



## Research Article

# A modular framework to generate robust biped locomotion: from planning to control

Mohammadreza Kasaei<sup>1</sup>  · Ali Ahmadi<sup>2</sup> · Nuno Lau<sup>1</sup> · Artur Pereira<sup>1</sup>

Received: 24 July 2020 / Accepted: 20 July 2021

Published online: 11 August 2021

© The Author(s) 2021 [OPEN](#)

## Abstract

Biped robots are inherently unstable because of their complex kinematics as well as dynamics. Despite many research efforts in developing biped locomotion, the performance of biped locomotion is still far from the expectations. This paper proposes a model-based framework to generate stable biped locomotion. The core of this framework is an abstract dynamics model which is composed of three masses to consider the dynamics of stance leg, torso, and swing leg for minimizing the tracking problems. According to this dynamics model, we propose a modular walking reference trajectories planner which takes into account obstacles to plan all the references. Moreover, this dynamics model is used to formulate the controller as a Model Predictive Control (MPC) scheme which can consider some constraints in the states of the system, inputs, outputs, and also mixed input-output. The performance and the robustness of the proposed framework are validated by performing several numerical simulations using MATLAB. Moreover, the framework is deployed on a simulated torque-controlled humanoid to verify its performance and robustness. The simulation results show that the proposed framework is capable of generating biped locomotion robustly.

**Keywords** Robust biped locomotion · Model Predictive Control (MPC) · Dynamics model · Humanoid robots

## 1 Introduction

Humanoid robots are more adapted to our real environment for helping us to perform our daily-life tasks. Developing a robust walking framework for humanoid robots has been researched for decades, and it is still a challenging problem in the robotics community. The complexity of this problem derives from several different aspects like considering an accurate hybrid dynamics model, designing appropriate controllers, and formulating suitable reference trajectory planners. The wide range of applications for these types of robots motivates researchers to tackle such a complex problem using different approaches which

can be generally classified into two main classes: *model-free* [24, 30] and *model-based* [4, 7, 16].

Model-free approaches try to generate walking patterns by generating rhythmic patterns for each limb and without considering any dynamics model of the system. Some of them are inspired from neuro-physiological studies on humans and animals. These studies showed that periodic locomotion such as walking and running are generated by a set of oscillators at spinal cord which are connected together in a specific arrangement. Each oscillator has a set of parameters that are generally tuned by some trial-intensive method like trials and error, machine learning (ML) algorithms or both of them [25, 33]. Some other approaches in this class are designed based on learning

✉ Mohammadreza Kasaei, Mohammadreza@ua.pt; Ali Ahmadi, ali.ahmadi@isr.uc.pt; Nuno Lau, nunolau@ua.pt; Artur Pereira, artur@ua.pt | <sup>1</sup>IEETA / DETI, University of Aveiro, 3810-193 Aveiro, Portugal. <sup>2</sup>Institute of Systems and Robotics (ISR), University of Coimbra, 3030-290 Coimbra, Portugal.



from scratch and mostly are based on Reinforcement Learning (RL) algorithms [1, 19] which need many samples to be able to generate walking that takes a considerable amount of time. Unlike the model-free approaches, the fundamental core of the model-based approach is a dynamics model of the robot. The main idea of these approaches is modeling the overall behavior of the system in an abstract manner, then planners and controllers will be formulated according to this model. The first question in designing a model-based walking is *how accurate should the model be?*. To answer this question, two points of view exist: (1) using a whole-body dynamics model, (2) using an abstract model. We believe that for designing a dynamics model of a system, a trade-off between accuracy and simplicity should be taken into account. Although the whole body dynamics model is more accurate than an abstract model, it is not only platform-dependent but also needs a high computational effort. The main idea behind using an abstract model is fading the complexity of the control system and organizing the control system as a hierarchy. In hierarchical control approaches, a simplified dynamics model is used to abstract the overall behaviors of the system, and then these behaviors will be converted to individual actuator inputs using a detailed full-body inverse dynamics [7]. In such approaches, the overall performance of the system depends on the matching between the abstract model and the exact model.

In this paper, we proposed a modular walking framework capable of generating robust locomotion for humanoid robots. This work follows our previous work on model-based biped locomotion framework [14]. The present work aims to offer more details about this framework. In particular, we present the overall architecture of the proposed framework and describe each module in detail. Additionally, we deploy the framework on a full-size simulated torque-controlled humanoid robot and conduct a set of simulations to validate its robustness, performance and portability. In this framework, the overall dynamics of the robot is modeled using a three-mass model which takes into account dynamics of the legs and the torso. According to this model, the problem of dynamics locomotion will be formulated as a linear MPC to predict the behavior of the system over a prediction horizon and to determine the optimum control inputs. Additionally, the process of generating walking reference trajectories will be decomposed into three levels including (1) path and footstep planning, (2) planning ZMP, hip and swing reference trajectories and (3) planning a set of reference trajectories for the controller. Moreover, the proposed framework has a hierarchical structure which is composed of several reusable blocks. The designed architecture reduces the complexity of the framework and can be used to advancements in research and development. Besides, the results of several

simulations will be presented to show the performance and the robustness of the proposed walking scheme.

Our contribution is twofold. First, the development of a modular framework that reduces the complexity and increases the flexibility of generating robust locomotion. Second, formulating the walking controller as a Linear MPC based on three-mass model. The remainder of this paper is structured as follows: Sect. 2 gives an overview of the related work. The overall architecture of the framework will be presented in Sect. 3. Later, in Sect. 4, the reference trajectory planners will be presented. Afterwards, the three-mass dynamics model will be reviewed briefly in Sect. 5 and then we will explain how this model will be used to design a walking controller based on MPC scheme in Sect. 6. In Sect. 7, a set of simulation scenarios will be designed and performed to examine the tracking performance and robustness of the proposed controller. Afterwards, in Sect. 8, a baseline framework based on LIPM will be used to compare and highlight the effectiveness of the proposed framework. Finally, conclusions and future research will be presented in Sect. 9.

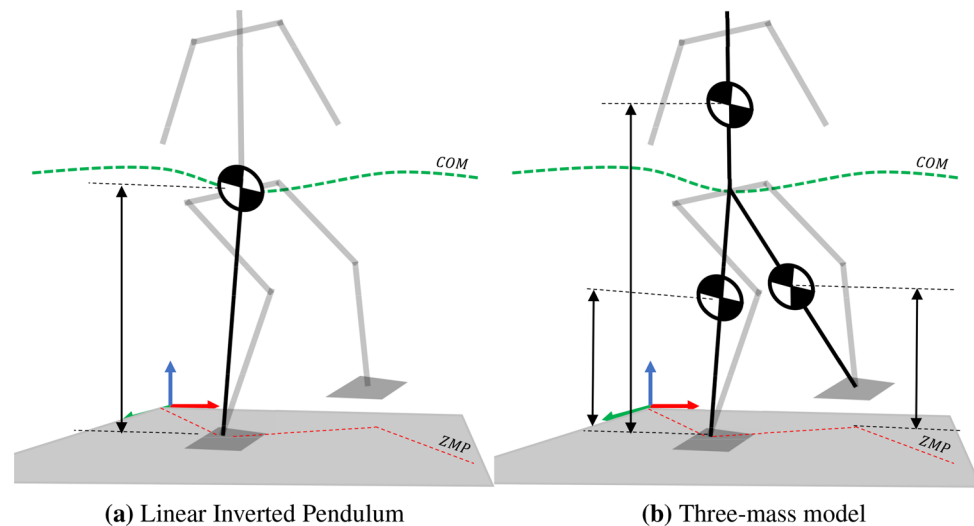
## 2 Related work

Several simplified dynamics models have been proposed that abstract the overall dynamics of humanoid robots. In this section, some of these models will be reviewed briefly and then some recent works that used these models to generate robust locomotion or to develop push recovery strategies will be presented.

### 2.1 Dynamics models

Linear Inverted Pendulum Model (LIPM) is one of the common dynamics models in the literature. The popularity of this model originates from its linearity and simplicity and from its ability to generate a feasible, fast, and efficient trajectory of the Center of Mass (COM) [13]. This model describes the dynamics of a humanoid robot just by considering a single mass that is connected to the ground via a mass-less rod (see Fig. 1a). In this model, the single mass is assumed to move along a horizontal plane and based on this assumption, the motion equations in sagittal and frontal planes are decoupled and independent. Several studies used this model to develop an online walking generator based on optimal control [17] and also linear MPC [3, 10, 22]. Several extensions of LIPM have been proposed to increase the accuracy of this model while keeping the simplicity level [7, 18, 21, 27]. The three-mass model is one of the extended versions of LIPM [26, 27, 29]. As it is shown in Fig. 1b, this model considers the masses of legs and the torso to increase the accuracy of the model. It should be

**Fig. 1** Abstract dynamics models: **a** represents the Linear Inverted Pendulum Model (LIPM) which abstracts the overall dynamics of a humanoid robot into a single mass with restricted vertical motion; **b** represents the three-mass model which takes into account the dynamics of the legs and the torso with restricted vertical motions



mentioned that to keep the model linear, the vertical motions of the masses are considered to be smooth and the vertical accelerations are neglected.

Kajita et al. [13] introduced the Three Dimensional Linear Inverted Pendulum Model (3D-LIPM) and explained how this model can be used to generate walking. Afterwards, in [12], they used the concept of ZMP to develop a control framework based on preview control to generate stable locomotion. The performance of their framework has been validated using several simulations.

Albert and Gerth [2] extended LIPM by considering the mass of the swing leg to improve gait stability and they named it Two Mass Inverted Pendulum Model (TMIPM). In their approach, the swing leg and the ZMP trajectories should be generated firstly, then the trajectory of the COM will be generated by solving a linear differential equation. They proposed an extended version of TMIPM which considers the mass of the thigh, the shank, and the foot of the swinging leg which has been called Multiple Masses Inverted Pendulum Model (MMIPM). TMIPM and MMIPM are more accurate than LIPM, they need an iterative algorithm to define the COM trajectory due to the dependency of masses' motions on each other.

Shimmyo et al. [27] extended the LIPM by considering three masses to decrease the modeling error and improve the performance. In their model, the masses were located on the torso, the right leg, and the left leg. They made two assumptions to design a preview controller which were considering the constant vertical position for the masses and also constant mass distribution. The performance of their approach has been validated through several experiments.

