraints allows to include straightforwardly the dynamic balance condition that the ZMP must remain at any time within the support polygon. Second, it provides considerable robustness to perturbations thanks to its capability to predict future events through the system model. Moreover, if formulated as a quadratic programming (QP) problem, with constraints linear in the decision variable it can be used for real-time computations.

In our formulation, the Intrinsically Stable MPC (IS-MPC) [42], the motion model used is the LIPM (3.1) with a dynamic extension

$$\begin{pmatrix} \dot{x}_{\text{c}} \\ \ddot{x}_{\text{c}} \\ \dot{x}_{\text{z}} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ \eta^2 & 0 & -\eta^2 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_{\text{c}} \\ \dot{x}_{\text{c}} \\ x_{\text{z}} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \dot{x}_{\text{z}} \tag{3.10}$$

to have as input the ZMP velocity $\dot{x}_{\text{z}}$, assumed to be piece-wise constant over time intervals of duration δ.

The decision variables of the optimization problem are the ZMP velocities $(\dot{x}_{\text{z}}^i, \dot{y}_{\text{z}}^i)$, $i = 1, \dots, N$ and the footstep positions and orientations $(x_{\text{f}}^j, y_{\text{f}}^j, \theta_{\text{f}}^j)$, $j = 1, \dots, M$ over the prediction horizon $T_h = N\delta$. The optimization will take care of choosing appropriate ZMP velocities and footstep positions and orientations $(x_{\text{f}}^j,\ y_{\text{f}}^j,\ \theta_{\text{f}}^j)$ to track high-level reference velocities. However to maintain linearity in the MPC formulation, the footstep orientations must be chosen before the computation of their positions [27]. Therefore, the optimization is divided in two stages, the first to decide the appropriate footstep orientations θ_{f}^j, while the second is to determine the footstep positions $(x_{\text{f}}^j, y_{\text{f}}^j)$ and the ZMP velocities $(\dot{x}_{\text{z}}^i, \dot{y}_{\text{z}}^i)$.

Figure 3.3 shows a block scheme of the gait generation process. The MPC block takes as input the vector $v = (v_{\text{x}}, v_{\text{y}})$ of reference sagittal and coronal velocities and the reference angular velocity ω to be tracked. These signals are exogenous, provided by a high-level velocity generation block, according to the required task for the robot. The angular velocity is used to compute the optimal footstep orientations θ_{f}^j, that are sent with the reference linear velocities as input to the second stage of the IS-MPC block. This block produces as output a reference trajectory for the robot CoM and feet, that are then transformed by the kinematic controller into reference velocities for the robot joints.

In the following we will describe the designed cost functions and the associated constraints.

3.3.1 Cost Function

As said, the MPC gait generator takes as input high-level reference velocities and generates a CoM trajectory and a set of footsteps such that to track those references. In the double stage formulation, to maintain the constraint linear in the decision variables, the footstep orientations must be chosen via a preliminary optimization stage w.r.t the decision of the positions and the ZMP velocities.

To do so the reference velocity ω is used inside a cost function of the form

$$\sum_{j=1}^{M} \left(\frac{\theta_{\mathrm{f}}^{j} - \theta_{\mathrm{f}}^{j-1}}{T_s} - \omega \right)^2 , \tag{3.11}$$

where T_s is the constant duration of a step. This first optimization problem is subject to the linear constraint $|\theta_{\mathrm{f}}^{j} - \theta_{\mathrm{f}}^{j-1}| \leq \theta_{max}$, that limits to θ_{max} the maximum difference in orientation between two consecutive footsteps.

Once footstep orientations have been chosen, the gait must be completed by the CoM trajectory and the footstep locations. The cost function

$$\begin{aligned}\sum_{i=1}^{N} \big((\dot{x}_{\mathrm{z}}^{k+i})^2 + (\dot{y}_{\mathrm{z}}^{k+i})^2 + \\ k_x(\dot{x}_{\mathrm{c}}^{k+i} - v_x\cos(i\omega\delta) + v_y\sin(i\omega\delta))^2 + \\ k_y(\dot{y}_{\mathrm{c}}^{k+i} - v_x\sin(i\omega\delta) - v_y\cos(i\omega\delta))^2\big)\,.\end{aligned} \tag{3.12}$$

is designed with the purpose of minimizing the control effort and to penalize the deviation from the reference velocities (v_x,v_y). Note that footstep locations do not appear in the cost function, but influence the QP problem through the linear constraints to witch the problem is subject, i.e. the ZMP constraint for maintaining balance, the CoM boundedness constraint and the kinematic feasibility on the footstep locations. These constraints are discussed in the following section.

3.3.2 Constraints

When dealing with humanoid locomotion, there are two constraints that must be enforced in order to generate a feasible and executable gait. The first is the balance constraint, to ensure that the computed ZMP will always lie inside the current support polygon to maintain dynamic balance. The second constraint is on the footstep positions, since the feasible area where the robot can actually step is limited by the workspace that the robot can kinematically reach. Moreover the chosen footsteps must not lead the robot to self-collisions. In addition to these two constraint, the boundedness constraint (3.8) must be enforced.

The ZMP constraint is defined as a rectangle with sides $d_x^{\mathrm{z}}, d_y^{\mathrm{z}}$, and the form of the linear constraint is

$$R_j^T \begin{pmatrix} \delta \sum_{l=k}^{k+i-1} \dot{x}_{\mathrm{z}}^l - x_{\mathrm{f}}^j \\ \\ \delta \sum_{l=k}^{k+i-1} \dot{y}_{\mathrm{z}}^l - y_{\mathrm{f}}^j \end{pmatrix} \leq \frac{1}{2} \begin{pmatrix} d_x^{\mathrm{z}} \\ \\ d_y^{\mathrm{z}} \end{pmatrix} - R_j^T \begin{pmatrix} x_{\mathrm{z}}^k \\ \\ y_{\mathrm{z}}^k \end{pmatrix}, \tag{3.13}$$

where R_j is the rotation matrix associated to the angle θ_{f}^j.

The kinematic feasibility constraint for the footsteps is

$$R_{j-1}^T \begin{pmatrix} x_{\mathrm{f}}^j - x_{\mathrm{f}}^{j-1} \\ \\ y_{\mathrm{f}}^j - y_{\mathrm{f}}^{j-1} \end{pmatrix} \leq \pm \begin{pmatrix} 0 \\ \\ \ell \end{pmatrix} + \frac{1}{2} \begin{pmatrix} d_x^{\mathrm{f}} \\ \\ d_y^{\mathrm{f}} \end{pmatrix}, \tag{3.14}$$

where d_x^{f} and d_y^{f} are the sides of a rectangle defining the feasibility zone, and l is a reference distance between two consecutive footsteps.

Finally the boundedness constraint (3.8) takes the form

$$\frac{1}{\eta} \frac{1 - e^{\delta\eta}}{1 - e^{N\delta\eta}} \sum_{i=1}^{N} e^{i\delta\eta} \dot{x}_{\mathrm{z}}^{k+i} = x_{\mathrm{c}}^k + \frac{\dot{x}_{\mathrm{c}}^k}{\eta} - x_{\mathrm{z}}^k. \tag{3.15}$$

and guarantees the boundedness of the CoM trajectory w.r.t. the reference ZMP profile. Also, enforcing this constraint allows to reduce the prediction horizon T_h without harming the gait stability. This also leads to an improvement of the resolution time of the optimization problem.

Figures (3.4-3.5) show a comparison between the IS-MPC and the standard MPC, highlighting the instability issue when the prediction horizon is reduced. The IS-MPC is always able to generate a non-diverging trajectory of the CoM even when the prediction horizon is reduced. Conversely, the standard formulation of the MPC generates a diverging CoM trajectory when the prediction horizon is reduced.

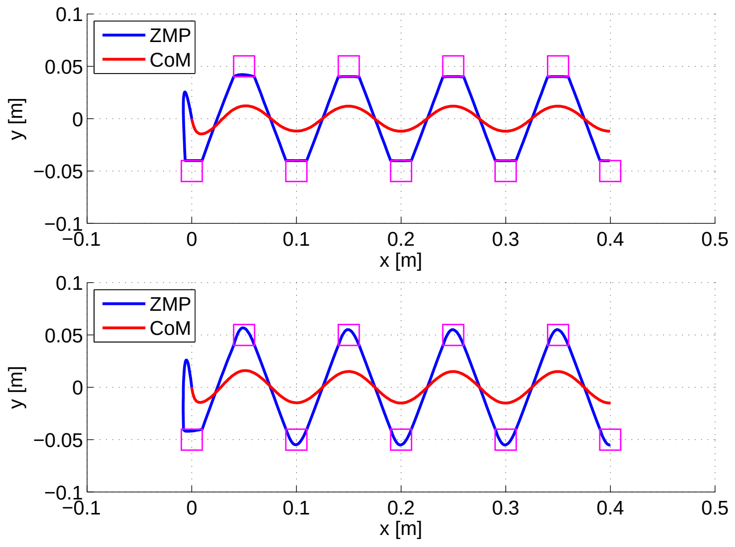


Fig. 3.4: $T_{\mathrm{h}} = 0.9$ s: Intrinsically stable (top) vs. standard MPC (bottom).

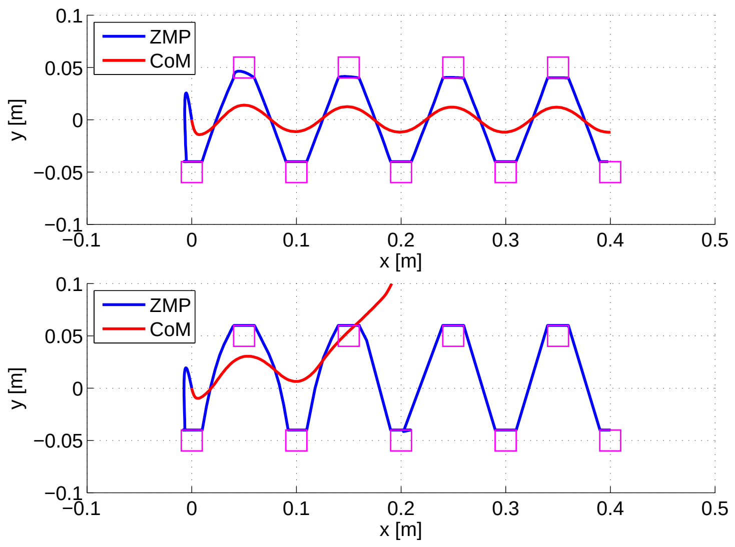


Fig. 3.5: $T_{\mathrm{h}} = 0.6$ s: Intrinsically stable (top) vs. standard MPC (bottom).

A single-stage formulation of the Intrinsically Stable MPC have been proposed in [43]. In that formulation, the first stage in charge of choosing the footstep orientation has been eliminated by choosing the footstep orientations at the same time of the other decision variables and properly redefining all constraints to preserve linearity, with the exception of the stability constraint (3.15) which remains linear and is therefore unchanged.

The ZMP constraint (3.13) becomes nonlinear if the orientation θ_f^j of the footstep is still a decision variable. To avoid this problem, we redefined the constraint so that it becomes independent on the foot orientation. In particular, consider the construction in Figure 3.6, left. 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 as

$$\begin{pmatrix} \delta \sum_{l=k}^{k+i-1} \dot{x}_{\mathrm{z}}^l - x_{\mathrm{f}}^j \\ \delta \sum_{l=k}^{k+i-1} \dot{y}_{\mathrm{z}}^l - y_{\mathrm{f}}^j \end{pmatrix} \leq \frac{1}{2} \begin{pmatrix} \tilde{x}_z^{\mathrm{dim}} \\ \tilde{y}_z^{\mathrm{dim}} \end{pmatrix} - \begin{pmatrix} x_{\mathrm{z}}^k \\ y_{\mathrm{z}}^k \end{pmatrix},$$

where $\tilde{x}_z^{\mathrm{dim}} = x_{\mathrm{z}}^{\mathrm{dim}}/\sqrt{2}$ and $\tilde{y}_z^{\mathrm{dim}} = y_{\mathrm{z}}^{\mathrm{dim}}/\sqrt{2}$.

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 foosteps

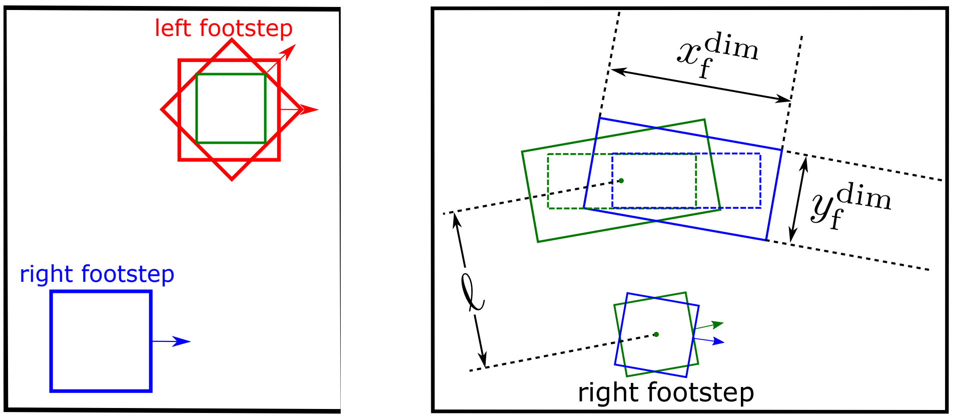


Fig. 3.6: Redefining the ZMP constraint (left) and the kinematic feasibility constraint for footstep locations (right).

orientation in the main MPC formulation (see Figure 3.6, right). As the ZMP constraint, the kinematic feasibility constraint (3.14) becomes nonlinear if the orientations of the footsteps have not been chosen yet. To circumvent this problem, we redefine also this constraint appropriately. Figure 3.6, right, 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 depend 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 footstep 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

$$\begin{pmatrix} x_{\rm f}^{j+1} - x_{\rm f}^{j} - \ell\,\theta^{j} \\ y_{\rm f}^{j+1} - y_{\rm f}^{j} \end{pmatrix} \leq \begin{pmatrix} \tilde{x}_{\rm f}^{\rm dim}/2 \\ \ell + \tilde{y}_{\rm f}^{\rm dim}/2 \end{pmatrix},$$

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

Chapter 4

Safety Framework

This chapter is dedicated to the description of the safety framework for human-humanoid coexistence. This framework has its roots in a collection of guidelines that should be followed in the process of motion planning and included in the robot controllers. Starting from the safety guidelines, we have developed a set of safety behaviors, i.e. a set of actions that the robot must execute to enhance the overall safety level, according to its state and the state of the surrounding environment.

In the following of this chapter we will first describe the safety guidelines and then propose the safety behaviors that will actually realize those guidelines. Also, to manage this behavior-based framework, we propose a state machine to handle the transitions between behaviors. Finally we show a possible control architecture.

4.1 Safety Guidelines

An important feature for a robot is that during its operation it is aware of the surroundings. In general this knowledge is obtained through vision sensors (RGB/RGB-D cameras) or lasers, that are commonly used to build a map of the environment and to detect objects in it through the process of feature extraction. It is evident that to achieve these results it's necessary to appropriately control the robot gaze. The first safety guideline, *watch what you're doing* is inspired by this need for the robot to control it's gaze to always watch the main area of operation. This is of course true during static manipulation tasks, that are generally visual-servoed, but it is crucial also during locomotion. In fact while the robot walks, it should focus on the area where it has planned to step in order to be ready to react to unexpected situations.

Even when the robot is not performing any particular task, it should always *be on the lookout*, meaning that it must monitor the environment to keep its knowledge of the surroundings up-to-date, and eventually focus on unmapped objects (e.g., moving objects) and be ready to react.

Evade if you can is the guideline to follow in case a moving object approaches the robot while it's not performing any task. If it can be done safely, the robot can try to perform an evasive action to avoid any possible collision and increase the distance between itself and the incoming obstacle.

During any robot operation, whether it is static manipulation or locomotion, the robot should be ready and able to stop the task as soon as possible in a situation of clear and actual danger. This is the meaning of the guideline *stop if you must.*

In the case the robot is sharing its workspace with humans, an intuitive way of enhancing the safety level is to scale down velocities and forces in order to reduce potential damage in the case of a collision, and also to ease the halting process in case of emergency. This means that the robot must *respect humans* that share their environment with it.

Finally, in challenging conditions such as stairs, the robot must *look for support*, meaning that it should try to establish additional contacts with the environment so that it has at least two support points at all times.

