Walking on Non-planar Surfaces using an Inverse Dynamic Stack of Tasks

Oscar E. Ramos, Nicolas Mansard, Olivier Stasse, Philippe Souères
CNRS, LAAS, 7 av. du Colonel Roche, F-31400 Toulouse, France
University of Toulouse, LAAS, F-31400 Toulouse, France
Email: {oramos, nmansard, ostasse, soueres}@laas.fr

***Abstract*—This paper presents a method to handle walking on non-planar surfaces. The method obtains the trajectory of the center of mass and the next position of the foot from a pattern generator. Then, an inverse dynamics control scheme with a quadratic programming optimization solver is used to let the foot go from its initial to its final position, controlling also the center of mass and the waist. This solver is able to handle an arbitrary number of contact points. When the swinging foot is going down, collision points are detected and they are added as contact points to the model as soon as they appear. If there are three or more contact points, the foot can safely step, but if there are one or two contact points, the foot rotates properly to generate the largest support polygon. Using this heuristic, the foot can stand on non-planar surfaces. The results show the simulation of HRP-2 walking on a surface with obstacles.**

I. Introduction

Biped walking is a sequencial task that alternates the motion of both legs and that strongly relies on contact forces between the feet and the ground in order to realize stable motions. Static balance keeps the center of mass (CoM) inside the support polygon, whereas dynamic balance keeps the zero moment point (ZMP) inside the support polygon allowing the CoM to be outside of it for a certain period of time. Dynamic balance, which represents the way that humans walk, is the preferred walking strategy in robotics research but classical schemes allow mainly walking only on planar surfaces.

However, there are several works that have succeded to walk on uneven terrain. In [1], a control method for adapting to a horizontally composed plane with unknown steps height is proposed. The method allows walking on non-flat floor by using the sole sensors that measure a distance between the sole itself and the ground before landing. In [2], a pattern generation system that updates the CoM trajectory in a short cycle using preview control and a feedack controller that tracks the ZMP reference was proposed. This method also enabled a robot to change the walking pattern according to the change of applied force. A method that uses preview control from the current inclination of the upper body to a stable future state in order to update the upper body trajectory was proposed in [3]. Another method [4] is based on a predictive attitude compensation control and a nonlinear compliance control using only force sensors and a particular biped foot system that adapts to a rough terrain.A technique to stabilize the upper body postion under varying ground reaction forces is proposed in [5]. This technique generates moments to handle disturbances that cannot be handled by ground reaction forces by using horizontal and rotational acceleration of the upper body and step duration change: as a result, a landing position is locally modified.

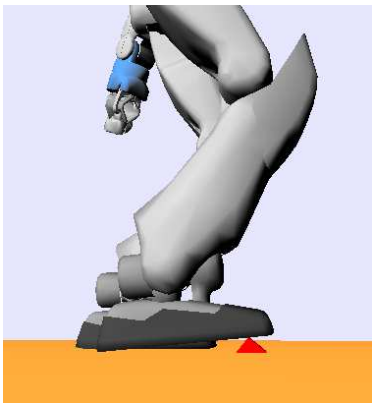


Fig. 1: The robot HRP-2 stepping over a non-planar obstacle with its right foot.

More recently, a strategy for adjusting the ZMP reference to generate online repetitive walking has been proposed [6]. The strategy consists on changing the reference ZMP inside the sole area, changing the position of the next step and changing the duration of the current step. In [7], proper impedance gains of the feet and updates of the desired landing position of the CoM pattern after a contact is detected are used as a method for a collision of a swinging foot on uneven terrain.

In this paper, we propose to use a generic hierarchical optimization solver to track the position of the CoM and to autonomously control the swinging foot based on the next foot position on the ground given by a pattern generator. The considered solver consists of the inverse-dynamics control cascade proposed in [8], [9], and modified in [10] to allow faster computations. As the foot goes down, possible collisions are detected and handled by the solver as contact points, and, if necessary, the foot is properly rotated to move towards the ground without losing dynamic stability. Figure 1 shows an example of an obstacle on which the robot can stand keeping its stability.