Faraji and Ijspeert [7] proposed a dynamics model which is composed of three linear pendulums to model the dynamics of the legs and the torso and they named it

3LP. They argued that, due to the linearity of this model, it is fast and computationally similar to LIPM such that it can be used in modern control architectures from the computational perspective. The performance of their model has been proven using a set of simulations. Simulation results showed that their framework was able to generate a robust walking with a wide range of speeds without requiring off-line optimization and tuning of parameters.

Several extended versions of LIPM have been proposed to consider the momentum around the COM [23] but they do not take into account the dynamics of the legs specifically. In these models, the legs are considered to be massless and their motions do not have any effect in dynamics. In our previous work [15], we extended the dynamics model presented in [23] by considering the mass of the stance leg, and we explained how this model can be represented by a differential equation which can be used to design a controller to plan and track the walking reference trajectories.

## 2.2 Locomotion framework

Herd et al. [9] proposed an online walking generator with automatic footstep placement. They used LIPM to formulate the walking pattern generator as a Linear MPC. In their framework, instead of considering a set of predefined footsteps, they introduce new control variables to generate footsteps automatically. The performance of their approach has been validated through a set of simulations. The simulation results showed that the framework is able to track a given reference speed of COM that can be modified any time.

Brasseur et al. [3] designed a robust linear MPC to generate online 3D locomotion for humanoid robots in a single computation with guaranteed kinematic and dynamic

feasibility. They used Newton and Euler equations of motion to model the dynamics of a humanoid robot. They considered some assumptions to reduce the complexity of the model which are almost the same as LIPM's assumptions, but unlike LIPM, they take into account the vertical motion of the COM to generate more efficient locomotion in terms of energy and speed. The performance of their approach has been tested in two simulation scenarios including walking on a flat train and climbing stairs using a simulated HRP-2 robot. The simulation results validated the performance of their approach.

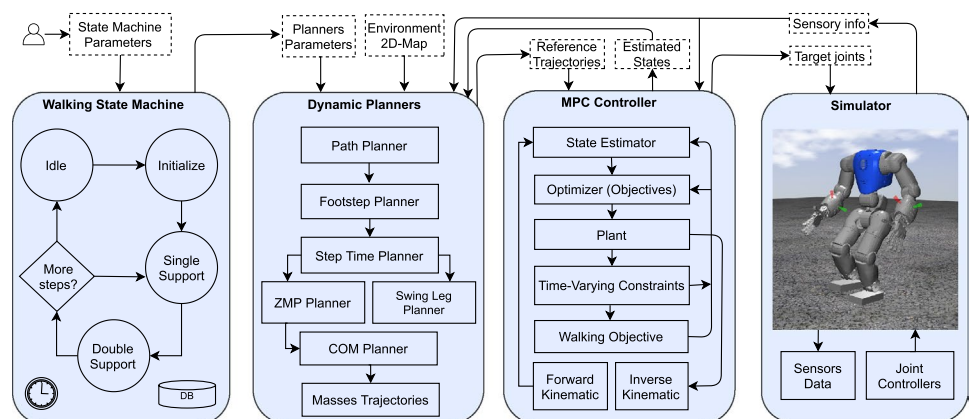
Jianwen Luo et al. [20] proposed a hierarchical control structure to generate robust biped locomotion. Their structure is composed of three independent control levels. At the highest level, they used  $A^*$  to plan a path to the destination, and then they designed a MPC based on LIPM to generate the COM trajectory. At the middle level, they designed a hybrid controller which is composed of two controllers to control the oscillation and to eliminate the chattering problem. At the lowest level, they used a whole body operational space (WBOS) controller to compute the joint torques analytically. The effectiveness of their approach has been validated through simulation.

Most of the aforementioned works use LIPM as their dynamics model and do not take into account the dynamics of the swing leg and the torso. In this paper, we propose a modular framework to generate a robust biped locomotion which takes into account the dynamics of torso and swing leg. Particularly, we use the three-mass model as the core of this framework and based on that, we design a MPC controller to formulate walking as a set of quadratic functions and a set of time-varying constraints based on states, inputs, and outputs of the system. This controller is not only able to track the reference trajectories optimally even in presence of measurement noise but also it is robust against uncertainties such as external disturbances. Furthermore, we will performed a set of simulations to explore the impacts of the proposed controller and to validate the performance of the framework.

### 3 Architecture

This section is focused on presenting the overall architecture of the proposed framework which is depicted in Fig. 2. This framework is composed of four main modules which are coupled together to generate robust biped locomotion. The first module is *Walking State Machine* which is responsible for controlling the overall process of the walking. According to the periodic nature of the walking, this state machine is composed of four main states such that state transitions are triggered based on an associated timer and also state conditions. In the *Idle state*, the robot stands and is waiting for a start walking command. Once a walking command is received, the state transits to the *Initialize state* and the robot shifts its COM to the first support foot to be ready for taking the first step. In the *Single Support State* and the *Double Support State*, the planners parameters (e.g., robot's target position and orientation, max speed, etc.) will be updated. These parameters along with a 2D map of the environment and the sensory information provided by the *Simulator* as well as the estimated states provided by the *MPC Controller* will be fed into the *Dynamic Planner* to generate a set of reference trajectories. *Dynamic Planner* starts by generating a collision-free path which will be used to plan a set of reference trajectories including footsteps, ZMP, swing, COM, and masses trajectories. Finally, these references will be fed into the *MPC Controller*. These trajectories along with the sensory information (e.g., joint encoders, IMU, robot's position and orientation, etc.) provided by the *Simulator* will be used to estimate the robot's states. The estimated states and a set of objectives are used by the optimizer to specify the control inputs subject to a set of time-varying constraints for tracking the generated reference trajectories. The corresponding joint motions will be generated using the *Inverse Kinematics* which takes into account kinematic

**Fig. 2** Overall architecture of the proposed framework. This framework is composed of four main modules which are connected together to generate robust locomotion. The highlighted boxes represent the functional blocks and the exchanged information among the modules are represented by the white boxes





feasibility constraints. The target joint setpoints will be fed into the `Simulator` which is responsible for simulating the interaction of the robot with the environment and producing sensory information as well as the global position and orientation of the robot in the simulated environment to be used by `Dynamic Planner`. In the following sections, the details of each module will be explained separately, and then a set of simulation scenarios will be designed and conducted to valid the performance of the framework.

## 4 Dynamic planners

Dynamic Planners module is responsible for planning of the reference trajectories according to the input parameters as well as the environment's structure. To reduce the complexity of the planning, the planning process is divided into a set of sub-planners which are solved separately and connected in a hierarchical manner (see Fig. 2). In the rest of this section, we will explain how the reference trajectories will be generated.

### 4.1 A\* footstep planning

Footstep planning is generally based on a graph search algorithm with a rich history [8, 11]. In our target framework, the footstep planning is composed of two stages: (1) generating a collision-free 2D body path; (2) planning the footsteps based on the generated path. In the first stage, the environment is modeled as a 2D grid map consisting of cells that are marked as free or occupied. In this work, the size of the cell is considered to be  $0.1 \text{ m}^2$ , the height of obstacles is not considered, and the robot can not step over them. In order to avoid collision with the obstacles, the size of the obstacles are considered larger than the real size (scale = 1.1). In this stage, after modeling the environment, the A\* search method is applied to find an optimum path over the free cells from the current position of the robot to the goal. Euclidean distance to the goal is used to guide the search. In the second stage, the footsteps should be generated according to the generated path. To do so, a state variable is defined to describe the current state of the robot's feet:

$$s = (x_l, y_l, \theta_l, \phi_l, x_r, y_r, \theta_r, \phi_r) \quad (1)$$

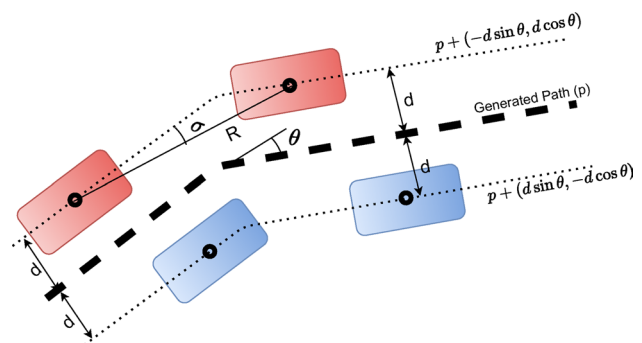
where  $x_l, y_l, \theta_l, x_r, y_r, \theta_r$  represent the current position and orientation of the left and the right feet, respectively.  $\phi_l, \phi_r$  represent the state of each foot which is 1 if the foot is the swing foot, and  $-1$  otherwise. According to the nature of walking which is generated by moving the right

and left legs, alternating, a step action is parameterized by a distance and an angle from the swing foot position at the beginning of step  $a = (R, \sigma)$ . Based on the current state and the generated path, action should be taken. In this paper, we consider a constant step size ( $R = 0.1 \text{ m}$ ) which is a kind of medium step size that has been determined based on the robot capability and  $\sigma$  is determined according to the generated path and the current state of the feet. Figure 3 shows an example of planning footsteps after generating the collision-free path.

It is worth mentioning that in some scenarios like tele-operation tasks, an operator wants to determine the actions without any autonomous path planning. In such scenario, the step positions will be planned just according to the input command. It should be mentioned that the action is always passed through a first-order lag filter to ensure a smooth update. By applying the selected action, the state transits to a new state,  $s' = t(s, a)$ . It should be noted that after each transition, the current footstep will be saved ( $f_i, i \in \mathbb{N}$ ), also  $\phi_l$  and  $\phi_r$  will be toggled. After generating the footsteps, the step time should be planned based on the robot's target speed.