The presented guidelines, can be considered as a starting point to realize safety behaviors that are activated to improve the level of safety in dangerous situations. On the other hand, they can be taken into account also at the planning/control stage. For example, *watch what you're doing*, as said, calls for visual-servoed manipulation or locomotion strategies, while *look for support* has consequences at the planning stage, since the generation of stair climbing motions must include handrail grasping and releasing to always have at least two supporting points.

4.2 Safety Behaviors

The safety behaviors are actions, based on the previously defined safety guidelines, that the robot should perform during the execution of its operations. We classify the behaviors in *override* behaviors and *proactive* behaviors.

Override behaviors will actually stop (or put on hold) the current task and force the robot to take action against a dangerous situation that has been detected.

Proactive behaviors, on the other hand, do not stop the task, but try to increase the overall safety level by calling for an adaptation or enhancement of the current robot activity.

4.2.1 Override behaviors

As the name suggests, the override behaviors will take control over the current robot operation, providing the robot with a new task to properly react to a particularly dangerous situation. We define two override behaviors, namely the safe fall behavior and the emergency stop behavior.

Despite it is a situation that we want to avoid, the occurrence of a fall cannot be excluded. The robot may lose balance while performing a step or due to a hardware/software fault. An external perturbation (e.g., a collision with an object) may also lead to a loss of balance. This is the reason why a safe fall procedure to deal with the loss of balance must be included in the robot controller. When the robot detects an unrecoverable loss of balance, it must immediately stop any task execution and fall so as to minimize damages to itself and the environment.

Many situations that interfere with safe operations may occur (e.g. the battery level is too low or a moving object is getting to close too the robot area of operation). In that case, following the *stop if you must* guideline, the robot must be ready to immediately interrupt the execution of its tasks. Clearly, with a humanoid robot the emergency stop must be handled properly since, especially during locomotion, we want the robot to keep its balance.

4.2.2 Proactive behaviors

Contrary to override behaviors, proactive behaviors are actions intended to modify the current robot task in order to increase its overall safety, without interrupting the main task.

Following the *evade if you can* guideline, the first proactive behavior that we propose is the evasion. If the robot is not performing any particular task (e.g. is waiting for another robot or a human to complete a task) and it detects an approaching moving object, it performs an evasive maneuver to avoid the collision.

The second proactive behavior is the visual tracking. Whenever possible, without interfering with the current task, the robot keeps its gaze directed at the closest unexpected object (e.g. something that is not in the robot local map) in its field of view, as suggested by the *be on the lookout* guideline.

During locomotion, the robot is in general controlled via a high-level task, such as tracking of a reference velocity, or reaching a specific location in the workspace. The footstep adaptation behavior allows the robot to modify its footsteps to avoid collision with unexpected objects in its path or to react to external disturbances that require extra stepping to recover from a possible loss of balance.

In general the robot must be able to coexist with other agents in the environment. For this reason, according to the guideline *respect humans*, the velocity/force scaling behavior is introduced. If a close unexpected object is perceived, the robot reduces its velocities/forces to reduce the risk of injures in case a collision occurs. Clearly, with a humanoid robot this must be done properly, considering that the humanoid robot must always keep its balance and scaling velocities/forces directly at the joint level might lead the robot to a fall.

Finally, the add contact behavior, following the *look for support guideline* is a procedure that allows the robot to try to establish new contacts for additional support, if this can be done without harming the current task.

In the next section we will define the state machine that controls the behaviors described so far, and we will also give a deeper definition of the behaviors.

4.3 State Machine

This section is dedicated to the definition of the finite state machine that controls the robot behaviors. First we will define a set of states for the robot, each of which generalizes the current robot operation. Then we will present some assumptions that we made throughout the realization of this framework. Finally all the behaviors introduced in the previous section will be better described using the terminology of finite state machines.

The robot states charachterize what the robot is doing at a certain time instant. We identify five robot states:

- *idle*: the robot is standing in double support at a fixed position and is not performing any particular task;
- *manipulation*: the robot is standing in double support and it is executing a task that does not require stepping;

- *locomotion*: the robot is moving in the environment by taking steps (includes walking, multi-contact locomotion and stair climbing/descending);
- *falling*: the robot has lost balance and is falling;
- *stopping*: the robot is performing an emergency stop.

We assume that at any time instant the robot is in one of the states previously defined. Moreover we assume that:

- **robot is localized**: the robot is aware of its location w.r.t. a 3D map (SLAM);
- **robot aware of risk of fall**: the robot can estimate the risk of fall r_{fall} based on IMU measurement and ZMP support area;
- **robot aware of battery level**: the robot knows the level of charge of its battery $l_{battery}$;
- **robot aware of contact surfaces**: the robot knows the surfaces in the 3D map where it is safe to establish new contacts. The flag f_{cs} indicates the presence of a reachable surface in the robot workspace;
- **robot aware of unexpected objects**: the robot knows the minimum distance d_{uo} to the closest unexpected object;
- **robot aware of unexpected contacts**: the robot is able to detect if a contact occurred. It may also provide other information such as contact location. When a contact is detected, the flag f_{uc} is raised.

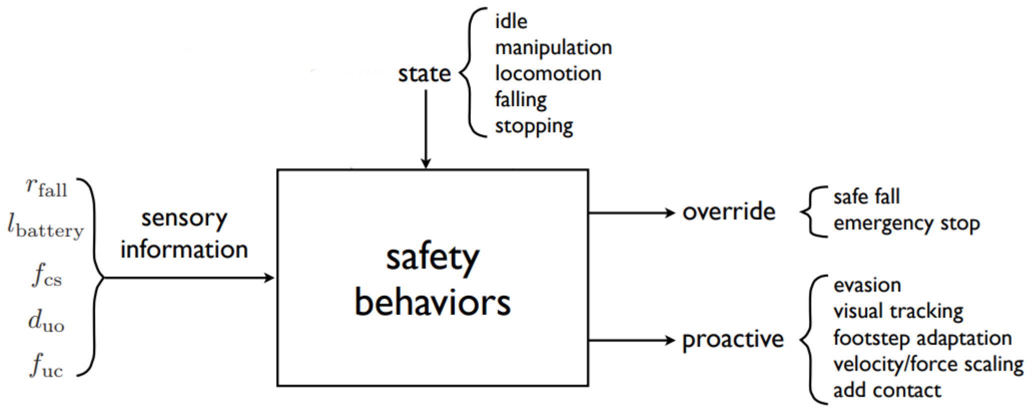


Fig. 4.1: Safety Behaviors.

We will not discuss in any detail the perception processes that provide such information.

4.3.1 Behaviors

Each behavior described in section 4.2 is characterized by:

- a *triggering condition*, that specifies the robot state(s) from which the behavior can be activated, together with the events that actually cause the activation, which are invariably expressed in terms of information coming from the sensory system (fig. 4.1);
- the *actions* that realize the behavior and the robot state during such actions;
- the *release condition* for terminating the behavior, and the state to which the robot is returned upon release.

All triggering conditions are continuously checked by the robot. If its triggering condition is met, a behavior can therefore interrupt another behavior prior to its natural release. For example an emergency stop may occur at any time during normal robot operation.

Safe Fall

In the case the robot loses balance, this behavior is activated, starting a procedure that will try to minimize the damages to the robot and the surrounding environment.

The safe fall behavior is triggered if:

- **State**: any.
- **Event**: $r_{fall} > r_{fall}^{high}$

where r_{fall}^{high} is a threshold associated to an unrecoverable loss of balance. The safe fall behavior is specified as follows:

- **Action**: the robot acts so as to minimize the potential damage to its own structures and/or the environment. To this end, several aspects must be considered [44] including (i) how to fall, i.e., which internal posture to assume before impact to preserve robot integrity (ii) where to fall, i.e., how to choose the landing surfaces so as to avoid fragile components.
- **State during action**: falling. Since a fall is regarded as a catastrophic event, the safe fall behavior can only be released by the intervention of a human operator, who will also reset the robot to an appropriate state.

- **Release condition**: by human intervention.
- **Release state**: by human intervention.

Emergency Stop

The emergency stop behavior requires a proper definition since, while the robot tries to interrupt is current task as soon as possible, it has to avoid any loss of balance. The triggering condition for emergency stop is:

- **State**: any except falling.
- **Event**: three events may independently trigger an emergency stop:
  - E1: $d_{uo} < d_{uo}^{stop}$, where d_{uo}^{stop} is the radius of the robot proximal area;
  - E2: f_{uc} is active (unexpected contact);
  - E3: $l_{battery} < l_{battery}^{low}$, where $l_{battery}^{low}$ is the minimum acceptable battery level for safe operation.

The emergency stop behavior is defined as follows.

- **Action**: the specific action depends on the triggering conditions:
  - A1: If the robot state was *idle* and the triggering event was E1 or E3, the robot will augment its support polygon and/or assume a low-impact configuration (e.g., by folding its arms);
  - A2: If the robot state was *idle* and the triggering event was E2, the robot will immediately decrease joint stiffness on the kinematic

chain where the contact has occurred, provided that the latter is on the upper body. Otherwise, the robot will maintain its current pose;

  - A3: If the robot state was *manipulation* the robot should safely abort the task and stop its motion as soon as possible, regardless of the triggering event;
  - A4 If the robot state was *locomotion*, the robot should stop walking as soon as possible, regardless of the triggering event.

- **State during action**: stopping.
- **Release condition**: if the triggering event was E1, the behavior can be released when $d_{uo} > d_{uo}^{stop}$. If the triggering event was E2 or E3, a human operator should intervene to release the robot.
- **Release state**: If the triggering event was E1, the robot can be returned to the previous state (i.e. the one before activating the emergency stop behavior). Otherwise, by human intervention.

Evasion

If an unexpected object approaches the robot in the *idle* state, the latter should execute an evasive maneuver, provided this can be done safely (*evade if you can* guideline). The evasion behavior is triggered if:

- **State**: idle.
- **Event**: $d_{uo}^{stop} < d_{uo} < d_{uo}^{evasion}$ AND $\dot{d}_{uo} < 0$

Here, $d_{uo}^{evasion}$ is a distance threshold under which evasion is assumed to be advisable; clearly, $d_{uo}^{evasion} > d_{uo}^{stop}$. The condition on the time derivative of d_{uo} can be implemented by looking at variations of this quantity over a small time interval. The evasion behavior is defined as follows.

- **Action**: an evasion maneuver is planned in real time based on the spatial relationship between the robot and the approaching object [45]. Feasibility of the maneuver with respect to the current 3D map is continuously checked. Whenever the maneuver becomes unfeasible, the associated flag $f_{evasion}$ is set to FALSE and the emergency stop behavior is invoked.
- **State during action**: locomotion.
- **Release condition**: $d_{uo} > d_{uo}^{evasion}$ OR $f_{evasion}$ = FALSE.
- **Release state**: idle (if $d_{uo} > d_{uo}^{evasion}$) or stopping (if $f_{evasion}$ = FALSE)

Visual Tracking

If the robot is in idle and an unexpected object appears in its field of view, the robot will direct its gaze at it (*be on the lookout* guideline). The visual tracking behavior is triggered if:

- **State**: idle
- **Event**: $d_{uo}^{evasion} < d_{uo} < d_{track}$

Note that this behavior cannot be triggered if the robot is in the *manipulation* or the *locomotion* states. In these cases, in fact averting the gaze from the current task can be dangerous (*watch what you're doing* guideline). The *visual tracking* behavior is defined as follows

- **Action**: the robot should move its head so as to track the closest unexpected obstacle in its field of view.
- **State during action**: idle.
- **Release condition**: $d_{uo} > d_{track}$
- **Release state**: idle

Footstep adaptation

During locomotion, it is possible that unexpected objects (either moving, such as humans, or fixed, like cables on the ground) may interfere with the planned footsteps. In this case, the robot should replan its footsteps using the new information. The footstep adaptation behavior is triggered if:

- **State**: locomotion
- **Event**: $d_{uo} < d_{uo}^{footstep}$, where $d_{uo}^{footstep}$ is a distance threshold under which footstep adaptation may be advisable

The footstep adaptation behavior is defined as follows

- **Action**: the robot re-invokes walking pattern generation after adding to the current map the unexpected objects that are closer than $d_{uo}^{footstep}$.
- **State during action**: locomotion.

- **Release condition**: $d_{uo} > d_{uo}^{footstep}$ OR (footsteps successfully adapted).
- **Release state**: locomotion

Velocity/force scaling

In the vicinity of an unexpected object, which may be a human, tighter bounds are enforced on robot velocities/forces (*respect humans* guideline). The velocity/force scaling behavior is triggered if:

- **State**: *manipulation* or *locomotion*
- **Event**: $d_{uo} < d_{uo}^{scale}$, where d_{uo}^{scale} is a distance threshold under which scaling is advisable. Clearly $d_{uo}^{scale} > d_{uo}^{stop}$.

The velocity/force scaling behavior is defined as follows.

- **Action**: robot velocities and/or forces are scaled down so as to fit the tighter bounds.
- **State during action**: remains *manipulation* or *locomotion.*
- **Release condition**: $d_{uo} > d_{uo}^{scale}$
- **Release state**: remains *manipulation* or *locomotion.*

If the robot state is locomotion, velocity/force scaling also affects the CoM trajectory; this requires proper handling at the level of walking pattern generation. As for the scaling threshold, $d_{uo}^{scale} = d_{uo}^{footstep}$ appears to be a sensible choice.

Add contact

In the presence of a moderate risk of fall, the robot tries to establish new contacts for additional support (*look for support* guideline). The add contact behavior is triggered if:

- **State**: *idle* or *manipulation*
- **Event**: $r_{fall}^{low} < r_{fall} < r_{fall}^{high}$ AND (f_{cs} = TRUE)

Here, r_{fall}^{low} is the threshold above which the risk of fall is considered to be moderate. Note the role of flag f_{cs} which indicates the presence of a reachable contact surface in the robot workspace. *Locomotion* is not a triggering state for this behavior because it is intrinsically risky to try to establish a new contact while the robot is walking. Moreover, if the robot is climbing/descending stairs, additional contact with the handrail has already been enforced at the planning stage. The add contact behavior is defined as follows

- **Action**: the robot selects an additional support point and establishes contact
- **State during action**: *manipulation*
- **Release condition**: the additional contact has been established
- **Release state**: returns to the initial state, i.e. either *idle* or *manipulation.*

Figure 4.2 gives a compact view of the five robot states and the transitions among them resulting from safety behaviors. Red arrows indicate transitions

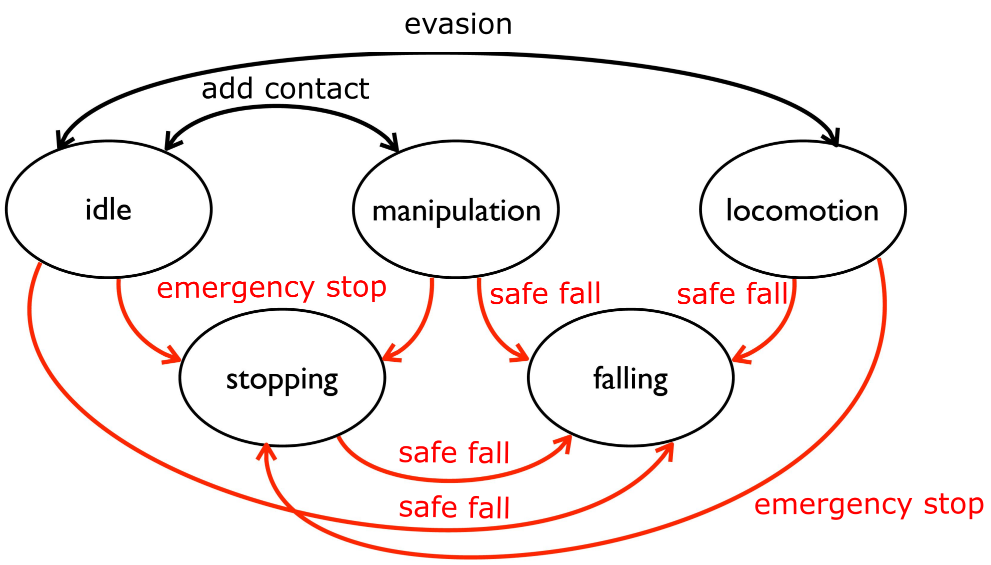


Fig. 4.2: State transitions resulting from safety behaviors: red arrows indicates override behaviors, black arrows indicate proactive behaviors.

resulting from override behaviors: in particular, transitions entering the *stopping* state are the result of an emergency stop behavior, whereas transition entering the *falling* state are the result of a safe fall behavior. Black arrows identify transitions due to proactive behaviors; for example, the evasion behavior brings the state from *idle* to *locomotion* and then back to *idle*.