This paper is organized as follows. Section II presents the inverse dynamics control including the contact constraints. Section III gives a brief summary of the pattern generator

used in this work. The usage of the inverse dynamics scheme with the pattern generator is presented in Section IV. Finally, the results of the simulation are presented in Section V.

II. Inverse dynamics control

A. Generic Dynamic Model

The humanoid robot configuration will be represented by the generalized coordinates $q = [x_b \; q_a]^T$, where x_b represents the position and orientation of the robot (more specifically, the base, free-floating or free-flyer) with respect to the world, and q_a represents the n actuated joints. It will be assumed that there are n_f points in contact with the environment. Let $x_p = [x_1 \; x_2 \; \cdots \; x_{n_f}]^T \in \mathbb{R}^{3n_f}$ represent the contact points in the world frame, and $f = [f_1 \; f_2 \; \cdots \; f_{n_f}]^T \in \mathbb{R}^{3n_f}$ represent the punctual contact forces, with f_i acting at point x_i. The dynamic model of the robot is

$$S^T\tau = A\ddot{q} + b + J_f^T f \tag{1}$$

where $S = [0 \; I]$ is a matrix that selects the actuated joints, τ is the torque vector on the joints, A is the whole-body inertia matrix, b is the drift including Coriolis, centrifugal and gravity forces, and $J_f = \frac{\partial x_p}{\partial q}$ is the Jacobian of the contact points.

Additionally, for rigid contacts, the contact points must not move and the restriction $\ddot{x}_p = 0$ must be added. This restriction is equivalent to

$$J_f\ddot{q} + \dot{J}_f\dot{q} = 0 \tag{2}$$

provided that $\dot{x}_p = J_f\dot{q}$. Let S^n be a matrix that selects the normal component (third component) of each force in f and of each point in x_p, with its elements given by $\{s^n_{ij}\} = \delta(3i - j)$, where δ represents the Kronecker delta function. When considering the model of "point contact" [11], the entire contact constraint is described simultaneously by $S^n x_p = 0$ and $S^n f \geq 0$, if the contact persists. The first part of the constraint is already satisfied by (2), and the second part relating the contact forces

$$S^n f \geq 0 \tag{3}$$

must be explicitly considered in the model. The unidirectionallity of (3) guarantees that the ZMP will exist inside the support polygon, ensuring the dynamic stability condition [9].

B. Decoupled Dynamics

The explicit separation of the dynamic equations into motion and actuation has been proposed in [10]. The decoupling allows a faster computation since the decoupled variables are of lower dimension than the coupled original variables. This section will give a brief review of the decoupled model. Using the Choleski decomposition, the inverse of the inertia matrix A can be expressed as $A^{-1} = BB^T$, and thus $A = B^{-T}B^{-1}$, with B an invertible triangular matrix. Multiplying the dynamic model (1) by B^T on the left we obtain:

$$B^T S^T\tau = B^{-1}\ddot{q} + B^T b + G^T f \tag{4}$$

where $G = J_f B$. The result of multiplying (4) by G on the left, using (2) to guarantee no kinematic motion at the rigid contact points, is:

$$GB^T S^T\tau = GB^T b + GG^T f - \dot{J}_f\dot{q} \tag{5}$$

and solving for $B^T S^T\tau$ gives

$$B^T S^T\tau = B^T b + G^T f + \delta_c + Vu \tag{6}$$

where $\delta_c = -G^+\dot{J}_f\dot{q}$ is the contact drift, V is a basis of the kernel of G such that $VV^T = I - G^+G$, and u is a vector in this kernel. The matrix $\bar{S} = [I \; 0]$ is defined so that it cancels out S^T ($\bar{S}S^T = 0$), and thus the torque. By multiplying both terms of (6) by $\bar{S}B^{-T}$ on the left, a torque-free relation between u and the contact forces f is obtained:

$$0 = \bar{S}b + \bar{S}J_f^T f + \bar{S}B^{-T}\delta_c + \bar{S}B^{-T}Vu \tag{7}$$

This expression is composed of both (1) and (2), but explicitly using u and f as variables. The other contact condition related to the unidirectionallity of the forces (3) can be directly included in the decoupled model as $S^n f \geq 0$.