### 4.2 ZMP, hip and swing reference trajectories

Studies on human locomotion showed that while a human is walking, ZMP moves from heel to the toe during the single support phase and it moves towards the COM during the double support phase [5, 6]. In this paper, we do not consider the ZMP movement during the single support, but instead, we keep it in the middle of the support foot to prevent it from approaching to the support polygon edge and to avoid the loss of



**Fig. 3** Example of planning footsteps. After generating a collision-free path (thick dashed black line), two lines parallel to the path will be generated (dotted black lines) which are  $d$  m away from the generated path where  $2d$  is the distance between feet. Then, the left and right footsteps (red and blue rectangles) should be selected from these path so that  $R$  m away from its previous one. Afterwards,  $\sigma$  will be determined based on the generated footsteps using inverse tangent function

equilibrium of the robot in some situation like unexpected external disturbances. Thus, the ZMP reference trajectory planning can be formulated based on the generated footsteps as follows:

$$p_{st} = \begin{cases} f_i & 0 \leq t < T_{ss} \\ f_i + \frac{SL \times (t - T_{ss})}{T_{ds}} & T_{ss} \leq t < T_{ss} + T_{ds} \end{cases} \quad (2)$$

where  $p_{st} = [p_{st}^x \ p_{st}^y]^T$  is the generated ZMP,  $t, T_{ss}, T_{ds}$  represent the time, duration of single and double support phases, respectively.  $SL = [SL^x \ SL^y]$  is a vector that represents the step length and step width which are determined based on  $(R, \sigma), f_i = [f_i^x \ f_i^y]$  represents the planned foot steps on a 2D surface. It should be noted that  $t$  will be reset at the end of each step ( $t \geq T_{ss} + T_{ds}$ ).

After generating the ZMP reference trajectory, the hip trajectory will be planned according to the generated ZMP. To do that, we assume that the COM of the robot is located at the middle of the hip, and, based on this assumption, the overall dynamics of the robot is firstly restricted into COM and then the reference trajectory for the hip will be generated using the analytical solution of the LIPM as follows:

$$p_h(t) = p_{st} + \frac{(p_{st} - p_{h_r}) \sinh((t - t_0)\omega) + (p_{h_0} - p_{st}) \sinh((t - t_f)\omega)}{\sinh((t_0 - t_f)\omega)}, \quad (3)$$

where  $t_0$  and  $t_f$  represent the beginning and the ending times of a step,  $\omega = \sqrt{\frac{g}{l_h}}$  is the natural frequency of LIPM and  $p_{h_0}, p_{h_r}$  are the corresponding positions of COM at these times, respectively.

After generating the ZMP and the hip trajectory, the swing trajectory should be planned. To have a smooth trajectory during lifting and landing of the swing leg, a Bézier curve is used to generate this trajectory according

to the generated footsteps and a predefined swing height.

### 4.3 Masses reference trajectories

The trajectories of the masses can be easily generated based on the geometric relations between the generated hip, ZMP, and swing leg trajectories. According to Fig. 1b, the mass of the stance leg is located in the middle of the line between the ZMP and the hip. Similarly, the mass of the swing leg is located in the middle of the line between the swing foot and the hip. To show the performance of the planners presented in this section, an example path planning scenario has been set up, which is depicted in Fig. 4. In this scenario, the robot stands at the *START* point and wants to reach the *GOAL* point. The planning process is started by generating an optimum obstacle-free path (black-dashed line). As it is shown in Fig. 4, the generated path is far from the obstacles enough and, based on this path, the footsteps, corresponding ZMP and hip trajectories have been generated successfully.

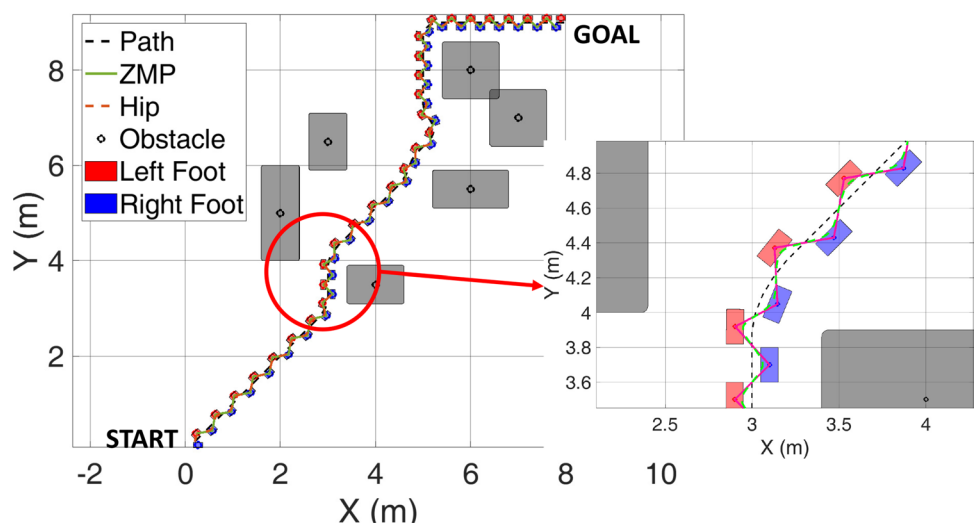
## 5 Dynamics model

This section will be started by a brief review of the Zero Momentum Point (ZMP) concept which is one of the well-known criteria in developing stable dynamic walking. Afterwards, the ZMP concept will be used to define the overall dynamics of a biped robot as a state-space system.

### 5.1 Zero momentum point and gait stability

Several criteria have been introduced to analyze the stability of a biped robot. ZMP is a well-known criterion to

**Fig. 4** An example walking references trajectories: the gray rectangles represent the occupied cells that robot can not step; the black-dashed line represents the output of the A\* path planner; red and blue rectangles represent the footsteps which are generated based on the output of the A\*; the magenta line is the ZMP which is generated based on the outputs of the footstep planner; the lime-dashed line is the hip reference trajectory



develop dynamically stable locomotion and its popularity comes from its performance and ease of use. In fact, ZMP is the point on the ground plane where the ground reaction force (GRF) acts to cancel the inertia and gravity [32]. For a dynamics model which is composed of  $n$  parts, ZMP can be calculated using the following equation:

$$p = \frac{\sum_{k=1}^n m_k c_k (\ddot{z}_k + g) - \sum_{k=1}^n m_k z_k \ddot{c}_k}{\sum_{k=1}^n m_k (\ddot{z}_k + g)}, \quad (4)$$

where  $g$  represents the acceleration of gravity,  $p = [p^x \ p^y]^T$  represents the position of ZMP,  $m_k$  is the mass of each part,  $c_k = [c_k^x \ c_k^y]^T$  and  $\ddot{c}_k$  denote the ground projection of the position and acceleration of each mass,  $z_k, \ddot{z}_k$  are vertical position and vertical acceleration of each mass, respectively.

### 5.2 Three-mass dynamics model

The three-mass model abstracts the dynamics model of a biped robot by considering three masses. As shown in Fig. 1b, the masses are placed at the legs and the torso of the robot. To simplify and linearize the model, each mass is restricted to move along a horizontal plane. According to this assumption, the ZMP equations are independent and equivalent in the frontal and sagittal planes. Thus, in the remainder of this paper, just the equations in the sagittal plane will be considered. Based on (4), a state-space system can be defined to analyze the behavior of the system:

$$\frac{d}{dt} \underbrace{\begin{bmatrix} c_1^x \\ \dot{c}_1^x \\ \ddot{c}_1^x \\ c_2^x \\ \dot{c}_2^x \\ \ddot{c}_2^x \\ c_3^x \\ \dot{c}_3^x \\ \ddot{c}_3^x \end{bmatrix}}_X = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_A \underbrace{\begin{bmatrix} c_1^x \\ \dot{c}_1^x \\ \ddot{c}_1^x \\ c_2^x \\ \dot{c}_2^x \\ \ddot{c}_2^x \\ c_3^x \\ \dot{c}_3^x \\ \ddot{c}_3^x \end{bmatrix}}_X + \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_B \underbrace{\begin{bmatrix} \ddot{c}_1^x \\ \ddot{c}_2^x \\ \ddot{c}_3^x \end{bmatrix}}_U \quad (5)$$

$$y = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ \frac{m_1}{M} & 0 & \frac{-m_1 z_1}{Mg} & \frac{m_2}{M} & 0 & \frac{-m_2 z_2}{Mg} & \frac{m_3}{M} & 0 & \frac{-m_3 z_3}{Mg} \end{bmatrix}}_C \underbrace{\begin{bmatrix} c_1^x \\ \dot{c}_1^x \\ \ddot{c}_1^x \\ c_2^x \\ \dot{c}_2^x \\ \ddot{c}_2^x \\ c_3^x \\ \dot{c}_3^x \\ \ddot{c}_3^x \end{bmatrix}}_X$$

where  $\ddot{c}_1^x, \ddot{c}_2^x, \ddot{c}_3^x$  are the manipulated variables in jerk dimension,  $M$  is weight of the robot,  $m_1, m_2, m_3$  represent the mass of stance leg, torso and swing leg, respectively. Based on the output equation (y), the positions of stance leg, swing leg, and also ZMP are measured at each control cycle. In the next section, the defined system will be discretized to discrete-time implementation and we will explain what type of walking objectives and constraints should be considered to generate stable dynamic walking.

## 6 Online walking controller based on MPC

In this section, the problem of the online walking controller is formulated as a linear MPC which is not only robust to uncertainties but also able to consider some constraints in the states, inputs and outputs. To do that, firstly, the presented continuous system (5) should be discretized for implementation in discrete time. Afterward, the walking objective will be formulated as a set of quadratic functions and finally, walking constraints will be formulated as linear functions of the states, inputs, and outputs.

### 6.1 Discrete dynamics model

To discretize the system, we assume that  $\ddot{c}_1, \ddot{c}_2, \ddot{c}_3$  are linear and, based on this assumption,  $\ddot{c}_1, \ddot{c}_2, \ddot{c}_3$  are constant

within a control cycle. Thus, the discretized system can be represented as follows:

$$\begin{aligned} X(k+1) &= A_d X(k) + B_d u(k) \\ y(k) &= C_d X(k) \end{aligned} \tag{6}$$

where  $k$  represents the current sampling instance,  $A_d, B_d, C_d$  are the discretized version of the  $A, B, C$  matrices in (5), respectively. Based on this system, the state vector  $X(k)$  can be estimated at each control cycle. Thus, according to the estimated states, some objectives, and constraints, the problem of determining the control inputs can be formulated as an optimization problem [a quadratic program (QP)] at each control cycle. The optimization solution specifies the control inputs which are used until the next control cycle. This optimization just considers the current timeslot, to take into account the future timeslots, a finite time horizon is considered for the optimization process, but only the current timeslot will be applied and for each timeslot, this optimization will be repeated. Indeed, the optimization solution determines  $N_c$  (control horizon) future moves  $(\Delta U = [\Delta u(k), \Delta u(k+1), \dots, \Delta u(k+N_c-1)]^T)$  based on the future behavior of the system  $(Y = [y(k+1|k), y(k+2|k), \dots, y(k+N_p|k)]^T)$  over a prediction horizon of  $N_p$ .