4.4 Safety areas and thresholds

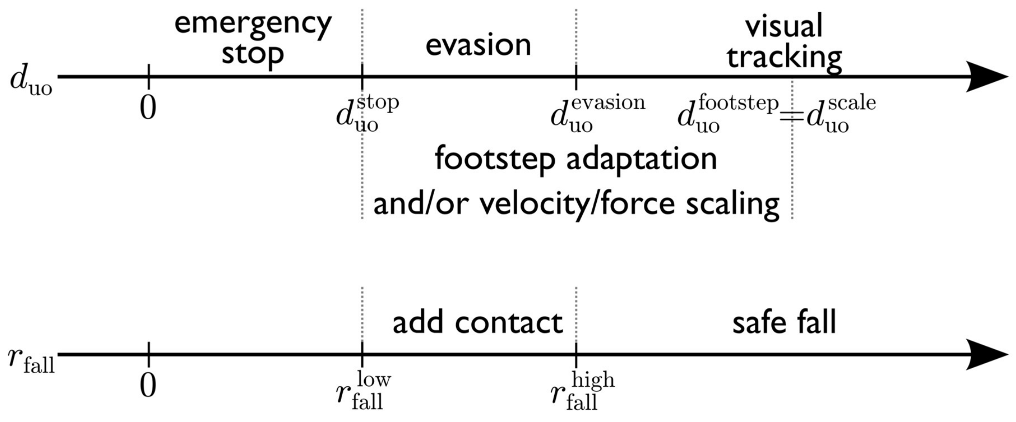

emergency stop
evasion
visual tracking
d_{uo}
0
$d_{\text{uo}}^{\text{stop}}$
$d_{\text{uo}}^{\text{evasion}}$
$d_{\text{uo}}^{\text{footstep}} = d_{\text{uo}}^{\text{scale}}$
footstep adaptation and/or velocity/force scaling
add contact
safe fall
r_{fall}
0
$r_{\text{fall}}^{\text{low}}$
$r_{\text{fall}}^{\text{high}}$

Fig. 4.3: Thresholds associated to the distance of the robot from an unexpected obstacle d_{uo} (top) and on the estimated risk of fall r_{fall} (bottom).

Most safety behaviors are triggered by some value being higher or lower than some given threshold. This can be the distance of the robot from an unexpected obstacle d_{uo} or the estimated risk of fall r_{fall}.

Figure 4.3, top, summarizes the different thresholds that have been defined on d_{uo}: $d_{\text{uo}}^{\text{track}}$, $d_{\text{uo}}^{\text{footstep}}$, $d_{\text{uo}}^{\text{evasion}}$, $d_{\text{uo}}^{\text{scale}}$ and $d_{\text{uo}}^{\text{stop}}$. These thresholds implicitly define a set of concentric annular areas, shown in Fig. 4.4, around the robot:

- $\mathcal{S}^{\text{track}}$ - visual tracking area: this is the area defined by $d_{\text{uo}}^{\text{evasion}} < d_{\text{uo}} < d_{\text{uo}}^{\text{track}}$. If the unexpected moving obstacle enters this area the robot should be tracking if *Idle*.

- $\mathcal{S}^{\text{footstep}}$ - footstep adaptation area: this is the area defined by $d_{\text{uo}}^{\text{stop}} < d_{\text{uo}} < d_{\text{uo}}^{\text{footstep}}$. If an unexpected fixed obstacle is in this area the robot should adapt its footsteps to avoid collision with it.

- $\mathcal{S}^{\text{evasion}}$ - evasion area: this is the area defined by $d_{\text{uo}}^{\text{stop}} < d_{\text{uo}} < d_{\text{uo}}^{\text{evasion}}$. If the unexpected moving obstacle enters this area and the robot is *Idle*, it should plan an evasion maneuver and execute it.

- $\mathcal{S}^{\text{scale}}$ - velocity/force scaling area: this is the area defined by $d_{\text{uo}}^{\text{stop}} < d_{\text{uo}} < d_{\text{uo}}^{\text{scale}}$. If the robot enters *Manipulation* and an unexpected moving obstacle enters this area, the robot should reduce the velocities and forces associated to the manipulation task to avoid damaging contacts with the obstacle.

- $\mathcal{S}^{\text{stop}}$ - emergency stop area: this is the area defined by $d_{\text{uo}} < d_{\text{uo}}^{\text{stop}}$. This is the innermost safety area. If any unexpected obstacle enters this area the robot must safely stop whatever action it is performing and request human intervention.

Figure 4.3, bottom, shows the thresholds used for the risk of fall r_{fall}. As soon as r_{fall} becomes significant ($r_{\text{fall}}^{\text{low}} < r_{\text{fall}} < r_{\text{fall}}^{\text{high}}$) the robot activates the *add contact* behavior. If a fall is deemed inevitable ($r_{\text{fall}} > r_{\text{fall}}^{\text{high}}$), the robot will perform a *safe fall*.

4.5 Control Scheme

The implementation of the safety behavior architecture will invariably depend on the specific platform and control architecture. Here we will refer to a generic architecture for illustration.

Figure 4.5 shows a general overview of the control scheme. As already noted, some details are intentionally left unspecified as they can be imple-

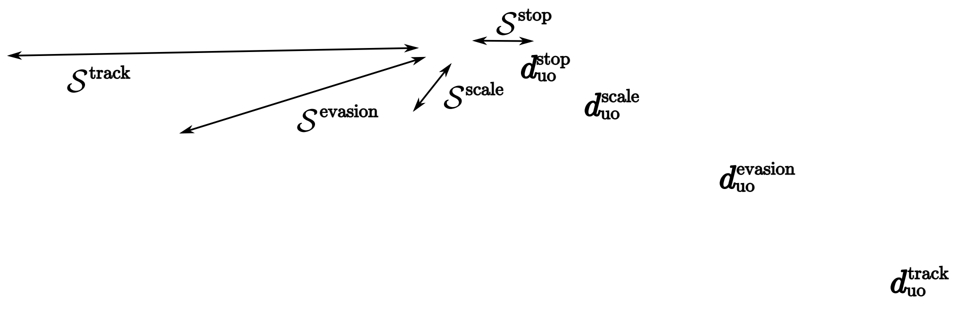

$\mathcal{S}^{\text{stop}}$
$d_{\text{uo}}^{\text{stop}}$
$\mathcal{S}^{\text{track}}$
$\mathcal{S}^{\text{scale}}$
$d_{\text{uo}}^{\text{scale}}$
$\mathcal{S}^{\text{evasion}}$
$d_{\text{uo}}^{\text{evasion}}$
$d_{\text{uo}}^{\text{track}}$

Fig. 4.4: Safety areas assuming $d_{\text{uo}}^{\text{evasion}} = d_{\text{uo}}^{\text{footstep}}$.

mented in different ways, which may not be relevant for safety purpose (e.g., control of the robot head).

Notice that, while most behaviors can be realized with the proposed scheme by appropriately activating and deactivating the relevant blocks, the *safe fall* behavior will need a separate controller which overrides the scheme in Fig. 4.5.

4.5.1 Head task generator

The Head Task Generator is in charge of generating suitable control inputs for the robot head based on information available through vision. Its input will basically depend on the *visual-tracking* behavior.

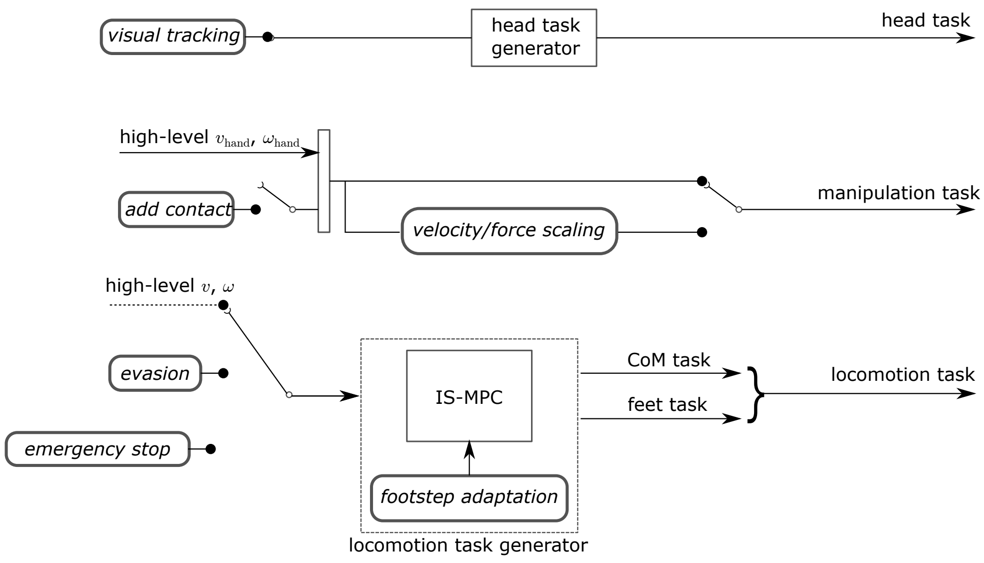


Fig. 4.5: Control architecture with the safety behaviors shown as rounded blocks.

4.5.2 Manipulation task generator

The Manipulation Task Generator provides the humanoid with a task for its hands. The *velocity/force scaling* and *add contact* behaviors may be activated to modify the high-level reference inputs.

4.5.3 Locomotion task generator

The locomotion task generator is based on the Intrinsically Stable Model Predictive Control (IS-MPC) [42] described in Sect 3.3. In particular, the module produces a trajectory of the robot CoM and feet such that balance is guaranteed at all times (by ensuring that the ZMP is at all times within the support polygon) based on the high-level motion commands and the environment map.

The proposed scheme is suitable not only for regular gaits but also for executing sudden avoidance maneuvers as well as temporary or emergency stops, all actions which are typically necessary in a safety framework.

The following chapter is dedicated to analysis and description of those behaviors related to the Locomotion task generator.

Chapter 5

Locomotion-based Safety Behaviors

In this chapter we will give the details of those behaviors directly connected to locomotion, namely the Evasion, the Footsteps Adaptation and the Emergency Stop. These behaviors, that strongly rely on the gait generation techniques proposed in Chapter 3 are one of the main contributions of this work.

5.1 Evasion

Consider a scenario where the humanoid robot is standing in a workspace, when a moving obstacle, like a human or another robot, heads towards it. The humanoid must be able to plan and execute in real time a maneuver to avoid the possible collision. Once the moving obstacle is detected, its approach direction relative to the robot is determined and, on the basis of this information an appropriate *evasion maneuver* is generated. This maneuver

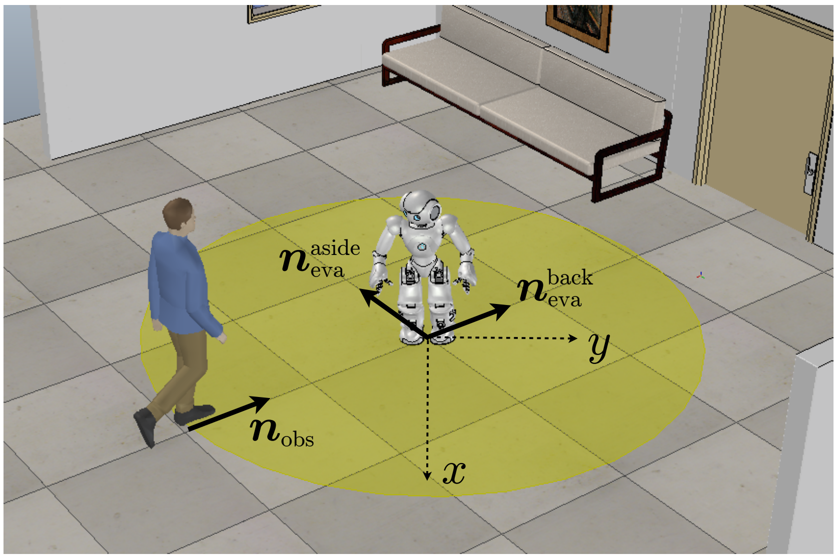

$n_{\text{eva}}^{\text{aside}}$
$n_{\text{eva}}^{\text{back}}$
y
n_{obs}
x

Fig. 5.1: The situation of interest. A moving obstacle enters the safety area of a humanoid and heads towards it. The humanoid must plan and execute a fast evasive motion. Note the moving frame associated to the humanoid.

consists of a set of footsteps and a trajectory for the robot's CoM and ZMP.

In the following we will tackle two different evasion problems: in the first, we assume that the moving obstacle heading towards the robot, keeps its direction of motion constant, meaning that the obstacle aims directly at the humanoid initial position. This assumption is relaxed in the second problem, where we consider the worst-case scenario where the obstacle is actively trying to reach and collide with the humanoid.

5.1.1 Evasion

Whenever the robot is not executing any particular task, i.e. it is in the *idle* state, and a moving obstacle enters the robot's vicinity, heading towards it, the humanoid must plan and execute a fast evasive motion in order to prevent any collision from happening [45]. The proposed method goes through several

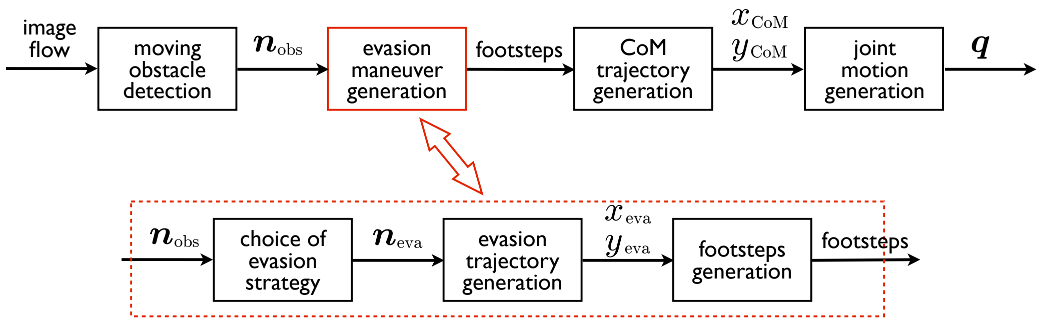


Fig. 5.2: A block scheme of the proposed approach for planning and executing evasive motions.

conceptual steps. Once the moving obstacle is detected, its approach direction relative to the robot is determined. From this information, an evasion maneuver represented by footsteps is generated using a controlled unicycle as a reference model. From the footstep sequence, we compute an appropriate trajectory for the Center of Mass of the humanoid using the ZMP-based gait generation technique of Sect. 3.2.

In the interest of safety, it is essential that the reaction time between the detection of the moving object and the beginning of the evasive motion is as small as possible. This is achieved by making use of closed-form expressions that make the overall algorithm suitable for real-time implementation. In particular, we rely on the existence of the analytical expressions relating a desired Zero Moment Point trajectory to the associated bounded Center of Mass trajectory 3.9, introduced in Sect. 3.2.

The proposed algorithm for planning and executing the evasive motion is summarized in the block scheme in Fig. 5.2. The first step is to detect the obstacle approach direction $\mathbf{n}_{obs}$ relative to the robot, as shown in Fig. 5.1. Starting from this information the robot chooses an evasion direction $\mathbf{n}_{eva}$ to align with, based on an a-priori defined evasion strategy. We considered two different evasion strategies:

- *move back:* the humanoid aligns its direction with the obstacle approach direction and moves backwards;
- *move aside:* the humanoid aligns with a direction that is orthogonal to the obstacle approach direction and moves backwards.

First, it should be noted that both strategies dictate that the humanoid moves backwards. For the *move back* strategy, this is obvious — moving forward would mean approaching the obstacle. For the *move aside*, this requirement is related to the possibility of keeping the obstacle in view, as moving backwards allows to maintain the obstacle in the half-plane in front of the robot.

The *move back* strategy is obviously aimed at maximizing the distance between the humanoid and the obstacle. In humans, this is a very instinctive reaction related to an evolutionary rooted tendency called *approach aversion* [46]. However, moving back is not sufficient to avoid collision if the obstacle moves faster than the humanoid.

The *move aside* strategy embodies a different policy, i.e., moving the humanoid as fast as possible away from the course of the obstacle. If executed sufficiently fast, this strategy may allow to avoid obstacles faster than the robot. The downside is that the distance between the humanoid and the obstacle may decrease before increasing again.

In practice, the *move back* strategy is realized by setting $\mathbf{n}_{\text{eva}} = \mathbf{n}_{\text{obs}}$, whereas *move aside* corresponds to $\mathbf{n}_{\text{eva}} = \mathbf{n}_{\text{obs}}^{\perp}$, where $\mathbf{n}_{\text{obs}}^{\perp}$ is the normal unit vector to $\mathbf{n}_{\text{obs}}$ in the half-plane behind the robot (see Fig. 5.1).