It is also possible to recover the torques from this model multiplying (6) by SB^{-T} on the left to obtain:

$$\tau = Sb + SJ_f^T f + SB^{-T}(\delta_c + Vu) \tag{8}$$

where the fact that $SS^T = I$ has been used.

C. Representation of the tasks

For any observable $s_i(q)$ of the robot configuration, a desired position and orientation s_i^* can be specified without loss of generality by the i^{th} task $e_i = s_i - s_i^*$. A usual choice for the reference acceleration of the task is the proportional derivative (PD) control law $\ddot{e}_i^* = -K_p e_i - K_v\dot{e}_i$ with $K_p > 0$ and $K_v = 2\sqrt{K_p}$, to make the error decrease exponentially to zero. At the acceleration level, the classical relation between the task space and the joint space becomes:

$$\ddot{e}_i = \dot{J}_i\dot{q} + J_i\ddot{q} \tag{9}$$

where $J_i = \frac{\partial e_i}{\partial q}$ is the i^{th} task Jacobian. The equivalence between (4) and (6) leads to the direct relation between u and the acceleration $\ddot{q}$ as:

$$\ddot{q} = B\delta_c + BVu \tag{10}$$

so that u can be interpreted as a "motion" variable. Then, using (10), the generic task representation (9) can be equivalently expressed in terms of the decoupled variable u as:

$$\ddot{e}_i = \dot{J}_i\dot{q} + J_i B\delta_c + J_i BVu \tag{11}$$

D. Spatial-force reduction

Let $x_c \in \mathbb{R}^6$ represent the position and orientation of a central point rigidly attached to the foot (for instance, its center of mass, or the ankle). The 6D Jacobian associated

with this frame is $J_c = \frac{\partial x_c}{\partial q}$, and the relation between J_c and the Jacobian J_f in (1) is $J_f = XJ_c$, with [10]:

$$X = \begin{bmatrix} I & -\hat{p_1} \\ \vdots & \vdots \\ I & -\hat{p_{n_f}} \end{bmatrix} \tag{12}$$

where p_i is the position of the contact point i with respect to the frame x_c, and $\hat{p_i}$ represents the skew-symmetric matrix obtained from p_i equivalent to a cross product pre-multiplication. When there are three or more contact points on the same body, X is full-column rank and the null space of J_f is equal to the null space of J_c. In this case, J_c can be used to compute the V basis as $V = ker(J_cB)$ instead of $V = ker(J_fB)$, and the contact drift as $\delta_c = -(J_cB)^+\dot{J}_c\dot{q}$. However, the null spaces are not equal when there are only 1 or 2 contact points, which is a case that will be discussed in section IV-B

E. QP for the dynamic inversion

The robot motion implies the fulfillment of one or more tasks satisfying at the same time the dynamic model and the contact constraints. Under this premise, the optimization problem for a task i is expressed using (11), (7), and (3) as

$$\min_{u,f} \parallel J_iBVu + \dot{J}_i\dot{q} + J_iB\delta_c - \ddot{e}_i^* \parallel$$

$$\text{s.t.} \quad (\bar{S}B^{-T}V)u + (\bar{S}J_f^T)f = -\bar{S}(b + B^{-T}\delta_c)$$
$$S^n f \geq 0$$

In [8], it has been proposed to use a hierarchical QP solver (HQP) [12] to solve the operational-space inverse dynamics. The hierarchy is denoted by $\prec$ where $a \prec b$ means that a has more priority than b. With this notation, the dynamic stack of m tasks is (7) $\prec$ (3) $\prec$ (11.1) $\prec \cdots \prec$ (11.m), and the optimization variables determined by the HQP solver are the decoupled u and f. Using these optimization variables, the torque τ obtained from (8) can be used as input to a torque-controlled robot or simulator, and the acceleration $\ddot{q}$ from (10) can be integrated to be used as input to a position-controlled robot.