### 6.2 Walking objective

Walking is a periodic locomotion which can be decomposed into two main phases: *single support* and *double support*. In the double support phase, the robot shifts its COM to the stance foot and during the single support, its swing leg moves towards the next step position. In order to develop stable locomotion, the robot should be able to track a set of reference trajectories while keeping its stability. As mentioned before, a popular approach to guarantee the stability of the robot is keeping the ZMP within the support polygon. According to (5), the position of the stance leg, swing leg, and ZMP are measured at each control cycle. Thus, the following objectives should be considered to keep the outputs at or near the references:

$$\begin{aligned} d_1 &= \|p_z - r_z\|^2 \\ d_2 &= \|p_{st} - r_{st}\|^2 \\ d_3 &= \|p_{sw} - r_{sw}\|^2, \end{aligned} \tag{7}$$

where  $p_z, r_z, p_{st}, r_{st}, p_{sw}, r_{sw}$  are measured and reference ZMP, and the measure and reference positions of the stance and swing legs, respectively. Moreover, to generate a smooth trajectories which are compatible with the robot structure, the control inputs are considered into the objectives for smoothing all motions:

$$\begin{aligned} d_4 &= \|\ddot{c}_1^x\|^2 \\ d_5 &= \|\ddot{c}_2^x\|^2 \\ d_6 &= \|\ddot{c}_3^x\|^2. \end{aligned} \tag{8}$$

According to the explained objective terms, the following cost function is defined to find an optimal set of control inputs:

$$J(z_k) = \sum_{i=1}^{N_p} \sum_{j=1}^6 \alpha_j d_j(z_k) \tag{9}$$

where  $k$  is the current control interval,  $z_k^T = \{\Delta u(k|k)^T, \Delta u(k+1|k)^T, \dots, \Delta u(k+N_p-1|k)^T\}$  is the QP decision and  $\alpha_j$  represents a positive gain that is assigned to each objective.

### 6.3 Time-varying constraints

To ensure the feasibility of the solution that is found by the MPC, a set of constraints should be considered to avoid generating infeasible solutions. For instance, the solution should be kinematically reachable by the robot and the ZMP should be kept inside the support polygon. Generally, a set of mixed input/output constraints can be specified in the following form:

$$Eu(k+j|k) + Fy(k+j|k) \leq G + \epsilon \tag{10}$$

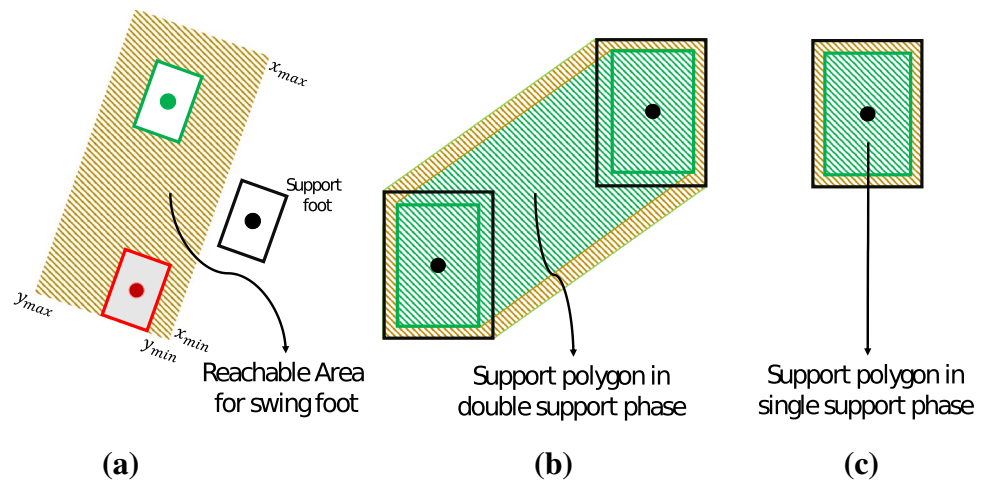
where  $j$  from 0 to  $N_p$ ,  $k$  is current control cycle,  $E, F, G$  are time-variant matrices, where each row represents a linear constraint,  $u(k+j|k)$  and  $y(k+j|k)$  are vectors of manipulated variables and output variables (stance leg, swing leg and ZMP), respectively,  $\epsilon$  is used to specify a constraint to be soft or hard. Additionally, using this equation, the inputs and outputs can be bounded to specified limitations. It should be noted that the constraints in the sagittal plane are similar to those in the frontal plane. In our target framework, the constraints are time-varying and they will be determined at the beginning of each walking phase. Graphical representations of the constraints are depicted in Fig. 5. Figure 5a shows that the landing position of the swing leg should be restricted in a kinematically reachable area. This area can be approximated by a rectangle which is defined as follows:

$$\begin{aligned} E &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad F = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -1 & 0 \end{bmatrix}, \\ G &= \begin{bmatrix} x_{max} \\ x_{min} \end{bmatrix}, \quad \epsilon = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \end{aligned} \tag{11}$$

where  $x_{min}, x_{max}$  are defined based on the current position of the support foot and the robot capability. In addition



**Fig. 5** Graphical representations of the constraints: **a** kinematically reachable area for the swing leg: red rectangle represents the position of the swing leg at the beginning of step, green rectangle represents the landing location of the swing leg; **b** ZMP constraint during double support phase: green area is considered as the support polygon to avoid ZMP from being close to the borders; **c** ZMP constraint during single support phase



to these constraints, to keep the ZMP within the support polygon, the following constraints are considered:

$$E = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad F = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & -1 \end{bmatrix}, \quad (12)$$

$$G = \begin{bmatrix} z_{x_{max}} \\ z_{x_{min}} \end{bmatrix}, \quad e = \begin{bmatrix} 0 \end{bmatrix},$$

where  $z_{x_{min}}, z_{x_{max}}$  are defined based on the walking phase and the size of the foot (please look at Fig. 5b, c). It should be mentioned that the foot size of the robot is considered a bit smaller (scale = 0.9) than the real foot size to prevent ZMP from being too close to the borders of the support polygon.

## 7 Simulation

In this section, a set of simulation scenarios will be designed to validate the performance and examine the robustness of the proposed framework. We firstly performed three simulations using MATLAB to evaluate the performance and robustness of the framework. Afterward, we will deploy our framework on a simulated COMAN (a passively compliant humanoid) [28] humanoid robot to validate the performance of the proposed framework on a full-size humanoid robot.

### 7.1 Simulation using MATLAB

To perform the simulations using MATLAB, a humanoid robot has been simulated according to the dynamics model

presented in Sect. 5. The most important parameters of the simulated robot and the controller are shown in Table 1.

#### 7.1.1 Tracking performance

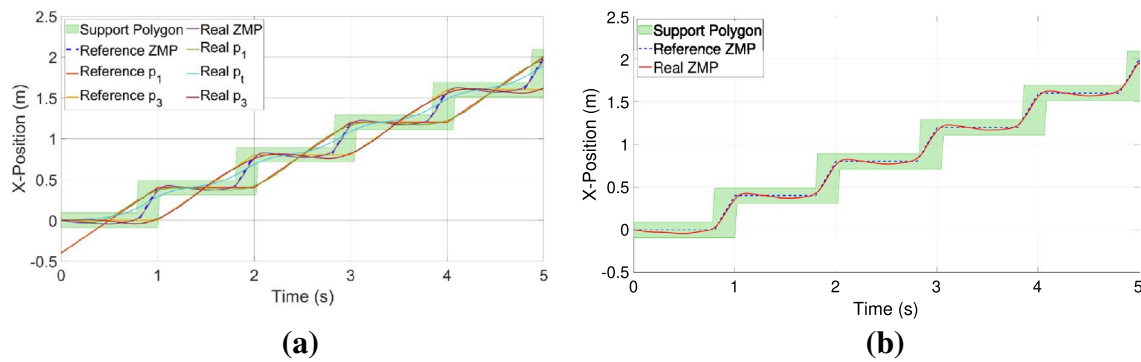
To check the tracking performance of the controller, the simulated robot is commanded to perform a five-step forward walk (step size = 0.8 m and step time = 1 s). In this simulation, the simulated robot is considered to be stopped and stands at the beginning of the simulation and the reference trajectories have been generated based on the presented methods in Sect. 4. The exemplary planned trajectories at the end of Sect. 4 are used as the input references and the controller should track these trajectories while keeping the stability of the robot. The simulation results are depicted in Fig. 6. The results show that the controller is able to track the references and the ZMP is always inside the support polygon during walking. Another interesting point in the results is the actual trajectory of the torso (see  $Real_{p_t}$  in Fig. 6). We did not determine any references for the torso explicitly, but it moves as we expected. As it is shown in Fig. 6a, it is almost near the support foot during the single support phase, then by starting the double support phase, it moves towards the next support foot.

#### 7.1.2 Robustness w.r.t. measurement noise

In the real world, measurements are always affected by noise, therefore they are never perfect. Noise can arise because of many reasons like the simplification in the modeling, discretizing and some mechanical uncertainties (e.g., backlash of gears), etc. A robust controller should be able to estimate the

**Table 1** Parameters used in the simulations

$m_1$	$m_2$	$m_3$	$l_{st}$	$l_t$	$l_{sw}$	Foot length	$T_s$	$N_p$	$N_c$	$\alpha_{1,2,3}$	$\alpha_{4,5,6}$
15 kg	50 kg	15 kg	0.5 m	1.2 m	0.5 m	0.1 m	0.02 s	80	20	20	0.1



**Fig. 6** The simulation results of analyzing tracking performance: **a** represents the positions of the masses while walking; **b** represents the reference ZMP and the real ZMP

correct state using noisy measurements and minimize the effect of noise. To examine the robustness of the proposed controller regarding measurement noise, the measurements are modeled as a stochastic process by adding Gaussian noise ( $-0.05m \leq v_i \leq 0.05m$   $i = 1, 2, 3$ ) to the system output and the previous scenario has been repeated. The simulation results are shown in Fig. 7. As can be seen, the controller is robust against the measurement noise and it can track the references even in the presence of noise.