Once the evasion direction $\mathbf{n}_{\text{eva}}$ is chosen, the robot should generate the trajectory to follow in order to align to the computed direction. To do so, we use a controlled unicycle as reference model. Studies on human locomotion [47, 48] have identified a gait model in which the orientation of the body is for most of the time tangent to the path. In other words, human trajectories closely resemble those typical of nonholonomic wheeled mobile robots, such as the unicycle. This kind of viewpoint was already effectively assumed in [49]. Refer to Fig. 5.3.

The reference unicycle starts at the origin of the humanoid frame, with the same heading, and obeys the well-known model

$$\begin{aligned} \dot{x} &= v\cos\theta && (5.1)\\ \dot{y} &= v\sin\theta && (5.2)\\ \dot{\theta} &= \omega, && (5.3) \end{aligned}$$

where x, y are the unicycle Cartesian coordinates, θ is its orientation, and (v, ω) are the driving and steering velocity inputs. The following control law will align the unicycle with the desired orientation θ_{eva}, while traveling at constant speed

$$\begin{aligned} v &= \bar{v} && (5.4)\\ \omega &= k\ \text{sign}(\theta_{\text{eva}} - \theta), && (5.5) \end{aligned}$$

The desired orientation θ_{eva} is computed according to the chosen evasion strategy. In case the *move back* strategy is chosen, $\theta_{\text{eva}} = \angle \mathbf{n}_{\text{obs}}$. If the *move aside* strategy is in use, $\theta_{\text{eva}} = \angle \mathbf{n}_{\text{obs}} \pm \pi/\mathbf{2}$.

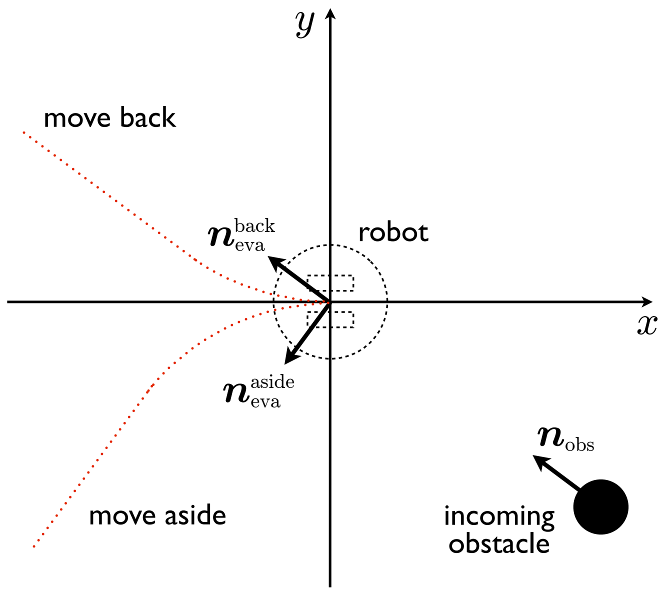

y
move back
$\boldsymbol{n}_{\text{eva}}^{\text{back}}$
robot
x
$\boldsymbol{n}_{\text{eva}}^{\text{aside}}$
$\boldsymbol{n}_{\text{obs}}$
move aside
incoming obstacle

Fig. 5.3: Generation of the evasion trajectory using a controlled unicycle as reference model.

At this point, to execute the evasive motion, the robot must generate a suitable gait: if the gait generation relies on the MPC, the reference velocities (v, ω) are directly used as input to the control scheme shown in Fig. 3.3. On the other side, in order to plan the robot CoM trajectory starting from a reference ZMP trajectory as described in Sect. 3.2, a set of footsteps must be provided.

To generate this set of footsteps, the unicycle model (5.1-5.3) is integrated under the control law (5.4-5.5) leading to a Cartesian trajectory that proceeds backwards along an arc of circle until the desired orientation θ_{eva} is achieved at the finite time instant $t_s = |\theta_{\text{eva}}|/k$ and then becomes a straight line:

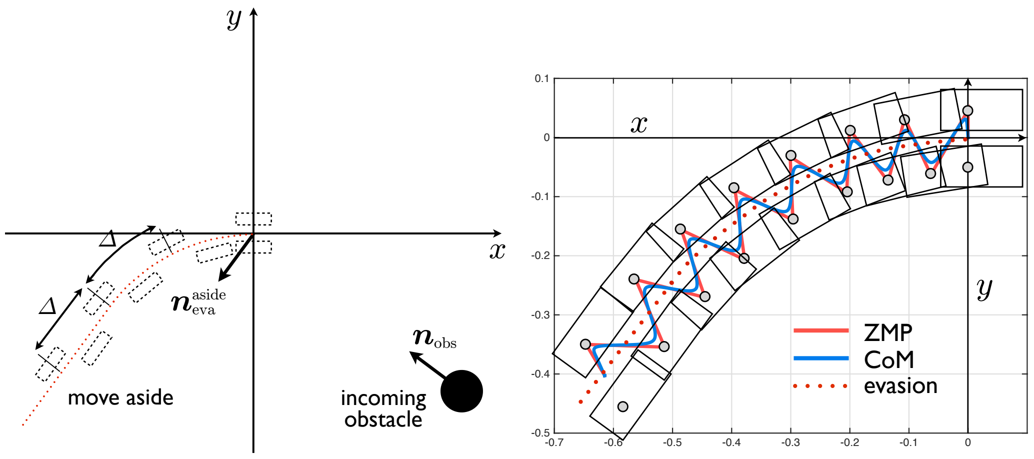


Fig. 5.4: Footstep placement around the evasion trajectory (left). Planned CoM-ZMP trajectory (right).

$$x(t) = \bar{v}\,\frac{\sin kt}{k} \tag{5.6}$$

$$y(t) = \operatorname{sign}(\theta_{\text{eva}})\,\bar{v}\,\frac{1-\cos kt}{k} \tag{5.7}$$

$$\theta(t) = \operatorname{sign}(\theta_{\text{eva}})\,kt \tag{5.8}$$

for $t \le t_s$ and

$$x(t) = x(t_s) + \bar{v}(t-t_s)\cos\theta_{\text{eva}} \tag{5.9}$$

$$y(t) = y(t_s) + \bar{v}(t-t_s)\sin\theta_{\text{eva}} \tag{5.10}$$

$$\theta(t) = \theta_{\text{eva}} \tag{5.11}$$

for $t > t_s$.

Figure 5.3 shows the trajectories associated to the two different evasion strategies. Once the Cartesian trajectory is computed, the sequence of footsteps is sampled from it using a constant time interval $\Delta t = \Delta/|\bar{v}|$ where Δ is a constant step size. This is realized by displacing the x, y, θ samples alternatively to the left and to the right of the trajectory, as depicted in Fig. 5.4, left. Figure 5.4, right, shows the footsteps and the associated CoM-ZMP trajectories for a *move aside* evasion maneuver

5.1.2 Evasion with malicious moving obstacle

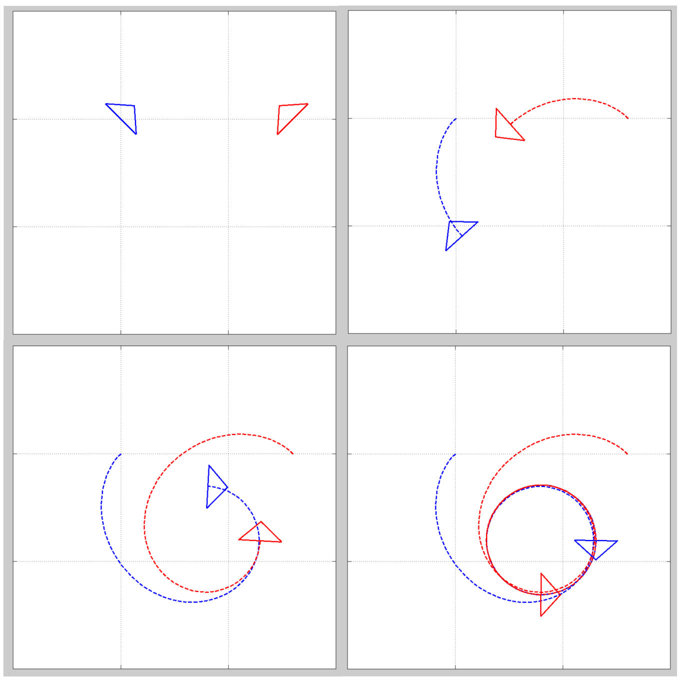


Fig. 5.5: Pursuit-evasion with unicycles: simulation for the pursuer (red) and for the evader (blue). Axis ticks are 0.5 m apart.

In this section we relax the assumption that the moving obstacle keeps constant its direction of motion, and consider the worst-case scenario where the obstacle is actively trying to reach and collide with the humanoid. This leads us to replace the moving obstacle with another humanoid, and to consider therefore a full-fledged pursuit-evasion problem with humanoids [50]. Our viewpoint is to consider a coupled dynamic system consisting of two identical humanoids with equivalent control schemes but different objectives: the pursuer tries to align with the line-of-sight to the evader, whereas the latter attempts to move away from the line-of-sight to the pursuer, e.g., in a direction orthogonal to it.

As before, our method is based on the use of the unicycle as a template model for real-time trajectory generation. For this reason, in this section we will discuss the pursuit-evasion problem with unicycles. The results obtained with humanoids are presented in the dedicated Chapter 7.

Both the pursuer and the evader are controlled in pure feedback mode. At any instant, the pursuer determines the line-of-sight to the evader, represented by the unit vector $\mathbf{n}_{\text{aim}}$, and steers its course so as to align with $\mathbf{n}_{\text{aim}}$. The evader determines the line-of-sight to the pursuer, represented by $-\mathbf{n}_{\text{aim}}$, computes from this an evasion direction $\mathbf{n}_{\text{eva}}$, and steers its course so as to align with $\mathbf{n}_{\text{eva}}$. Figure 5.6 depicts the geometry of the problem.

The evader is controlled with the same control laws (5.4-5.5) presented in the previous section. The pursuer's control law, has a different objective, i.e. align the unicyle with the line-of-sight to the evader $\mathbf{n}_{\text{aim}}$. Hence for the pursuer eq. 5.5 becomes

$$\omega = k\text{sign}(\theta_{\text{aim}} - \theta) \tag{5.12}$$

where $\theta_{\text{aim}} = \angle\mathbf{n}_{\text{aim}}$.

The gait generation process for the humanoids is the same of the previous

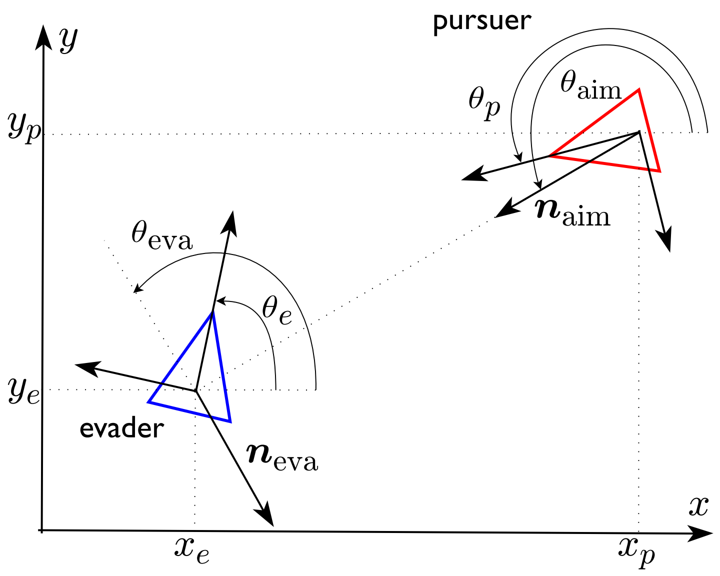


Fig. 5.6: Pursuit-evasion with unicycles: geometry of the problem

section. The coupled system interestingly converges to a limit cyle, where the two unicycles travel on a circle with a relative orientation of $\pi/2$, as shown in Fig. 5.5.

5.2 Footstep Adaptation

As seen in Chapter 3, at the end of the gait generation process, the robot is provided with a sequence of footsteps and a trajectory for its CoM and ZMP. However, the planned footsteps may become invalid because of some external disturbances that have not been taken into account, such as unmapped objects. The footstep adaptation behavior is activated whenever an unexpected object enters in the robot path and allows the robot to slightly modify the current footstep plan (and the associated CoM and ZMP trajectory) to avoid the obstacle and complete its locomotion task. We show how the humanoid can adapt its footsteps while walking to avoid obstacles, without interfering with its main locomotion task [51].

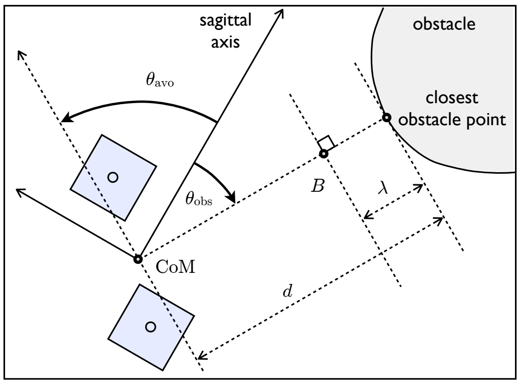


Fig. 5.7: Definition of the relevant quantities. The current robot placement is defined by its CoM: current footstep locations are also shown (light blue).

Consider the MPC-based gait generation strategy presented in Sect. 3.3. The robot is asked to track reference velocities (v_x, v_y, ω), and the MPC scheme provides the sequence of footsteps and the appropriate trajectory for the CoM and the ZMP. If, while walking, it detects the presence of an obstacle, it must take it into account during the planning of the optimal trajectories and footstep positions. To do so, we modify the MPC by changing the cost function and by adding a new constraint.

The cost function that we modify is the one in eq. (3.11), i.e. the cost function that decides the optimal foot rotations to track the reference angular velocity. We add a term that forces the robot to align its footsteps with the tangent of the closest obstacle point, so the new expression of the cost function becomes

$$\sum_{j=1}^{M} \left(\left(\frac{\theta_j - \theta_{j-1}}{T_s} - \omega \right)^2 + k_{\text{obs}} \frac{w(\theta_{\text{obs}})}{d^2} (\theta_j - \theta_{\text{avo}})^2 \right). \tag{5.13}$$

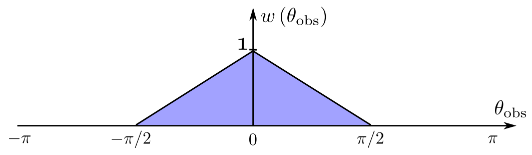


Fig. 5.8: The weight function $w(\theta_{\mathrm{obs}})$.

In the second term of (5.13) the difference in orientation between θ_j and θ_{avo} is penalized. $\theta_{\mathrm{avo}} = \theta_{\mathrm{obs}} \pm \pi/2$ is the orientation of the tangent half-line originating at the closest obstacle point, and θ_{obs} is the angle between the robot and the obstacle. Figure 5.7 gives a graphic representation of these quantities. Note that the second term is modulated through a scaling factor k_{obs} by a weight function $w(\theta_{\mathrm{obs}})$ and the inverse of the squared distance.

Figure 5.8 shows the definition of function w. The idea here is that the robot moves forward, and therefore only obstacles lying in its front half-plane should be considered.

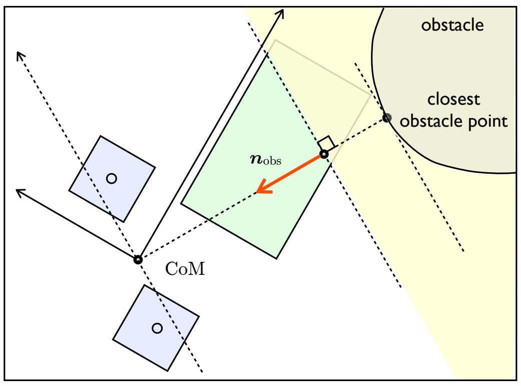


Fig. 5.9: Kinematically feasible zone (green) and forbidden zone (yellow) for the next footstep as defined in the MPC.