III. Pattern Generator

The pattern generator assumes that the robot CoM is kept at a certain constant distance z^{cm} above the ground, so that the position of the CoM can be represented only by two components (x^{cm}, y^{cm}). For simplicity in notation, $x_k^{cm} = x^{cm}(kT) = x^{cm}(t_k)$, where T is the sampling period and k is the k-th sample, and the state variables are $\hat{x}_k = [x_k^{cm}\ \dot{x}_k^{cm}\ \ddot{x}_k^{cm}]^T$. The ZMP position on the ground plane is represented as (z^x, z^y). In the following treatment only the x components will be explicited, but the y components are obtained in a similar way. The discrete dynamic system relating the ZMP and the CoM is [13]:

$$\begin{bmatrix} x_{k+1}^{cm} \\ \dot{x}_{k+1}^{cm} \\ \ddot{x}_{k+1}^{cm} \end{bmatrix} = \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_k^{cm} \\ \dot{x}_k^{cm} \\ \ddot{x}_k^{cm} \end{bmatrix} + \begin{bmatrix} T^3/6 \\ T^2/2 \\ T \end{bmatrix} \dddot{x}_k^{cm} \tag{13}$$

$$z^x = \begin{bmatrix} 1 & 0 & -z^{cm}/g \end{bmatrix} \begin{bmatrix} x_k^{cm} \\ \dot{x}_k^{cm} \\ \ddot{x}_k^{cm} \end{bmatrix} \tag{14}$$

Considering a prediction horizon of N samples, and using the previous dynamics recursively, it can be shown [3] that the velocity of the CoM from time t_{k+1} to t_{k+N} is expressed as:

$$\dot{X}_{k+1} = \begin{bmatrix} \dot{x}_{k+1}^{cm} & \cdots & \dot{x}_{k+N}^{cm} \end{bmatrix}^T = P_{vs}\hat{x}_k + P_{vu}\dddot{X}_k \tag{15}$$

where $\dddot{X}_k = [\dddot{x}_k^{cm} \ \cdots \ \dddot{x}_{k+N-1}^{cm}]^T$ is the jerk of the CoM from time t_k to t_{k+N-1}, and

$$P_{vs} = \begin{bmatrix} 0 & 1 & T \\ \vdots & \vdots & \vdots \\ 0 & 1 & NT \end{bmatrix},\ P_{vu} = \begin{bmatrix} T^2/2 & 0 & 0 \\ \vdots & \ddots & 0 \\ (1+2N)T^2/2 & \cdots & T^2/2 \end{bmatrix}$$

Also, the ZMP from time t_{k+1} to time t_{k+N} is expressed as:

$$Z_{k+1}^x = [z_{k+1}^x \ \cdots \ z_{k+N}^x]^T = P_{zs}\hat{x}_k + P_{zu}\dddot{X}_k \tag{16}$$

where

$$P_{zs} = \begin{bmatrix} 1 & T & T^2/2 - z^{cm}/g \\ \vdots & \vdots & \vdots \\ 1 & NT & N^2T^2/2 - z^{cm}/g \end{bmatrix}$$

$$P_{zu} = \begin{bmatrix} T^3/6 - Tz^{cm}/g & 0 & 0 \\ \vdots & \ddots & \vdots \\ (1+3M+3M^2)\frac{T^3}{6} - \frac{Tz^{cm}}{g} & \cdots & \frac{T^3}{6} - \frac{Tz^{cm}}{g} \end{bmatrix}$$

with $M = N - 1$.