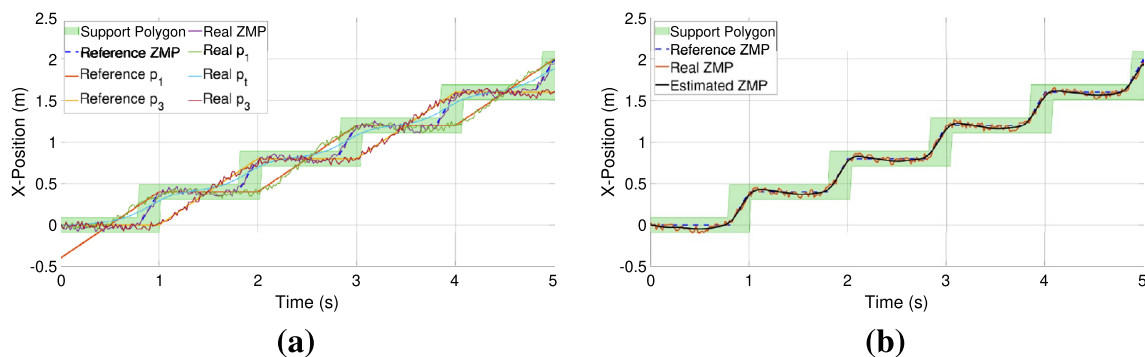
### 7.1.3 Robustness w.r.t. external disturbance

A robust controller should be able to reject an unwanted external disturbance that can occur in some situations like when a robot hits an obstacle or when it has been pushed by someone. In such situations, the controller cancels the effect of the impact and tries to keep ZMP inside the support polygon by applying a compensating torque. To examine the robustness of the controller w.r.t. external disturbances, an unpredictable external force is applied to the torso of the robot while it is performing the previous scenario. The impact has been applied at  $t = 1.6$  s and the impact duration is  $\Delta t = 100$  ms. This

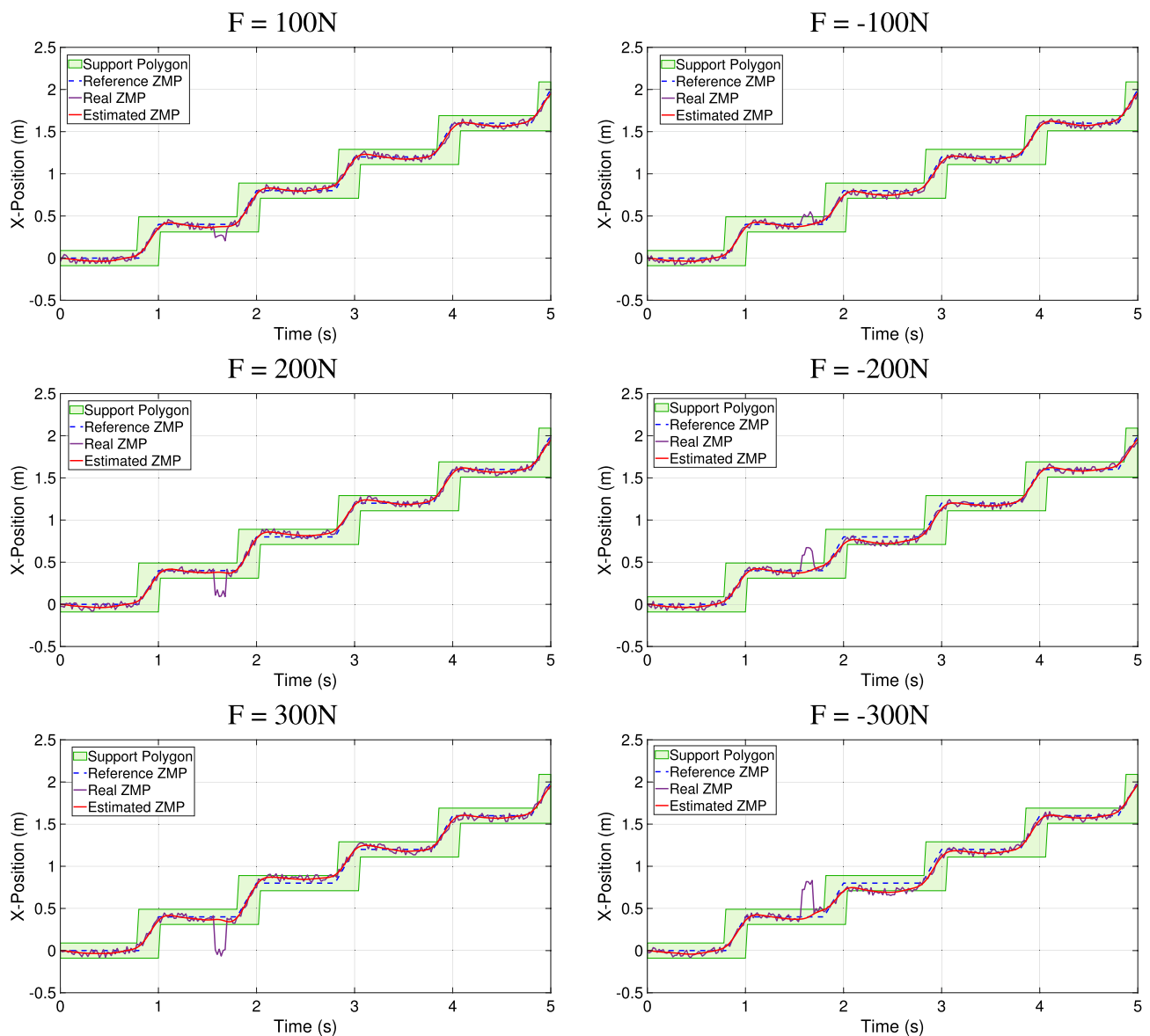
simulation was repeated multiple times with different amplitudes ( $-300 \text{ N} \leq F \leq 300 \text{ N}$ ). Moreover, to have realistic simulations, the measurements are confounded by measurement noise ( $-0.05m \leq v_i \leq 0.05m$   $i = 1, 2, 3$ ). The simulation results are shown in Fig. 8. Each plot represents the result of a single simulation. As these plots show, after applying an external force, the ZMP (fictitious ZMP [31]) goes out of the support polygon quickly whose distance from the support polygon edge proportionally relates to the intensity of the perturbation. The controller regains it back and keeps the stability of the robot. We increase the amplitude of the impact to find the maximum withstanding of the controller. After performing these simulations,  $F = 435 \text{ N}$  and  $F = -395 \text{ N}$  were the maximum levels of withstanding of the controller. According to the simulation results, the proposed controller is robust against external disturbance.

### 7.1.4 Checking the overall performance

To check the overall performance of the proposed framework, the exemplary path planning scenario which has been presented at the end of Sect. 4, is used (see Fig. 4).



**Fig. 7** The simulation results of examining the robustness w.r.t. measurement noise: **a** represents the real and estimated position of the masses; **b** represents the real and estimated ZMP



**Fig. 8** The simulation results of examining the robustness w.r.t. external disturbance: each plot represents a single simulation result. As the results show, after applying external force, ZMP (ficti-

tious ZMP) goes out of support polygon for a moment whose distance from the support polygon edge proportionally relates to the intensity of the perturbation

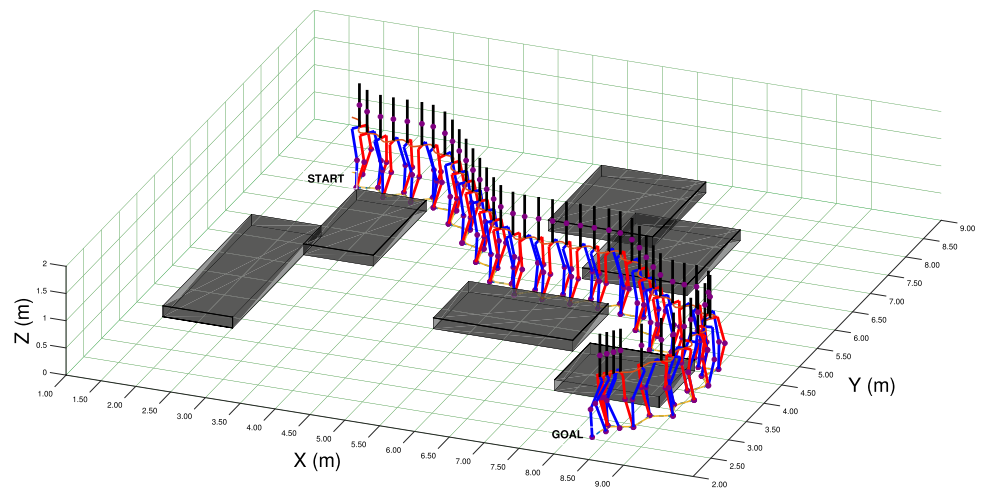
The simulation results are shown in Fig. 9. The results showed that the planner was able to generate walking reference trajectories and the controller was able to track the generated references successfully. A video of this simulation is available online at <https://youtu.be/zyOmgvaXyuA>.

## 7.2 Simulation using COMAN

To validate the portability and platform-independency of the proposed framework and to show the performance of the framework in controlling a full humanoid

robot, we performed a set of simulations using a simulated COMAN humanoid in the Gazebo simulator which is an open-source simulation environment developed by the Open Source Robotics Foundation (OSRF). The simulated robot is 0.95 m tall, weighs 31 kg, and has 23 DOF (6 per leg, 4 per arm, and 3 between the hip and torso). This robot is equipped with the usual joint position and velocity sensors, an IMU on its hip, and torque/force sensors at its ankles.

**Fig. 9** The simulation result of examining the overall performance of the proposed planner and controller. The simulated robot starts from START point and should walk towards the GOAL point while avoiding the obstacles



### 7.2.1 Walking around a disk

This simulation is focused on testing the performance of the framework for combining turning and forward walking. In this simulation, the robot initially stays next to a large disk of radius 1.35 m so that the center of the disk is 1.8 m far from the robot. The robot is waiting to receive a start signal and once the signal is generated, the robot should walk around the disk and return to the initial point while trying to keep 2 m distance from the center of the disk during walking. In this simulation, we fixed the step size by 0.15 m and the turning angle will be determined based on the current position of the robot. The sequences of the experiment are shown in Fig. 10a. In this simulation, the positions of the feet and COM have been recorded and they are depicted in Fig. 10b. The results showed that the framework is able to combine steering and forward walking command to follow a specific path. A video of this simulation is available online at <https://youtu.be/E8PGY05WzIQ>.