As a stronger level of safety, besides the new term in the cost function, we also introduce a constraint on the footstep locations, that depends on the obstacle position. With reference to fig. 5.7, consider a point $B = (x_B, y_B)$

located along the line connecting the CoM with the closest obstacle point at a safety distance λ from the latter, and draw the normal to the same line through B. The half-plane beyond this line (in yellow in Fig. 5.9, bottom) is a forbidden zone for the footstep locations. This constraint is easily written as

$$\mathbf{n}_{\text{obs}}^{\mathbf{T}} \left\{ \begin{pmatrix} x_{\text{f}}^{j} \\ y_{\text{f}}^{j} \end{pmatrix} - \begin{pmatrix} x_B \\ y_B \end{pmatrix} \right\} \geq \mathbf{0} \tag{5.14}$$

with $\mathbf{n}_{\text{obs}}$ the unit vector defined in Fig. 5.9.

With this modified version of the MPC, the robot will do its best to track the reference angular velocity and align to the direction tangent to the closest obstacle point. This should be enough to drive the robot away from the closest obstacle. In any case, if the robot gets too close to the object, the constraint will make impossible for the robot to step any further towards the obstacle, as shown in fig. 5.10.

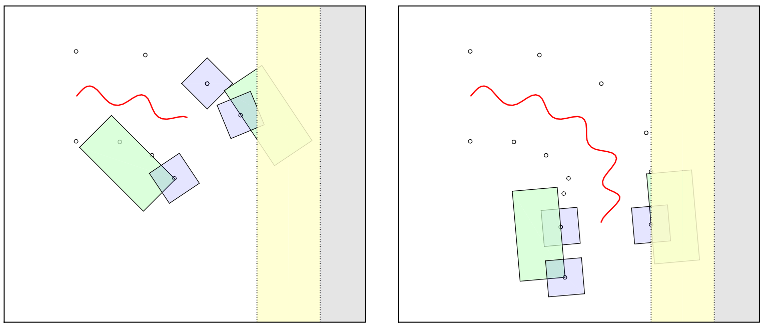


Fig. 5.10: Avoidance of a wall (shown in gray) with the proposed gait generation method. The CoM trajectory is shown in red.

5.3 Emergency Stop

While the robot is walking, an unforeseen event may rise the necessity to stop the robot's locomotion as soon as possible. In the case of an emergency stop, we want to reach a statically balanced configuration, i.e. we want both the ZMP and the CoM stationary inside the support polygon.

The fastest and safest way to reach a complete stop is to immediately set to zero the reference velocities (v_x, v_y, ω_r). In this way the MPC will generate a feasible and stable trajectory for the CoM such that to stop.

Another possible option relies on the concept of capture point x_{cp} [30]

$$x_{cp} = x_c + \frac{\dot{x}_c}{\omega}. \tag{5.15}$$

The capture point, is the point where the robot should step in order to reach a complete stop. Its position depends on the robot CoM velocity, so the fastest the robot is walking, the further is the capture point. If it is in a region (capture region) inside the robot workspace, this means that the robot can reach the capture point in one step (1-step capturability). If the capture point is already inside the support polygon (0-step capturability) the robot can immediately stop without stepping.

This concept should be taken into account when designing an emergency stop behavior. In our case, the capture point position is continuously checked, and if it lies inside the current support polygon the robot is able to immediately stop without stepping (if needed it can also stop with one foot hanging in the air). If on the other side the current robot velocity is too high and the capture point is not in the support polygon, the robot will have to take at least one step before stopping. Hence, setting to zero the reference velocity leads to a choice of the next step such that the robot will reduce as much as possible its CoM velocity, henceforth bringing the capture point inside the support area and finally stop.

Chapter 6

Other Safety Behaviors

The remaining behaviors introduced in 4.2 are described in this chapter. Those behaviors are not directly connected to locomotion, and we will analyze possible solutions for the implementations of the behaviors referring to existing works.

6.1 Safe Fall

The stepping capabilities of a humanoid robot are for sure one of their strengths when compared to locomotion abilities of wheeled robot, since humanoids can overcome obstacles or climb stairs through stepping. This however arises the problem of balancing. In the definition of a framework for safe human-humanoid coexistence, a situation in which the robot loses balance cannot be excluded, indeed it must be faced rigorously. When the humanoid robot loses its balance, it becomes an extremely dangerous, uncontrolled object that can cause serious damages to itself and the environment and injuries to people. A safe fall behavior has the aim of minimizing the dramatic consequences of a loss of balance by controlling the robot during the fall, trying to choose properly how and where to fall.

Choosing how to fall means assuming a proper configuration to absorb the impact with the floor, or plan a sequence of motions to distribute the impact along the kinematic chains of the robot. In [52] a falling motion controller that allows the robot to safely fall over backwards is presented, and is then extended in [53] to tackle the problem of falling forward. The idea is to lower the robot CoM as soon as possible, via crouching or knee-bending to restrain the force of impact. Authors in [54] face the problem of fall management in a similar way. According to the detected direction of fall they propose two control strategies to fall forward (knee-bending scheme) or to fall on the robot back.

Assuming a configuration that reduces the impact magnitude might not be enough, and some strategies might be useful to absorb the impact. In [44] a control strategy that combines robot reconfiguration and post-impact compliance is presented. During the falling phase the robot posture is controlled to avoid fall singularities, i.e. particular configurations that will not absorb properly the impact force. After the impact occurs, instead of shutting the actuators or imposing very high stiffness to the joints, an active compliant control is used to absorb the impact, with the motors behaving like spring-dumpers.

Other than how to fall, the robot should be able to choose where to fall, in order to avoid falling on people or other objects, and maybe exploit the environment to partially recover from the fall. [55] presents a fall controller capable of changing the fall direction of the robot such that the robot does not hit a particular object in the environment. This is further extended to multiple objects avoidance [56] and finally implemented on a real humanoid [57].

6.2 Visual Tracking

Through vision sensors like RGB-cameras a robot is able to obtain geometrical informations on the environment. These information, useful for motion planning and control, are obtained after image processing, i.e. a procedure to extract numerical information (features) from an image. An important measurement that can be derived from image processing, is the position of objects in the environment w.r.t. the camera.

The image processing starts form the so-called image segmentation. It consists in the partitioning of the image into multiple segments, i.e. clusters of pixels that share some characteristics. Starting from the segmented image, features can be extracted to obtain numerical information that can be used for motion planning and control.

The visual tracking behavior has the objective of controlling the robot gaze in order to keep it aware of a particular object detected in the environment (e.g. a moving object). The image acquired from the camera must be processed in order to extract the feature parameters and identify any unexpected object in the image. Then this features, representing the object, must be controlled to be kept inside the robot field-of-view. This is a typical visual servoing control problem, meaning that the control action is computed on the basis of visual measurements. In more details, we can consider this as an image-space visual servoing problem [58]: the control action is computed according to an error in the desired position of a feature directly in the image plane.

The visual tracking problem has been widely studied in the past years, and several solutions can be found in literature with applications to humanoid robots [59, 60, 61, 62].

Clearly, exploring the computer vision literature, it is easy to find several approaches to image processing for human detection and tracking [63].

6.3 Velocity/Force Scaling

A robot hitting a person is a clear hazard and, as well as for the falling problem, we cannot exclude the possibility of a collision between the humanoid and a human in the environment. The idea behind the Velocity/Force scaling behavior is to reduce the effect of a collision, by reducing the forces and velocities of the humanoid. If the robot is in the *manipulation* state, and a human gets too close ($d_{uo} < d_{uo}^{scale}$), it can straightforwardly scale down the velocities/forces of the joints in the kinematic chains involved in the manipulation task.

Although the reduction of velocities and torques at the joint level does not harm the normal operations of a manipulator or a wheeled mobile base, in the case of a walking humanoid this can be more dangerous than helpful. The problem of keeping balance is crucial and an abrupt reduction of the joint velocities/forces may lead to a fall. For this reason we need to face the

Velocity/Force problem at the Cartesian level, and let the walking pattern generator handle it.

In our MPC scheme, the robot is controlled through reference Cartesian velocities, and the optimization problem takes care of generating the proper gait. The reference velocity can be arbitrarily low (it can also be zeroed, as seen in 5.3), since the MPC will always find a feasible solution that will keep the robot balanced, thanks to the constraint enforced in the QP problem.

6.4 Add Contact

Humanoid robots are designed with the idea of having abilities similar to human skills in both locomotion and manipulation, and these two problems should not be considered separately. In fact, a humanoid robot walking in a cluttered environment, just like a human, can increase its locomotion capabilities by establishing extra contacts to achieve a more robust balance.

The issues to consider are basically two, i.e. how to balance and control the robot in a multi-contact configuration and where to establish contacts.

For the first problem, one must understand the interactions between the robot and the environment, and the consequences on the robot internal forces and CoM [64]. Also, the ZMP concept needs to be extended to redefine the idea of dynamic balance [65]. The balance and posture control problem is tackled in [66] relying on optimization of contact forces in combination with Model Predictive Control. In [67] a passivity-based whole-body controller is presented, that allows the robot to balance with a subset of its end effectors and use the remaining to interact with the environment, through task-hierarchy.

The goal of multi-contact planning is to generate a sequence of robot configurations that use not only the end effectors (hands and feet) to interact with the environment but the whole robot body (e.g. knees and elbows). In [68] an overview of how humans use multiple contacts to perform various locomotion and manipulation tasks is presented, while authors in [69] present a planner that generates robot motions to perform transitions between multi-contact stances. [70] propose a complete algorithm that starting from visual information allows the robot to define step locations, compute feasible multi-contact stances using hands and feet, and generate a motion plan to reach the desired goal even going through different contact states.

Clearly, multi-contact planning comes in handy also when planning dynamic motions for the humanoid [71, 72], however in our safety framework, we only want to consider the possibility of establishing extra contacts to enhance the robot balance during the execution of tasks that do not require stepping, like manipulation. It is evident that when considering the problem of stair climbing, the idea to grasp the handrail for additional support must be taken into account directly at the planning level, hence will not need an adaptation of the task.

Chapter 7

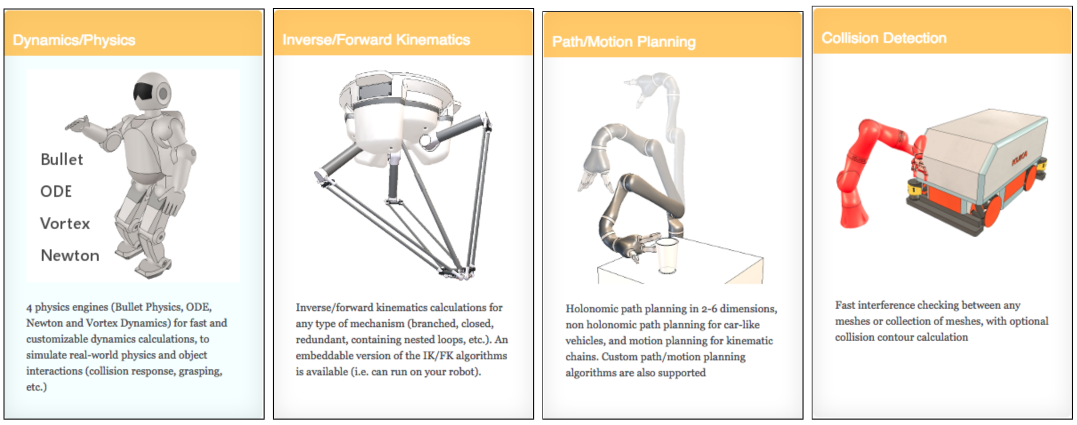
Results

This chapter is dedicated to the presentation of the simulation and experimental results. The simulations have been performed in the V-REP simulation environment, while the experiments were conducted with two different robotic platforms: the NAO humanoid robot and the HRP-4.

V-REP (Virtual Robot Experimentation Platform) is a powerful, cross-platform 3D robot simulator developed by Coppelia Robotics[1]. It has an integrated development environment (IDE) and it is based on a distributed control architecture: each object or model present in the scene can be controlled individually by an embedded script, a plugin or a ROS node. Controllers can be written in many different programming languages such as C/C++, Python, Lua and MATLAB. It is commonly used for fast algorithm development, factory automation simulations and fast prototyping and verification. It features different calculation methods specific for robotics development, like Inverse and Forward Kinematics, Collision Detection, Path/-

[1]www.coppeliarobotics.com

Motion Planning and Vision Sensor simulation.



Dynamics/Physics

4 physics engines (Bullet Physics, ODE, Newton and Vortex Dynamics) for fast and customizable dynamics calculations, to simulate real-world physics and object interactions (collision response, grasping, etc.)

Inverse/Forward Kinematics

Inverse/forward kinematics calculations for any type of mechanism (branched, closed, redundant, containing nested loops, etc.). An embeddable version of the IK/FK algorithms is available (i.e. can run on your robot).

Path/Motion Planning

Holonomic path planning in 2-6 dimensions, non holonomic path planning for car-like vehicles, and motion planning for kinematic chains. Custom path/motion planning algorithms are also supported

Collision Detection

Fast interference checking between any meshes or collection of meshes, with optional collision contour calculation

Fig. 7.1: Some V-REP Features

The main aspect is that V-REP features 4 different physics engine:

- Bullet physics library (open source): features 3D collision detection, rigid and soft body dynamics;
- Open Dynamics Engine (ODE): it has two main components, rigid body dynamics and collision detection;
- Vortex Dyanmics: commercial physics engine producing high fidelity physics simulations, mainly used in high performance/precisions industrial and research applications;
- Newton Dynamics: it implements a deterministic solver not based on LCP (Linear Complementary Problem) or iterative methods.

The dynamics engine allows simulating interaction between objects, allowing them to fall, collide, bounce ecc. Moreover V-REP provides many real robot models both mobile and non-mobile.

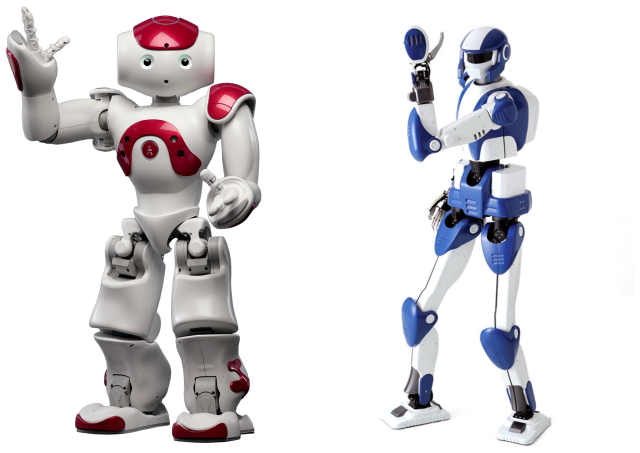


Fig. 7.2: NAO (left) and HRP-4 (right)

NAO is a small autonomous and programmable humanoid robot with 25 degrees of freedom (DoFs). It is 57.3 cm tall and its weight is 4.3 kg. It is equipped with an Intel Atom CPU running at 1.6 GHz and with several on-board sensors like cameras, microphones, sonars, pressure sensors, tactile sensors, IMU.

HRP-4 is a life-size "platform for research and development of working humanoid robots" developed by KAWADA industries in collaboration with the National Institute of Advanced Industrial Science and Technology (AIST) in Japan. HRP-4 is a light-weight robot designed to coexist with humans and assist or replace human operations or behaviors. It is 151 cm tall and weights 39 kg. It has 34 DoFs and is equipped with an Intel Pentium M runniung at 1.6 GHz.

In the following we will show the results obtained in the implementation of the safety behaviors detailed in Chapter 5 as well as a simulation of the complete framework.

7.1 Evasion

The evasion behavior presented in Chapter 5 has been validated via simulation and experiments using the NAO humanoid robot. The result of the simulation is shown in Fig. 7.3. Here the gait generation is based on the ZMP planning described in 3.2.

The humanoid is standing at the center of a room when a human walks in (1st snapshot), directed towards the desk on the right. As the human enters the safety area of the humanoid (2nd snapshot), the latter detects this fact and generates an evasive motion using the *move aside* strategy. In particular, an evasion trajectory is first generated using the controlled unicycle model (5.1-5.3), with $\bar{v} = 0.04$ m/s and $k = 0.2$ in eq. (5.4-5.5). Footsteps are placed around this trajectory, using a stepsize $\Delta = 0.08$ m, in the range of NAO capabilities. Then, a ZMP trajectory interpolating the footsteps is computed, with duration of the double and single support respectively at 0.122 s and 0.425 s. The corresponding bounded CoM trajectory is generated using 3.9. Finally, joint commands are produced via kinematic control. Overall, the achieved response time (between the detection of the moving obstacle entering the safety area and availability of the first joint command) is around 21 ms, confirming that the use of closed-form expressions in all stages of motion generation makes real-time evasion possible. As a result, the humanoid performs a successful *move aside* evasion (3rd to 5th snapshot). The simulation continues with another human crossing the room (6th snapshot) towards the door, and the robot executing another evasive motion (7th to 9th snapshot).