### 7.2.2 Omnidirectional walking

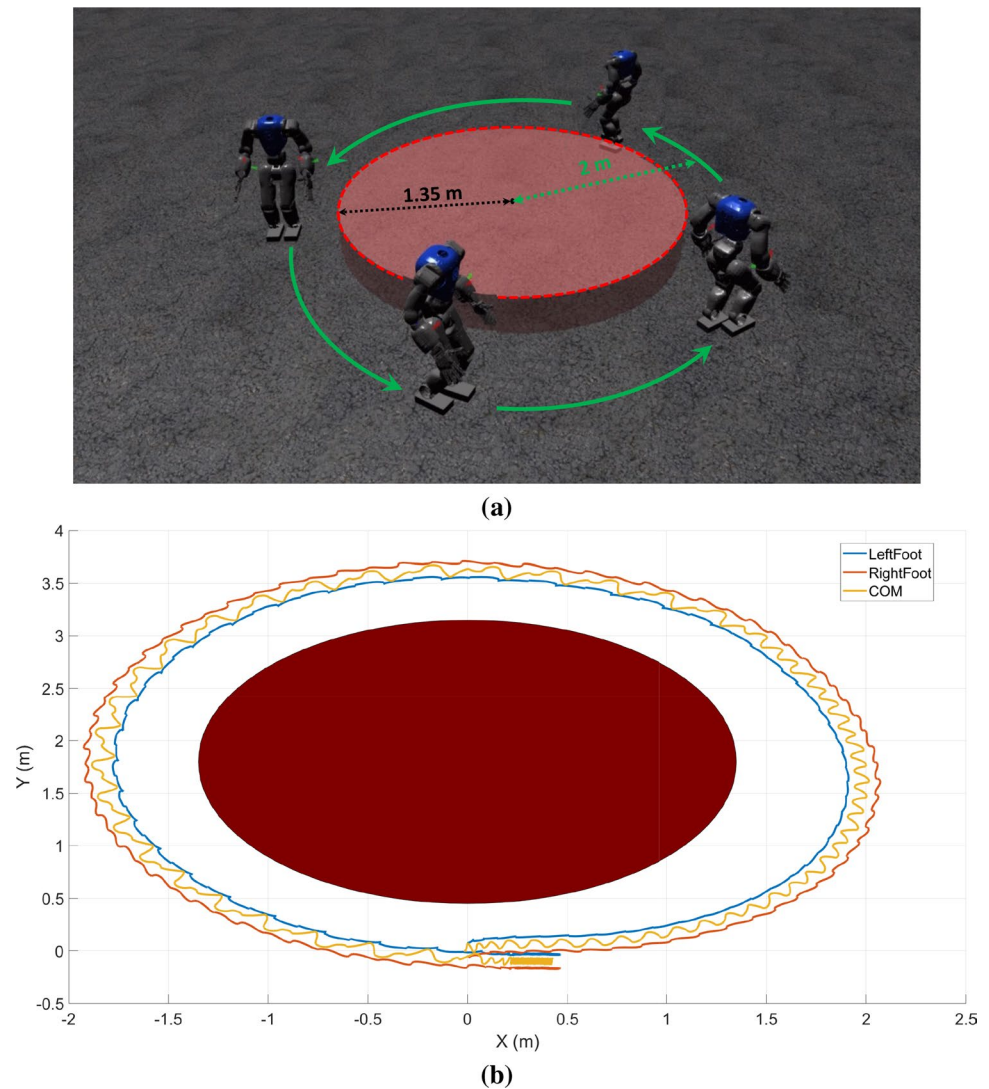
This scenario is focused on validating the performance of the proposed framework for providing omnidirectional walking. In this scenario, the simulated robot should track deterministic setpoints including step length ( $X$ ), step width ( $Y$ ), and step angle ( $\alpha$ ). At the beginning of the simulation, all the setpoints are zero and the robot is walking in place. At  $t = 2$  s, the robot is commanded to walk forward ( $X = 0.1$  m,  $Y = 0.0$  m,  $\alpha = 0.0$  deg/s), At  $t = 10$  s, the robot is commanded to walk diagonally ( $X = 0.10$  m,  $Y = 0.05$  m,  $\alpha = 0.0$  deg/s), at  $t = 20$  s, while it is performing diagonal walking, it is commanded to turn right simultaneously ( $X = 0.2$  m,  $Y = 0.05$  m,  $\alpha = 15$  deg/s), finally at  $t = 32$  s, all set points will be reset and the robot will walking in place. An overview of this scenario is depicted in

Fig. 11a. During this simulation, the positions of the feet and COM have been recorded to examine the behavior of the COM while walking and it is depicted in Fig. 11b. According to the recorded data, the COM tends to the support foot during the single support phase and moves to the next support foot during the double support phase. It should be mentioned that we used a first-order lag filter to update the new setpoints to have a smooth updating. The results showed that the framework can combine all the input commands simultaneously to provide an omnidirectional walking. A video of this simulation is available online at <https://youtu.be/1R56m4t6cHs>.

### 7.2.3 Push recovery

The goal of this simulation is to evaluate the withstanding of the framework in terms of external disturbance rejection. In this scenario, while the robot is walking in place ( $step\ time = 0.4$  s), an impulsive external disturbance will be applied to the middle of its hip and the robot should reject the disturbance and keep its stability while continuing walking in place. To validate the robustness of the framework and to characterize the maximum level of withstanding of the robot, this simulation will be repeated with different amplitudes and a fixed duration of impact (100 ms). The amplitude of the first impact is 80 N and it will be increased every 8 s by 20 N while the amplitude is less than 120 N otherwise by 10 N until the robot falls. Figure 12a shows a snapshot of the simulation while the robot is subject to a severe push 130 N and capable of rejecting this disturbance and keeping its stability. The simulation results show that the framework is robust against external disturbances and  $F = 150$  N was the maximum level of withstanding of the robot. Figure 12b shows the velocity of COM in X direction during this simulation. In this figure, the pink rectangles represent the push duration and as it is shown in this figure, after applying a push, the COM's

**Fig. 10** Walking around a disk: the robot should walk around the disk while keeping 2 m distance from the center of the disk. **a** An overview of the scenario; **b** the feet and COM in the XY plane



velocity increases impressively and the controller could be able to keep the stability. A video of this simulation is available online at <https://youtu.be/os3Dex07Op0>.

#### 7.2.4 Walking on uneven terrains

This scenario is designed to validate the performance of the framework for generating walking on uneven terrain. In this simulation, the robot is placed on an uneven terrain within a square area of side length 4 m.

The robot does not have any information about the terrain and should walk forward to step out of this area. In this simulation, the uneven terrain is generated by placing a set of tiles with the same size but random height (from 25 to 50 mm) next to each other. To have a clear representation of the generated terrain, the color of each tile is considered as a function of its height (see Fig. 13a). This simulation has been repeated using three different tile's size (100 cm<sup>2</sup>, 400

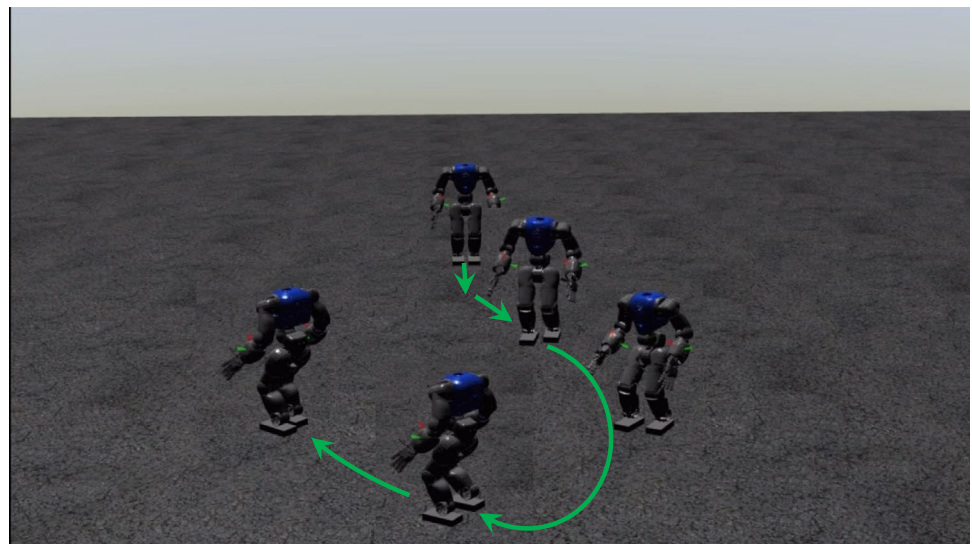
cm<sup>2</sup> and 625 cm<sup>2</sup>). The complexity of the terrain depends on the size and the height of the tiles, the terrain that is generated by the larger tile is more challenging than the others because of more ups and downs and more narrow edges which cause slipping the feet. The height variations of the COM has been recorded during the simulations and are presented in Fig. 13b. The simulation results showed that the framework is capable of providing stable walking on such terrains and handle such uncertainties. A video of this simulation is available online at <https://youtu.be/e9MK6Jy1KHg>.

## 8 Experiments

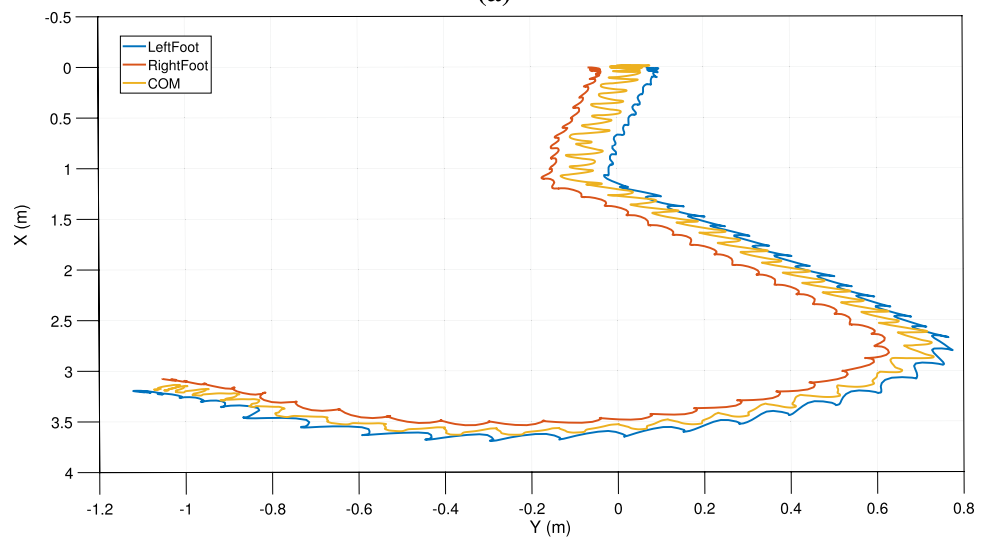
The simulation results showed that the framework was able to generate robust locomotion even in challenging situations. To explore the effectiveness of the proposed



**Fig. 11** Omnidirectional walking scenario: **a** an overview of the omnidirectional walking scenario; **b** the feet and COM in the XY plane



**(a)**

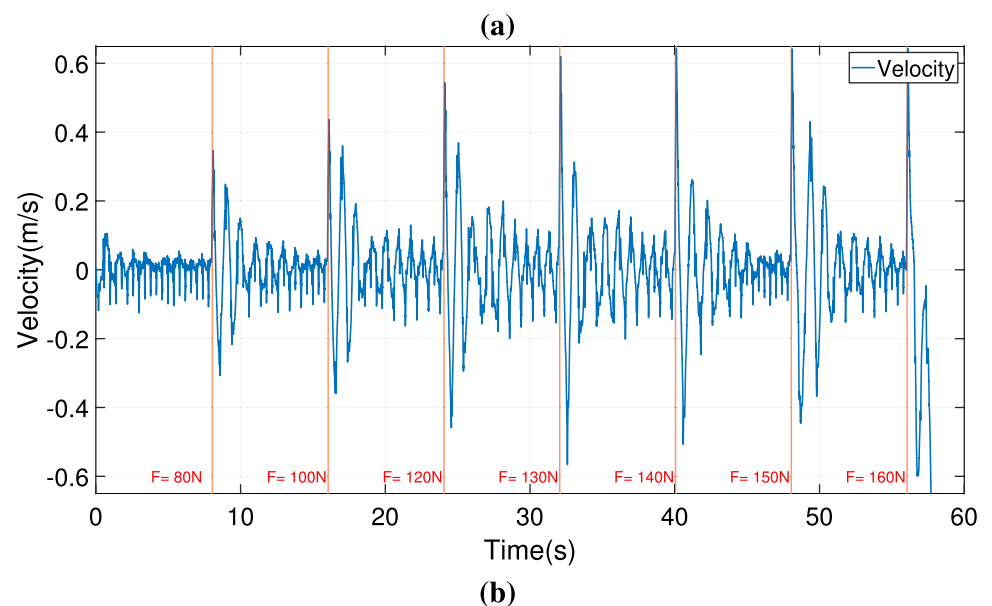
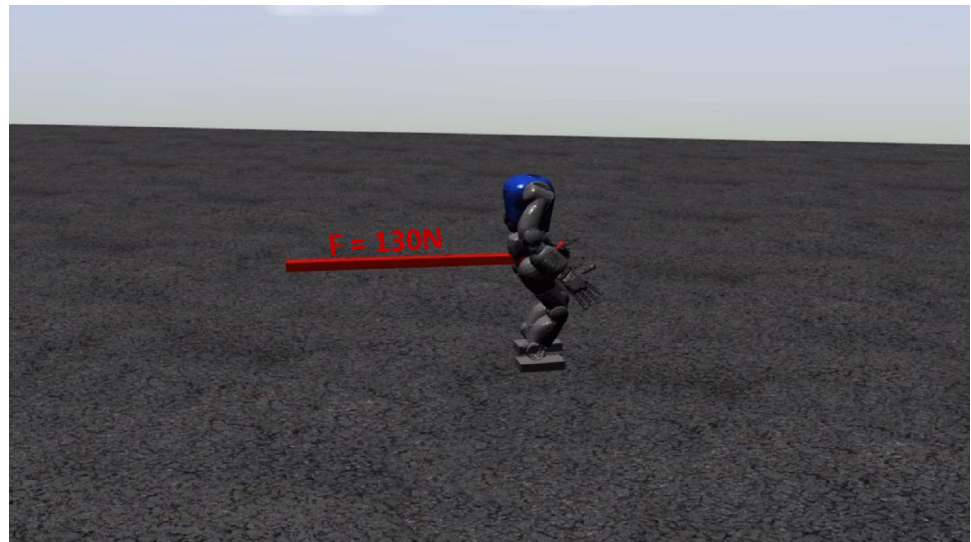


**(b)**

framework regarding the dynamics model and the controller, we developed a baseline framework based on LIPM and our proposed framework and conducted all the presented simulations in the previous section to compare the results. In *Walking Around A Disk* scenario, we increased the complexity of the scenario by increasing the step's lengths from 0.08 m to 0.25 m. The results showed that the baseline framework can successfully complete the simulation by maximum step length of 0.12 m while the proposed framework is capable of completing the simulation by 0.22 m which is 83% better than the baseline. In *Omnidirectional Walking* scenario, although the current commands were challenging for the baseline, both frameworks were able to complete the simulation successfully. Therefore, we defined a scale factor to increase the complexity of the scenario by changing the

input commands. The results showed that the scale factor can be increased up to 1.03 for the baseline and 1.16 for the proposed framework. For both frameworks, the last command ( $X = 0.2 \text{ m}, Y = 0.05 \text{ m}, \alpha = 0.26 \text{ rad/s}$ ) was the most challenging command in the simulation and the robot mostly lost its stability. Therefore, we selected this part of the simulation to compare the performance of both frameworks. The results showed that the proposed framework is 13% better than the baseline. In the next simulation, to compare the maximum level of the withstanding, we performed the *Push Recovery* scenario using the baseline and  $F = 100 \text{ N}$  was the maximum withstanding level which is 50% less than the proposed framework. In the last simulation, we examine the capability of baseline for walking on an uneven terrain. In the two first terrains (small and medium tiles), the robot

**Fig. 12** Push recovery scenario: while the robot is walking in place, an impulsive push will be applied. **a** A snapshot of push recovery scenario; **b** the blue line represents the velocity of COM in X direction during the simulation and the pink rectangles represent the push time and duration



could pass almost half of the terrain and once it put its feet on a narrow edge, it lost its stability and fall down. In the third terrain, it lost its stability immediately. Therefore, the baseline could not pass successfully the uneven terrains. A summary of the simulations results is given in the Table 2.

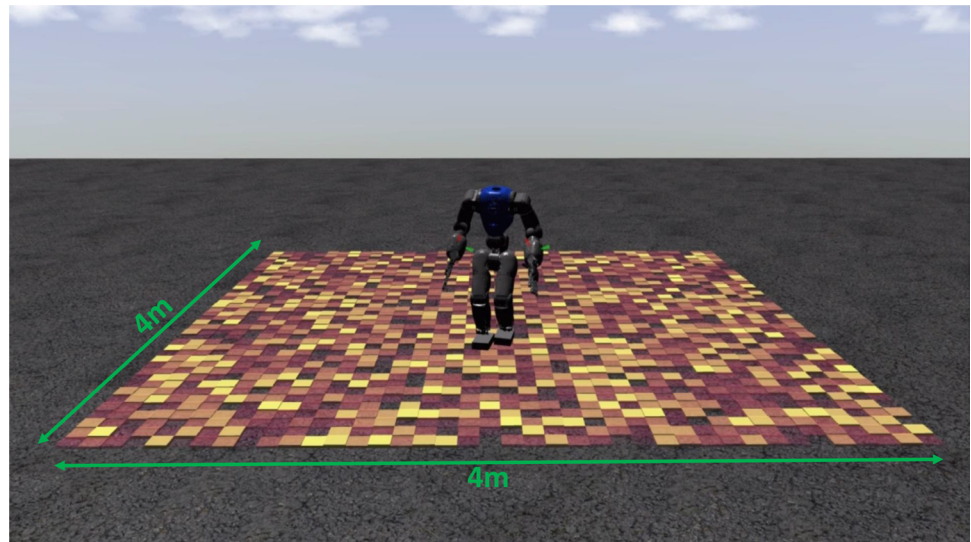
The simulation results validated the performance of the framework and showed that considering the dynamics of torso and legs extremely improved the performance in terms of stability and speed. Unlike [3, 9] we have not used the current state of the system to adjust the planner parameters online, which improves the performance in terms of stability and speed. Also, unlike [3], we did not take into account the vertical motion of masses to keep the linearity of our dynamics

model. Taking into account these motions along with the arm motions can improve the performance impressively.