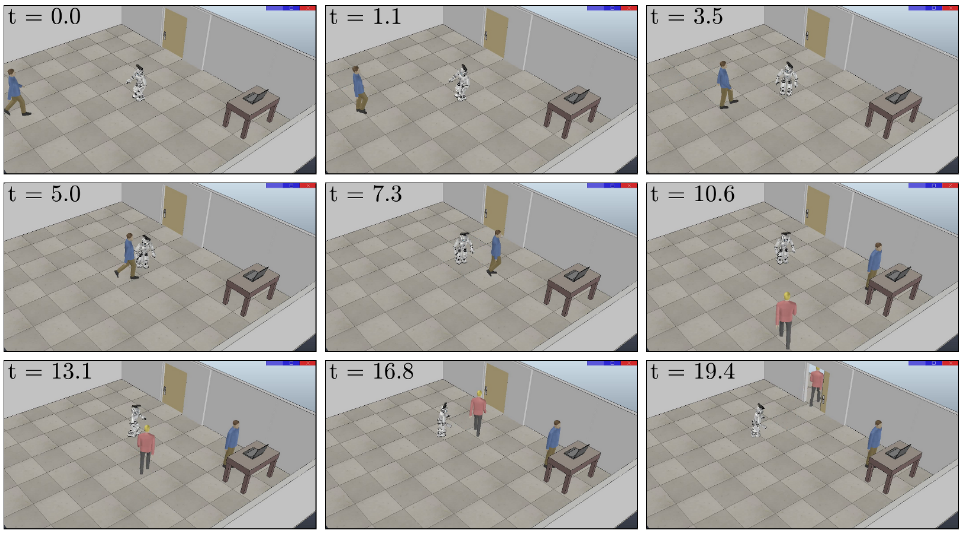


Fig. 7.3: Evasive motions using the *move aside* strategy: snapshots from a simulation.

Experimental validation was performed using two NAOs, one acting as the robot and the other (teleoperated) as the moving human. The result is shown in Fig. 7.4. Generation of the evasive motion is performed on-board, using the same parameters of the previous simulation. As expected, a *move aside* evasive motion is successfully executed.

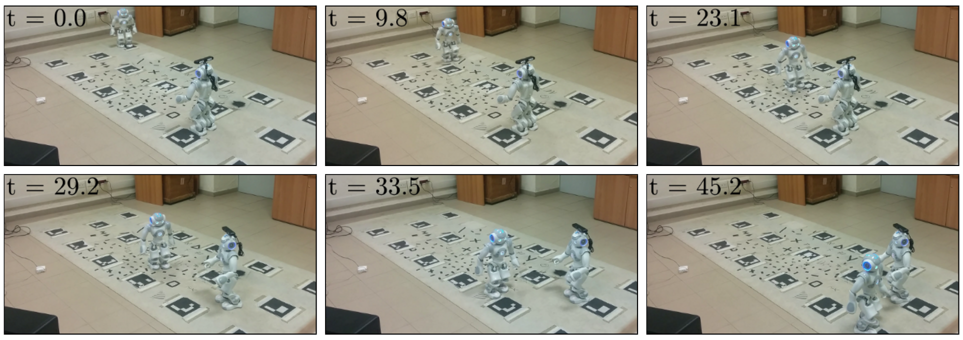


Fig. 7.4: Evasive motion using the *move aside* strategy: snapshots from an experiment.

The situation of malicious moving obstacle presented in 5.1.2 has been validated as well using two NAO humanoids. The results fully confirm the

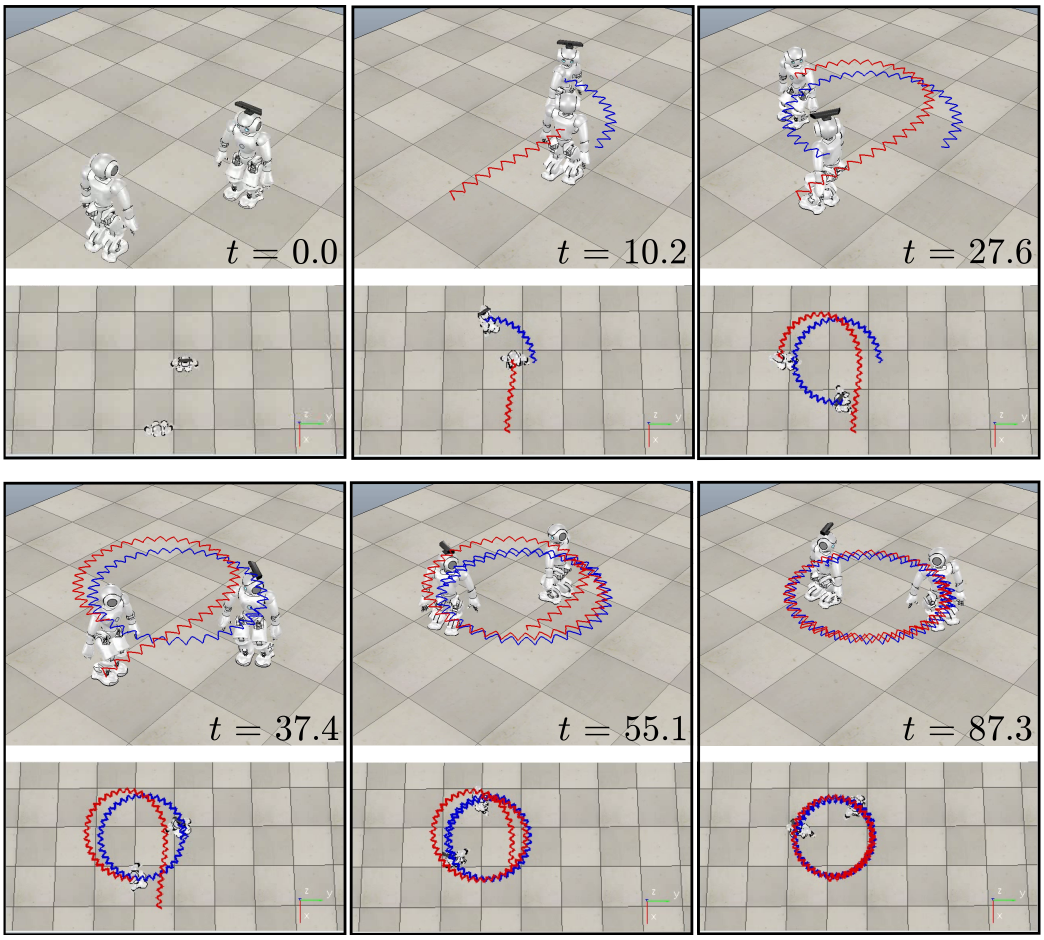


Fig. 7.5: Pursuit-evasion with humanoids: snapshots from a simulation. The trajectories of the CoMs are shown in red (pursuer) and blue (evader).

pursuit-evasion behavior introduced in Sect. 5.1.2: the two robots converge to a circular limit cycle, along which they travel at the same speed with a relative orientation of $\pi/2$.

For the experiments, in spite of the rather limited processing capabilities (each NAO is equipped with an Intel Atom running at 1.6 GHz) we were able to perform all computations on-board.

Figure 7.6 shows snapshots taken during an experiment. The expected limit cycle behavior is observed again, although its radius is slightly reduced

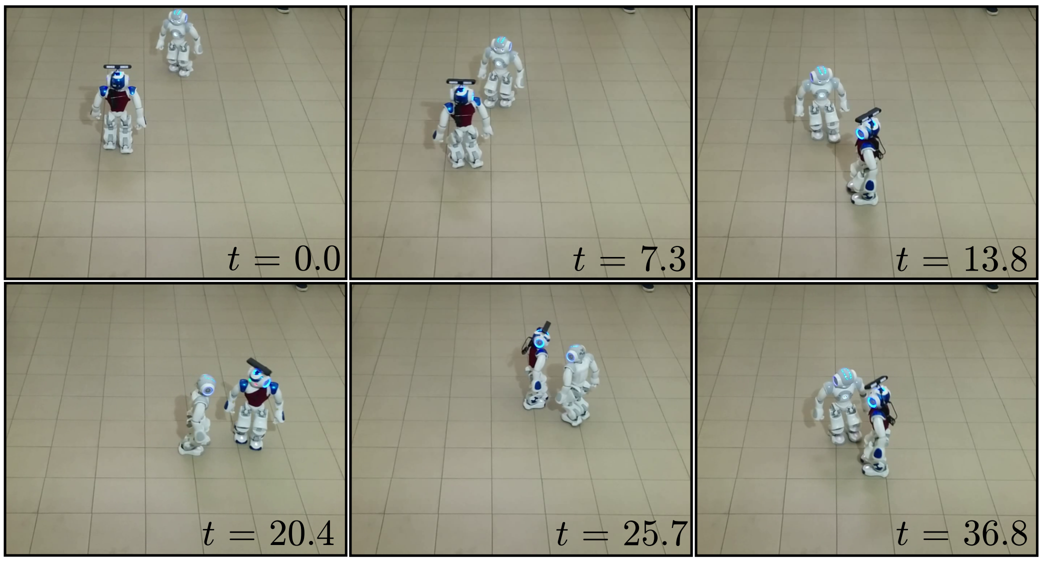


Fig. 7.6: Pursuit-evasion with humanoids: snapshots from an experiment.

with respect to the simulation. This is mainly due to the significant feet slippage on the smooth floor.

7.2 Footstep Adaptation

To evaluate the footstep adaptation behavior presented in Sect. 5.2, we started from the pursuit-evasion problem showcased in sect. 5.1.2, and we extended the problem by including obstacles in the environment. We also changed the gait generation strategy, preferring the MPC-based method described in sect. 3.3.

The proposed approach was validated in V-REP simulations using two NAO humanoids, one acting as the pursuer and the other as the evader. High-level reference velocities are produced by (5.4–5.5) with $\bar{v} = 0.1$ m/s and $k = 0.2$. Footsteps rotation is performed using $k_{\text{obs}} = 0.05$ and $\theta_{\max} =$

$\pi/16$. For MPC, we set $k_{\text{vel}}^x = k_{\text{vel}}^x = 10$ in the cost function (3.12) and used $\delta = 10$ ms over a prediction horizon $T_h = 0.6$ s. The duration of the single and double support phase is fixed at respectively 0.2 s and 0.1 s. For the ZMP bounding box we used $x_{\text{z}}^{\max} = y_{\text{z}}^{\max} = 0.02$ m, and for the feasibility constraints $L = 0.125$ m, $x_{\text{f}}^{\max} = 0.05$ m, $y_{\text{f}}^{\max} = 0.025$ m. Overall, the control scheme runs at 100 Hz and can be implemented on the NAO hardware. Figure 7.7 shows pursuit-evasion in an environment containing a

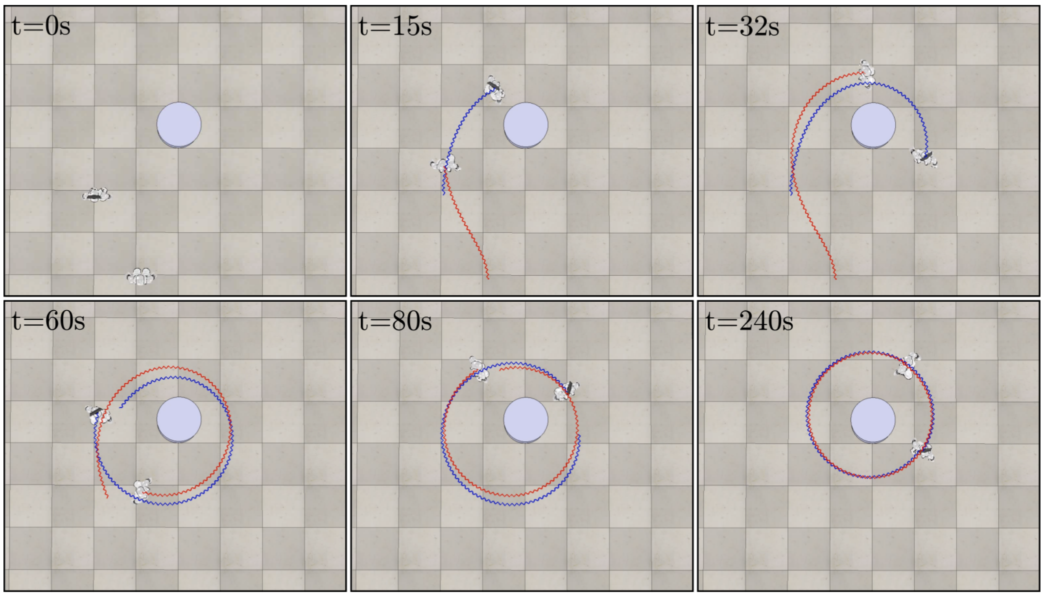


Fig. 7.7: Simulation 1: Pursuit-evasion in the presence of a cylindrical obstacle. COM trajectories are shown in red (pursuer) and blue (evader).

single cylindrical obstacle. In this case, the humanoids converge to a limit cycle around the obstacle. The effect of the latter can be appreciated in Fig. 7.8, where the motion of the robots is shown both in the absence and the presence of the obstacle. Note how the trajectory deformation is smooth thanks to the modulation mechanism with d used in the footsteps rotation module.

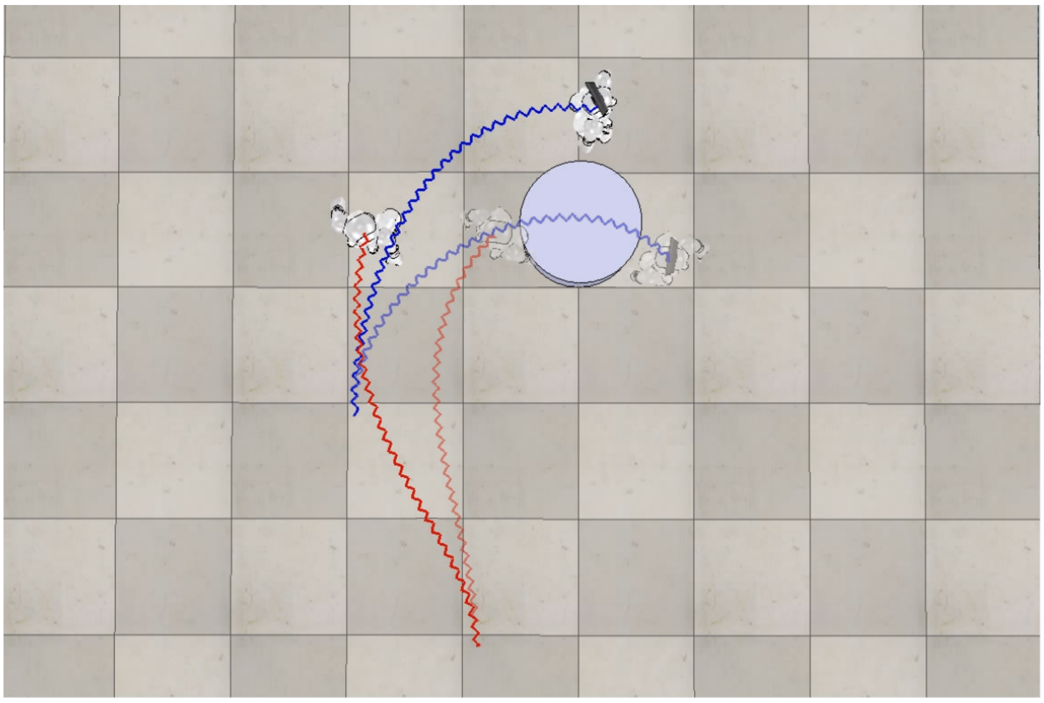


Fig. 7.8: Simulation 1: Comparison of motions with and without the obstacle.

In the second simulation, shown in Fig. 7.9, the environment obstacle is a long wall. At the beginning, the humanoids move as in the previous simulation and tend to align with the wall. In this case, however, the combination of the pursuit-evasion and the obstacle avoidance actions ultimately drives the robots away from the obstacle.