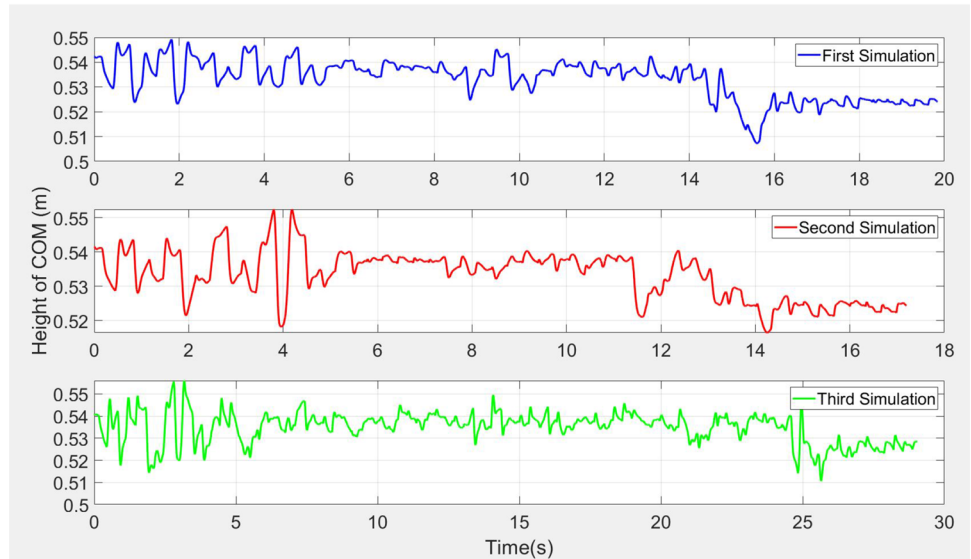
## 9 Conclusion

In this paper, we have developed a model-based walking framework to generate robust biped locomotion. The core of this framework is a dynamics model that abstracts the overall dynamics of a robot into three masses. In particular, this dynamics model and the ZMP concept were used to represent the overall dynamics model of a humanoid robot as a state-space system. Then, this state-space system was used to formulate the walking controller as a linear MPC which generates the control solution using

**Fig. 13** Walking on an uneven terrain: **a** a snapshot of the scenario; **b** the height variation of COM while robot is walking on the uneven terrain



(a)



(b)

**Table 2** Summary of the simulation results

Simulation scenario	Baseline framework	Proposed framework	Improvement (%)
Walking around a disk	Step length: 0.12 m	Step length: 0.22 m	83
Omnidirectional walking	Scale factor: 1.03	Scale factor: 1.16	13
Push recovery	Withstanding: 100 N	Withstanding: 150 N	50
Walking on uneven terrains	Failed	Success	100

an online optimization subject to a set of objectives and constraints. Later, we have presented a hierarchical planning approach which was composed of three main layers to generate walking reference trajectories. To examine the performance of the proposed planner, a path planning

simulation scenario has been designed to validate the performance of the planner. Afterward, according to the planned reference trajectories, a set of numerical simulations has been performed using MATLAB to examine the performance and robustness of the controller. Besides, the

proposed framework has been deployed on a simulated COMAN humanoid robot to conduct a set of simulations using an ODE based simulator environment. The simulation results validated the performance and robustness of the proposed framework. Additionally, the simulation results confirmed that considering the dynamics of torso and legs extremely improved the performance in terms of stability and speed.

In future work, we would like to extend this work to investigate the effect of vertical motion of the masses as well as the arm motions. Moreover, we would like to extend the framework capable of modifying online step position and duration to improve the gait stability. Additionally, we would like to develop a deep reinforcement learning module that combines with the proposed framework to regulate the framework parameters adaptively and to generate residuals to adjust the robot's target joint positions.

**Acknowledgements** This work is an extended version of our previous conference paper [14], focusing on offering more details about the framework and validating its performance, robustness, and portability through a set of simulations. This research is supported by Portuguese National Funds through Foundation for Science and Technology (FCT) through FCT scholarship SFRH/BD/118438/2016 and in the context of the Project UIDB/00127/2020.

## Declarations

**Conflict of interest** On behalf of all authors, the corresponding author states that there is no conflict of interest.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References

1. Abreu M, Reis LP, Lau N (2019) Learning to run faster in a humanoid robot soccer environment through reinforcement learning. In: Robot world cup. Springer, pp 3–15
2. Albert A, Gerth W (2003) Analytic path planning algorithms for bipedal robots without a trunk. *J Intell Rob Syst* 36(2):109–127
3. Brasseur C, Sherikov A, Collette C, Dimitrov D, Wieber PB (2015) A robust linear MPC approach to online generation of 3d biped walking motion. In: 2015 IEEE-RAS 15th international conference on humanoid robots (Humanoids), pp 595–601
4. Chang CH, Huang HP, Hsu HK, Cheng CA (2015) Humanoid robot push-recovery strategy based on cmp criterion and angular momentum regulation. In: 2015 IEEE international conference on advanced intelligent mechatronics (AIM), pp 761–766
5. Dasgupta A, Nakamura Y (1999). Making feasible walking motion of humanoid robots from human motion capture data. In: Proceedings 1999 IEEE international conference on robotics and automation (Cat. No. 99CH36288C), vol 2. IEEE, pp 1044–1049
6. Erbaturo K, Okazaki A, Obiya K, Takahashi T, Kawamura A (2002) A study on the zero moment point measurement for biped walking robots. In: 7th International workshop on advanced motion control. proceedings (Cat. No. 02TH8623). IEEE, pp 431–436
7. Faraji S, Ijspeert AJ (2017) 3lp: a linear 3d-walking model including torso and swing dynamics. *Int J Robot Res* 36(4):436–455
8. Griffin RJ, Wiedebach G, McCrory S, Bertrand S, Lee I, Pratt J (2019) Footstep planning for autonomous walking over rough terrain. arXiv preprint [arXiv:1907.08673](https://arxiv.org/abs/1907.08673)
9. Herdt A, Diedam H, Wieber PB, Dimitrov D, Mombaur K, Diehl M (2010) Online walking motion generation with automatic footstep placement. *Adv Robot* 24(5–6):719–737
10. Herdt A, Perrin N, Wieber PB (2010). Walking without thinking about it. In: 2010 IEEE/RSJ International conference on intelligent robots and systems (IROS). IEEE, pp 190–195
11. Hornung A, Bennewitz M (2012) Adaptive level-of-detail planning for efficient humanoid navigation. In: 2012 IEEE international conference on robotics and automation. IEEE, pp 997–1002
12. Kajita S, Kanehiro F, Kaneko K, Fujiwara K, Harada K, Yokoi K, Hirukawa H (2003) Biped walking pattern generation by using preview control of zero-moment point. In: Robotics and automation, 2003. Proceedings. ICRA'03. IEEE international conference on, vol 2. IEEE, pp 1620–1626
13. Kajita S, Matsumoto O, Saigo M (2001) Real-time 3D walking pattern generation for a biped robot with telescopic legs. In: Robotics and automation, 2001. Proceedings 2001 ICRA. IEEE international conference on, vol 3. IEEE, pp 2299–2306
14. Kasaei M, Ahmadi A, Lau N, Pereira A (2020) A robust model-based biped locomotion framework based on three-mass model: from planning to control. In: 2020 IEEE international conference on autonomous robot systems and competitions (ICARSC). IEEE, pp 257–262
15. Kasaei M, Lau N, Pereira A (2018) An optimal closed-loop framework to develop stable walking for humanoid robot. In: 2018 IEEE international conference on autonomous robot systems and competitions (ICARSC). IEEE, pp 30–35
16. Kasaei M, Lau N, Pereira A (2019) A fast and stable omnidirectional walking engine for the nao humanoid robot. In: Robot world cup. Springer, pp 99–111
17. Kasaei M, Lau N, Pereira A (2019) A robust biped locomotion based on linear-quadratic-Gaussian controller and divergent component of motion. In: 2019 IEEE/RSJ international conference on intelligent robots and systems. IEEE, pp 1429–1434
18. Kasaei SM, Lau N, Pereira A (2017) A reliable hierarchical omnidirectional walking engine for a bipedal robot by using the enhanced lip plus flywheel. In: Human-centric robotics-proceedings of the 20th international conference Clawar 2017. World Scientific, p 399
19. Li C, Lowe R, Ziemke T (2013) Humanoids learning to walk: a natural cpg-actor-critic architecture. *Front Neurobot* 7:5
20. Luo J, Su Y, Ruan L, Zhao Y, Kim D, Sentis L, Fu C (2019) Robust bipedal locomotion based on a hierarchical control structure. *Robotica* 37(10):1750–1767
21. Luo RC, Lee KC, Spalanzani A (2016). Humanoid robot walking pattern generation based on five-mass with angular momentum



- model. In: 2016 IEEE 25th international symposium on industrial electronics (ISIE). IEEE, pp 375–380
22. Mirjalili R, Yousefi-Korna A, Shirazi FA, Nikkhah A, Nazemi F, Khadiv M (2018) A whole-body model predictive control scheme including external contact forces and COM height variations. In: 2018 IEEE-RAS 18th international conference on humanoid robots (Humanoids). IEEE, pp 1–6
  23. Pratt J, Carff J, Drakunov S, Goswami A (2006). Capture point: a step toward humanoid push recovery. In: 2006 6th IEEE-RAS international conference on humanoid robots. IEEE, pp 200–207
  24. Santos CP, Alves N, Moreno JC (2017) Biped locomotion control through a biomimetic cpg-based controller. *J Intell Robot Syst* 85(1):47–70
  25. Saputra AA, Botzheim J, Sulistijono IA, Kubota N (2019) Layered neural-based locomotion for biped robot movement with carrying dynamic payload. *Procedia Computer Science* 159:418–427
  26. Sato T, Sakaino S, Ohnishi K (2010) Real-time walking trajectory generation method with three-mass models at constant body height for three-dimensional biped robots. *IEEE Trans Ind Electron* 58(2):376–383
  27. Shimmyo S, Sato T, Ohnishi K (2013) Biped walking pattern generation by using preview control based on three-mass model. *IEEE Trans Ind Electron* 60(11):5137–5147
  28. Spyarakos-Papastavridis E, Medrano-Cerda GA, Tsagarakis NG, Dai JS, Caldwell DG (2013) A push recovery strategy for a passively compliant humanoid robot using decentralized lqr controllers. In: 2013 IEEE international conference on mechatronics (ICM). IEEE, pp 464–470
  29. Takenaka T, Matsumoto T, Yoshiike T (2009) Real time motion generation and control for biped robot-1st report: walking gait pattern generation. In: Intelligent robots and systems, 2009. IROS 2009. IEEE/RSJ international conference on. IEEE, pp 1084–1091
  30. Tran DH, Hamker F, Nassour J (2018) A humanoid robot learns to recover perturbation during swinging motion. *IEEE Trans Syst Man Cybern Syst*
  31. Vukobratović M, Borovac B (2004) Zero-moment point-thirty five years of its life. *Int J Humanoid Robot* 1(01):157–173
  32. Vukobratovic M, Frank A, Juricic D (1970) On the stability of biped locomotion. *IEEE Trans Biomed Eng* 1:25–36
  33. Wang Y, Xue X, Chen B (2018) Matsuoka's cpg with desired rhythmic signals for adaptive walking of humanoid robots. *IEEE Trans Cybern* 50(2):613–626

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.