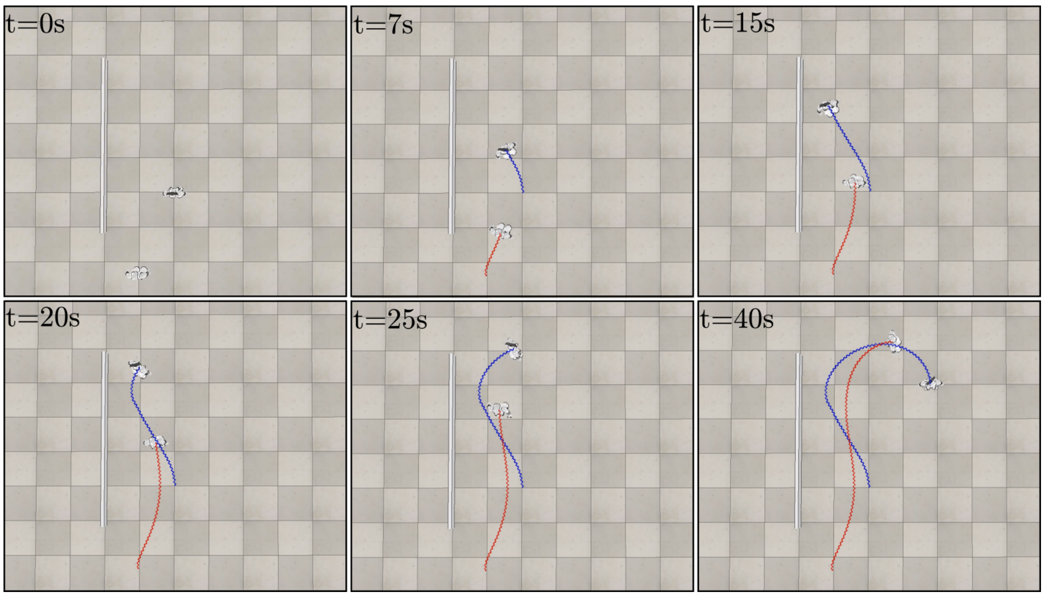


Fig. 7.9: Simulation 2: Pursuit-evasion in the presence of a wall obstacle. CoM trajectories are shown in red (pursuer) and blue (evader).

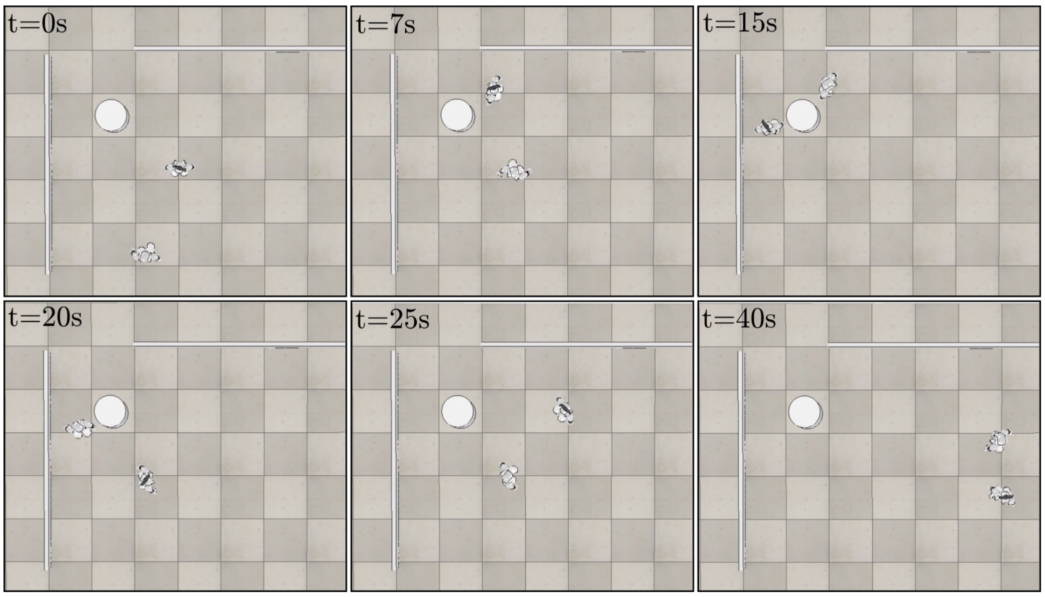


Fig. 7.10: Simulation 3: Pursuit-evasion in a complex environment.

Pursuit-evasion in an environment with several obstacles is simulated in

Fig. 7.10. In spite of the more complex geometry, the humanoids always manage to avoid the obstacles, with the pursuer aggressively chasing the evader but the latter always escaping.

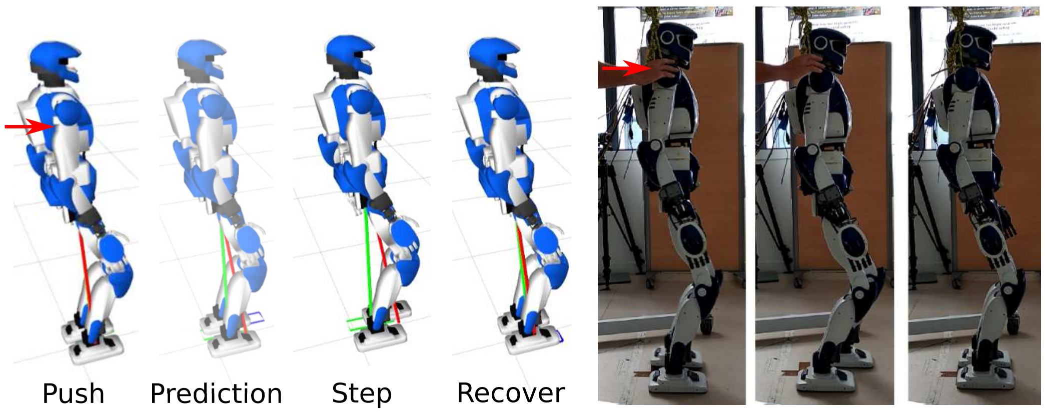


Fig. 7.11: Reaction to a push during a real experiment with HRP-4. During the push, the ZMP-CoM measurements reaches the edge of the foot (red line) while the robot's CoM is displaced forward. As a result, the MPC computes a new optimal footstep (blue square) and computes a footstep trajectory to bring its next swinging foot there.

Another interesting footstep adaptation behavior emerges from the intrinsic stability property of the MPC gait generation process. Since the MPC is always computing the optimal solution starting from the current robot state, it is able to fastly react to external perturbations like a push. In other words, if the robot is pushed, the MPC perceives a change in the robot state, and reacts to recover the loss of balance by stepping, hence adapting the robot current footsteps to counter the perturbation. To validate this behavior we let the robot step in place, and then we pushed it from behind. In order to not lose balance, due to the ZMP contraint (3.13) enforced in the MPC, the robot is forced to step forward, as shown in fig. 7.11.

7.3 Emergency Stop

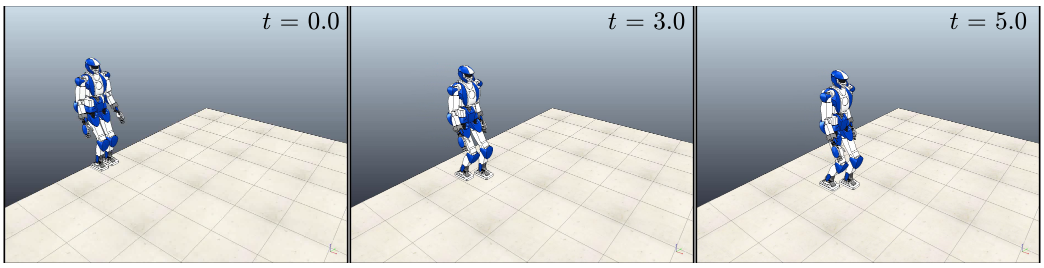


Fig. 7.12: Simulation of an emergency stop in double support

Here we present the results of the emergency stop behavior described in Sect. 5.3. As said, while walking the robot continuously checks the position of the capture point (5.15), and when the emergency stop signal is triggered, the robot performs an emergency stop according to its position.

If the capture point is inside the support polygon, the robot can immediately stop, without the need of taking additional steps. Figure 7.12 shows the snapshots of a simulation in which the emergency is triggered during the robot double support phase at $t = 5$s. Since at this time instant the capture point is inside the convex hull of the two feet, the robot is able to immediately stop. In fig. 7.13 is shown the complementary situation: the emergency stop signal is triggered during the single support phase at time $t = 7$s. However, since the robot speed is sufficiently low ($v_x = 0.2$m/s), the capture point lies inside the current support foot and the robot is able to stop in single support. After stopping with a foot hanging in the air, in a statically balanced configuration, the robot plans a motion to move the swinging foot back to where it was at the start of the step. This shows how the robot can, for instance, stop if anything unexpected appears along its path, and the only thing that

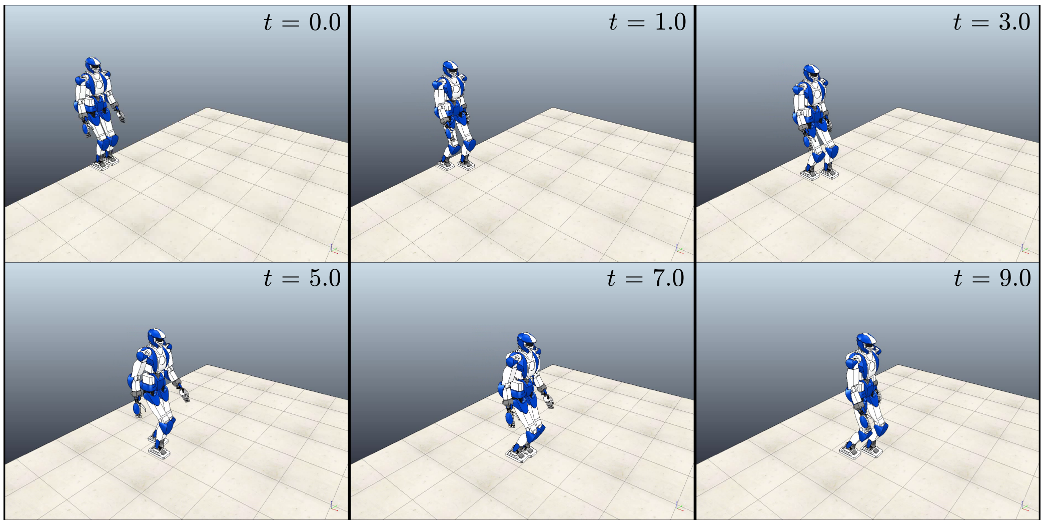


Fig. 7.13: Simulation of an emergency stop in single support

can be done is to step back. Finally, the snapshots of Fig. 7.14 show a sim-

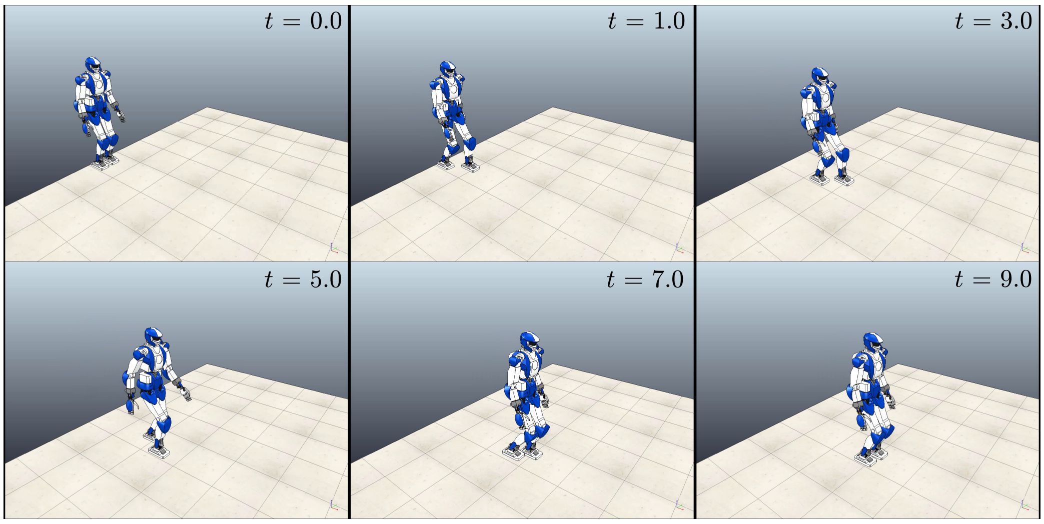


Fig. 7.14: Simulation of an emergency stop where the reference velocity is set to zero

ulation where the robot is walking at higher speed ($v_x = 0.3$m/s), when the emergency stop is triggered a time $t = 5$s. Due to the high CoM velocity, the robot cannot stop without taking one more step, so the reference velocity is immediately set to zero. This leads the robot to take one step ($t = 7$s), and finally come to a complete stop. A video of the simulations can be found at link[2].

7.4 Framework Simulation

A demo of the framework is presented here. In this simulation, we put the robot in condition to test the state machine. According to the robot states and perception, the state machine chooses the appropriate behavior. Snapshots of the simulations are depicted in Fig. 7.15 .

The robot is in the *idle* state and, following the *be on the lookout* guideline scans the environment with the camera equipped on its head. Suddenly a human gets close enough to trigger the visual tracking behavior (i.e. the human enters the largest red circle around the robot), so the robots start tracking it with its head (snapshots 1-2). At time $t = 19$s the human starts moving toward the robot, triggering the evasion behavior. The evasion maneuver lasts as long as the human is far enough ($d_{uo} > d_{uo}^{evasion}$) at $t = 30$s. The state is switched back to *idle* and, since the human is still close to the robot, the visual tracking behavior is triggered again, until the human completely exits the largest red circle at $t = 34$s.

Then, the humanoid is commanded to reach a specific goal in the environment, so it switches to the *locomotion* state heading toward it ($t = 70$s). However an obstacle is on its path, and as soon as the robot gets closer to it, the footstep adaptation behavior starts, and the robot deviates from its

[2] $https://drive.google.com/file/d/1uRcEHmMCqaOzzLGjEQ2smqKl_1bk97JH/view?usp = sharing$

path to avoid the collision ($t = 90$s).

Once the robot reaches the goal at at $t = 140$s, the reference velocity is set to zero and the robot safely stops. The robot state is switched again to *idle* and the robot starts again to scan the environment.

A video of this simulation is available at link[3].

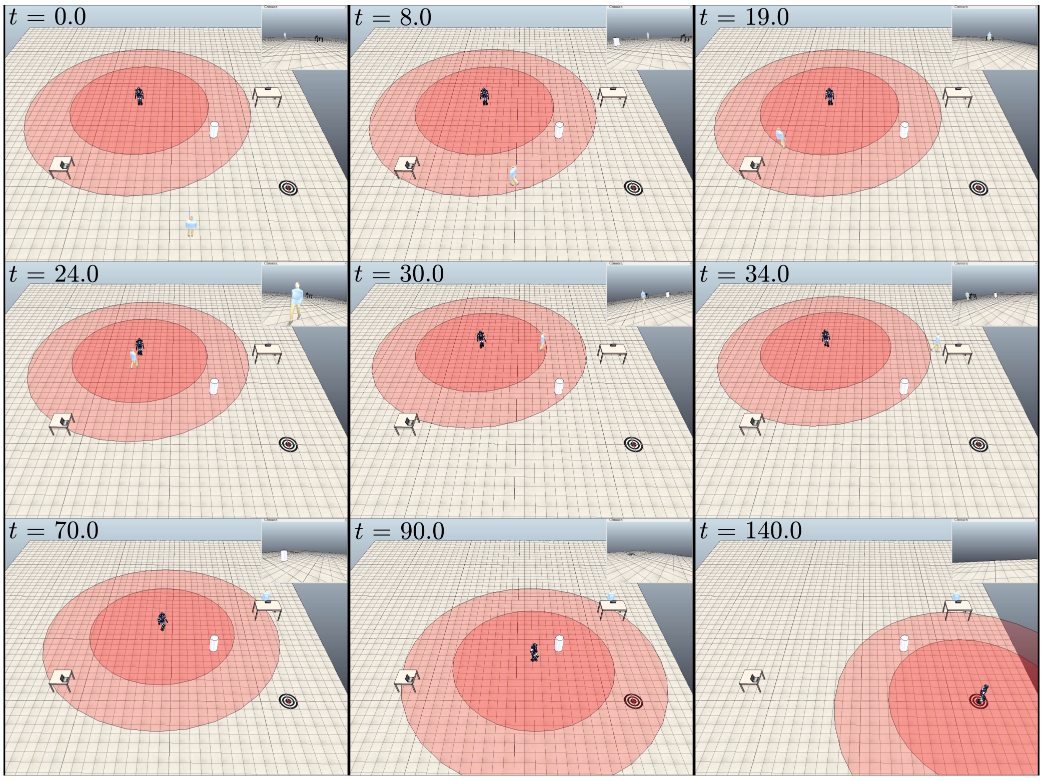


Fig. 7.15: Snapshots from a simulation of the complete framework

[3]$https://drive.google.com/file/d/1uRcEHmMCqaOzzLGjEQ2smqKl_1bk97JH/view?usp=sharing$

Chapter 8

Conclusions

Creating a society where robots and human beings can safely live together is the vision of modern robotics research. The most valid robot candidates for this shared world are humanoid robots, given their similarities with humans, and the potential to accomplish the same tasks. However, the state of the art is still far from this ideal scenario, since there are several problems yet to be solved. Humanoid robots are indeed extremely complex systems, due to their unstable nature, and the problem of balance control during locomotion is still a challenging one. Moreover, just as it was done in the past for robotic manipulators in factories, safety problems must be solved before humanoid robots are allowed to share an environment with humans. In this work we have faced the problem of human-humanoid safe coexistence starting from the problem of stable locomotion, up to the definition of a safety framework, made up of general rules, the safety guidelines, to which the robot must undergo in order not to endanger the surrounding environment.

Starting from the widely used Linear Inverted Pendulum Mode as a simplified model to plan and control humanoid gaits, we considered two different approaches for gait generation, one based on the real-time planning of the Zero-Moment Point, and the second relying on Model Predictive Control. In both approaches we have included an important theoretical result, the boundedness constraint, to guarantee that the generated trajectories do not lead to a diverging behavior. In the first approach, the boundedness constraint is used to directly connect, with a closed-form expression, the planned ZMP trajectory with the robot Center of Mass, allowing us to compute in real time all the desired quantities needed by the robot controller in the gait generation process. Subsequently the boundedness constraint has been rewritten so that it can be included in a Model Predictive Control scheme, the Instrinsically Stable MPC (IS-MPC). This control architecture generates a walking gait able to track high-level reference velocities, taking care of automatically selecting the appropriate footstep sequence and a suitable ZMP-CoM trajectory. The inclusion of the boundedness constraint let us to reduce the prediction horizon, drastically reducing the computational complexity of the MPC optimization, resulting in an algorithm suitable for real-time implementation.

After having dealt with the problem of locomotion, we developed a safety framework, giving general rules to be considered in the process of planning and control of the humanoid robot with the aim of increasing the safety of robot operations. The safety guidelines, can be summarized as follows:

- *watch what you're doing*: always watch the main area of operation

- *be on the lookout*: if idle, monitor the environment and focus on unexpected objects (e.g., moving objects)

- *evade if you can*: when a moving object approaches the robot, perform an evasive action if this can be done safely
- *stop if you must*: in a situation of clear and actual danger, stop any operation as soon as possible
- *respect humans*: scale down velocities and forces in order to reduce potential damage in the case of a collision
- *look for support*: in challenging conditions (e.g., stairs) try to establish additional contact with the environment so that the robot has at least two support points at all times

Inspired by these guidelines we have defined a set of safety behaviors that, when active, have the objective of adapting the robot current task, to guarantee safety for the robot and the environment. We defined two different sets of behaviors, namely *override* behaviors, that will actually stop (or put on hold) the current task and force the robot to take action against a dangerous situation that has been detected, and *proactive* behaviors that, on the other hand, do not stop the task, but try to increase the overall safety level by calling for an adaptation or enhancement of the current robot activity. Among these behaviors, we focused more on those that are directly related to locomotion, namely **evasion**, **footstep adaptation**, and **emergency stop**, with the clear intent of validating the gait generation techniques proposed so far.

The safety framework and the locomotion-based safety behaviors have been extensively and successfully validated through simulations in the V-REP simulation environment, and via experiments on two real robotic platforms: the NAO and the HRP-4 humanoid robots.

The proposed framework, with its guidelines and behaviors, opens the doors of shopfloors and factories to humanoid robot, to coexist and share the environment with human workers. However, coexistence implies the absence of physical interaction between humans and humanoids. Hence this framework may be further extended in future, taking into account the possibility of physical human-humanoid collaboration. This generates a new set of problems to face, starting with the problem of controlling the interaction forces, to guarantee that humans are not hurt. Moreover, the exchange of forces between human and humanoid, may lead to a loss of balance, so one must reconsider the problem of static and dynamic balancing taking into account forces intentionally exerted by external agents. Last but not least, the humanoid must be able to understand and somehow predict human actions, in order to foresee if there is the intention to start a collaboration and react accordingly.

Bibliography

[1] A. De Luca and F. Flacco, "Integrated control for PHRI: Collision avoidance, detection, reaction and collaboration," in *Biomedical Robotics and Biomechatronics (BioRob), 2012 4th IEEE RAS EMBS International Conference on*, pp. 288–295, 2012.

[2] K. Ikuta, M. Nokata, and H. Ishii, "General danger-evaluation method of human-care robot control and development of special simulator," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, vol. 4, pp. 3181–3188 vol.4, May 2001.

[3] M. Nokata, K. Ikuta, and H. Ishii, "Safety-optimizing method of human-care robot design and control," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, vol. 2, pp. 1991–1996 vol.2, May 2002.

[4] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *1985 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 500–505, 1985.

[5] B. Lacevic and P. Rocco, “Kinetostatic danger field - a novel safety assessment for human-robot interaction,” in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2169–2174, Oct 2010.

[6] B. Lacevic and P. Rocco, “Safety-oriented control of robotic manipulators ‚Äì a kinematic approach,” *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 11508 – 11513, 2011. 18th IFAC World Congress.

[7] B. Lacevic, P. Rocco, and A. Zanchettin, “Safety assessment and control of robotic manipulators using danger field,” *Robotics, IEEE Transactions on*, vol. 29, no. 5, pp. 1257–1270, 2013.

[8] F. Flacco, T. Kroeger, A. De Luca, and O. Khatib, “A depth space approach to human-robot collision avoidance,” in *2012IEEE Int. Conf. on Robotics and Automation*, pp. 338–345, 2012.

[9] F. Flacco, T. Kroeger, A. De Luca, and O. Khatib, “A depth space approach for evaluating distance to objects,” vol. 80, no. 1, pp. 7–22, 2014.

[10] M. Cefalo and G. Oriolo, “Dynamically feasible task-constrained motion planning with moving obstacles,” in *2014 IEEE Int. Conf. on Robotics and Automation*, 2014.

[11] J. Vannoy and J. Xiao, “Real-Time Adaptive Motion Planning (RAMP) of Mobile Manipulators in Dynamic Environments with Unforeseen Changes,” *IEEE Trans. on Robotics*, vol. 24, no. 5, pp. 1199–1212, 2008.

[12] M. Otte and E. Frazzoli, "RRT-X: Asymptotically optimal single-query sampling-based motion planning with quick replanning," *Int. J. of Robotics Research*, 2015.

[13] D. Kulic and E. A. Croft, "Safe planning for human-robot interaction," *Journal of Robotic Systems*, vol. 22, no. 7, pp. 383–396, 2005.

[14] D. Kulic and E. A. Croft, "Real-time safety for human-robot interaction," vol. 54, no. 1, pp. 1–12, 2006.

[15] D. Kulic and E. A. Croft, "Affective state estimation for human-robot interaction," *IEEE Trans. on Robotics*, vol. 23, no. 5, pp. 991–1000, 2007.

[16] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch, "Human-aware robot navigation: A survey," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1726 – 1743, 2013.

[17] E. Yoshida and F. Kanehiro, "Reactive robot motion using path replanning and deformation," in *2011 IEEE International Conference on Robotics and Automation*, pp. 5456–5462, 2011.

[18] S. M. Khansari-Zadeh and A. Billard, "A dynamical system approach to realtime obstacle avoidance," *Autonomous Robots*, vol. 32, no. 4, pp. 433–454, 2012.

[19] P. Michel, J. Chestnutt, J. Kuffner, and T. Kanade, "Vision-guided humanoid footstep planning for dynamic environments," in *2005 IEEE-RAS Int. Conf. on Humanoid Robots*, pp. 13–18, 2005.

[20] J. Chestnutt, "Navigation and gait planning," in *Motion Planning for Humanoid Robots* (K. Harada, E. Yoshida, and K. Yokoi, eds.), pp. 1–28, Springer, 2010.

[21] N. Bohorquez, A. Sherikov, D. Dimitrov, and P. B. Wieber, "Safe navigation strategies for a biped robot walking in a crowd," in *16th IEEE-RAS Int. Conf. on Humanoid Robots*, pp. 379–386, 2016.

[22] 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, 2017.

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

[24] 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*, pp. 1620–1626, 2003.

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

[26] 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.

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

[28] A. Ibanez, P. Bidaud, and V. Padois, "Emergence of humanoid walking behaviors from mixed-integer model predictive control," in *2014 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 4014–4021, 2014.

[29] 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*, pp. 1943–1950, 2014.

[30] J. Pratt, J. Carff, S. Drakunov, and A. Goswami, "Capture point: A step toward humanoid push recovery," in *6th IEEE-RAS Int. Conf. on Humanoid Robots*, pp. 200–207, 2006.

[31] 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," *Int. J. of Robotics Research*, vol. 31, no. 9, pp. 1094–1113, 2012.

[32] 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.

[33] M. Krause, J. Englsberger, P.-B. Wieber, and C. Ott, "Stabilization of the capture point dynamics for bipedal walking based on model pre-

dictive control," in *10th IFAC Symp. on Robot Control*, pp. 165–171, 2012.

[34] 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*, pp. 1763–1768, 2016.

[35] P. B. Wieber, "Viability and predictive control for safe locomotion," in *2008 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 1103–1108, 2008.

[36] 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*, pp. 951–958, 2014.

[37] L. Lanari and S. Hutchinson, "Inversion-based gait generation for humanoid robots," in *2015 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2015.

[38] 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 *IROS*, 2001.

[39] J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue, "Online footstep planning for humanoid robots," in *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, vol. 1, pp. 932–937 vol.1, Sept 2003.

[40] P. Karkowski, S. Oswald, and M. Bennewitz, "Real-time footstep planning in 3d environments," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 69–74, Nov 2016.

[41] D. Kanoulas, A. Stumpf, V. S. Raghavan, C. Zhou, A. Toumpa, O. V. Stryk, D. G. Caldwell, and N. G. Tsagarakis, "Footstep planning in rough terrain for bipedal robots using curved contact patches," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–9, May 2018.

[42] N. Scianca, M. Cognetti, D. D. Simone, L. Lanari, and G. Oriolo, "Intrinsically stable mpc for humanoid gait generation," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 601–606, Nov 2016.

[43] A. Aboudonia, N. Scianca, D. D. Simone, L. Lanari, and G. Oriolo, "Humanoid gait generation for walk-to locomotion using single-stage mpc," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pp. 178–183, Nov 2017.

[44] V. Samy and A. Kheddar, "Falls control using posture reshaping and active compliance," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 908–913, Nov 2015.

[45] M. Cognetti, D. D. Simone, L. Lanari, and G. Oriolo, "Real-time planning and execution of evasive motions for a humanoid robot," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4200–4206, May 2016.

[46] C. K. Hsee, Y. Tu, Z. Y. Lu, and B. Ruan, “Approach aversion: Negative hedonic reactions toward approaching stimuli,” *Journal of Personality and Social Psychology*, vol. 106, no. 5, pp. 699–712, 2014.

[47] K. Mombaur, A. Truong, and J.-P. Laumond, “From human to humanoid locomotion – an inverse optimal control approach,” *Autonomous Robots*, vol. 28, pp. 369–383, 2010.

[48] T.-V.-A. Truong, D. Flavigne, J. Pettre, K. Mombaur, and J.-P. Laumond, “Reactive synthesizing of human locomotion combining nonholonomic and holonomic behaviors,” in *3rd IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics*, pp. 632–637, 2010.

[49] A. Faragasso, G. Oriolo, A. Paolillo, and M. Vendittelli, “Vision-based corridor navigation for humanoid robots,” in *2013 IEEE International Conference on Robotics and Automation*, pp. 3190–3195, 2013.

[50] M. Cognetti, D. D. Simone, F. Patota, N. Scianca, L. Lanari, and G. Oriolo, “Real-time pursuit-evasion with humanoid robots,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4090–4095, May 2017.

[51] D. D. Simone, N. Scianca, P. Ferrari, L. Lanari, and G. Oriolo, “Mpc-based humanoid pursuit-evasion in the presence of obstacles,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5245–5250, Sept 2017.

[52] K. Fujiwara, F. Kanehiro, S. Kajita, K. Yokoi, H. Saito, K. Harada, K. Kaneko, and H. Hirukawa, "The first human-size humanoid that can fall over safely and stand-up again," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 2, pp. 1920–1926 vol.2, Oct 2003.

[53] K. Fujiwara, F. Kanehiro, S. Kajita, and H. Hirukawa, "Safe knee landing of a human-size humanoid robot while falling forward," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 1, pp. 503–508 vol.1, Sept 2004.

[54] A. Yasin, Q. Huang, Z. Yu, Q. Xu, and A. A. Syed, "Stepping to recover: A 3d-lipm based push recovery and fall management scheme for biped robots," in *2012 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 318–323, Dec 2012.

[55] S. Yun, A. Goswami, and Y. Sakagami, "Safe fall: Humanoid robot fall direction change through intelligent stepping and inertia shaping," in *2009 IEEE International Conference on Robotics and Automation*, pp. 781–787, May 2009.

[56] U. Nagarajan and A. Goswami, "Generalized direction changing fall control of humanoid robots among multiple objects," in *2010 IEEE International Conference on Robotics and Automation*, pp. 3316–3322, May 2010.

[57] S. Yun and A. Goswami, “Hardware experiments of humanoid robot safe fall using aldebaran nao,” in *2012 IEEE International Conference on Robotics and Automation*, pp. 71–78, May 2012.

[58] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control.* Springer Publishing Company, Incorporated, 1st ed., 2008.

[59] D. M. Kim, H. J. Choi, J. S. Kim, W. K. Lee, D. H. Song, and S. Jung, “Tracking control of a moving object for robokers with stereo visual feedback,” in *2007 IEEE International Conference on Integration Technology*, pp. 52–57, March 2007.

[60] A. A. Shafie, A. Iqbal, and M. R. Khan, “Visual tracking and servoing of human face for robotic head amir-ii,” in *International Conference on Computer and Communication Engineering (ICCCE'10)*, pp. 1–4, May 2010.

[61] N. Marturi, V. Ortenzi, J. Xiao, M. Adjigble, R. Stolkin, and A. Leonardis, “A real-time tracking and optimised gaze control for a redundant humanoid robot head,” in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 467–474, Nov 2015.

[62] M. Wan, H. Zhang, M. Fu, and W. Zhou, “Motion control strategy for redundant visual tracking mechanism of a humanoid robot,” in *2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, vol. 02, pp. 156–159, Aug 2016.

[63] D. T. Nguyen, W. Li, and P. O. Ogunbona, “Human detection from images and videos: A survey,” *Pattern Recognition*, vol. 51, pp. 148 – 175, 2016.

[64] L. Sentis, J. Park, and O. Khatib, “Compliant control of multicontact and center-of-mass behaviors in humanoid robots,” *IEEE Transactions on Robotics*, vol. 26, pp. 483–501, June 2010.

[65] K. Harada, S. Kajita, K. Kaneko, and H. Hirukawa, “Zmp analysis for arm/leg coordination,” in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 1, pp. 75–81 vol.1, Oct 2003.

[66] 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 International Conference on Intelligent Robots and Systems*, pp. 3253–3258, Sept 2014.

[67] B. Henze, M. A. Roa, and C. Ott, “Passivity-based whole-body balancing for torque-controlled humanoid robots in multi-contact scenarios,” *The International Journal of Robotics Research*, vol. 35, no. 12, pp. 1522–1543, 2016.

[68] C. Mandery, J. B. Sol, M. Jöchner, and T. Asfour, “Analyzing whole-body pose transitions in multi-contact motions,” *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 1020–1027, 2015.

[69] G. Mesesan, J. Englsberger, B. Henze, and C. Ott, “Dynamic multi-contact transitions for humanoid robots using divergent component of motion,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4108–4115, May 2017.

[70] A. Werner, B. Henze, D. A. Rodriguez, J. Gabaret, O. Porges, and M. A. Roa, “Multi-contact planning and control for a torque-controlled humanoid robot,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5708–5715, Oct 2016.

[71] S. Lengagne, J. Vaillant, E. Yoshida, and A. Kheddar, “Generation of whole-body optimal dynamic multi-contact motions,” *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1104–1119, 2013.

[72] H. Audren, J. Vaillant, A. Kheddar, A. Escande, K. Kaneko, and E. Yoshida, “Model preview control in multi-contact motion-application to a humanoid robot,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4030–4035, Sept 2014.