The pattern generator used in this work is the one proposed in [14], which basically regulates the speed of the CoM and obtains the foot placement as output of the optimization process. The considered optimization variable is $u_k = [\dddot{X}_k\ \dddot{Y}_k\ X_k^f\ Y_k^f]^T$, where (X_k^f, Y_k^f) corresponds to the positions on the ground of the following m foot steps. The optimization problem in [14] is stated as:

$$\begin{aligned} \min_{u_k} \quad & \frac{\alpha}{2}\left(\parallel \dot{X}_{k+1} - \dot{X}_{k+1}^{ref} \parallel^2 + \parallel \dot{Y}_{k+1} - \dot{Y}_{k+1}^{ref} \parallel^2\right) \\ & + \frac{\beta}{2}\left(\parallel Z_{k+1}^x - Z_{k+1}^{x\_ref} \parallel^2 + \parallel Z_{k+1}^y - Z_{k+1}^{y\_ref} \parallel^2\right) \\ & + \frac{\gamma}{2}\left(\parallel \dddot{X}_k \parallel^2 + \parallel \dddot{Y}_k \parallel^2\right) \end{aligned} \tag{17}$$

where $\dot{X}_{k+1}^{ref}$ and $\dot{Y}_{k+1}^{ref}$ are the desired mean value for the speed of the CoM, and $Z_{k+1}^{x\_ref}$, $Z_{k+1}^{y\_ref}$ are the references for the ZMP. These ZMP references are not fixed in advanced but are permanently recomputed from the feet position decided by the algorithm so that the ZMP lies in the middle of the foot. Let (X_k^{fc}, Y_k^{fc}) be the current position of the foot on the ground. Then, the ZMP references are given by:

$$Z_{k+1}^{x\_ref} = U_{k+1}^c X_k^{fc} + U_{k+1} X_k^f \tag{18}$$

$$Z_{k+1}^{y\_ref} = U_{k+1}^c Y_k^{fc} + U_{k+1} Y_k^f \tag{19}$$

with

$$U^c_{k+1} = \begin{bmatrix} 1 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad U_{k+1} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 1 & 0 & \cdots & 0 \end{bmatrix}$$

Additionally, during the single support phase, the constraint on the ZMP to ensure that it lies inside the support polygon is expressed as:

$$\begin{bmatrix} d_x(\theta) & d_y(\theta) \end{bmatrix} \begin{bmatrix} z^x - x^f \\ z^y - y^f \end{bmatrix} \leq b(\theta) \tag{20}$$

where (x^f, y^f) is the position of the foot, θ is its orientation, $d_x(\theta)$, $d_y(\theta)$ are column vectors containing the x, y coordinates of the normal vectors to the edges of the feet, and $b(\theta)$ is the column vector containing their position with a security margin.

IV. Using the pattern generator with the inverse dynamic control scheme

The pattern generator gives the desired trajectory of the CoM and of the swinging foot. Classical schemes track these elements using only kinematics and no notion of contact with the ground. The method proposed here uses the inverse dynamics stack of tasks control scheme described in section II with the online output of the pattern generator. While the CoM and the wais are tracked as tasks, the swinging foot is not tracked. Since the solver can handle the notion of contact with a surface (the ground or a local irregularity), only the next position of the swinging foot is considered. Then, an interpolation task to take the foot from its initial position to its final position is used. This task is interrupted as soon as a contact appears leading to a possible rotation of the foot to reach stability.

A. Interpolation task

The interpolation task is used to take the swinging foot from its initial position to its final position. The objective is to achieve a desired final position and velocity in a specific period of time, assuming that the current position and velocity are known. Let the evolution of a generic feature be described by $x(t)$, and let the initial time be t_0 with corresponding initial position and velocity given by $x_0 = x(t_0)$ and $\dot{x}_0 = \dot{x}(t_0)$, respectively. After a desired time duration T, the desired position and velocity will be $x_f = x(t_0 + T)$ and $\dot{x}_f = \dot{x}(t_0 + T)$. The control is performed via the acceleration $\ddot{x}(t)$ which is set to a linear function of the form [15]:

$$\ddot{x}(t) = \ddot{x}_0 + \frac{\ddot{x}_f - \ddot{x}_0}{T}(t - t_0) \tag{21}$$

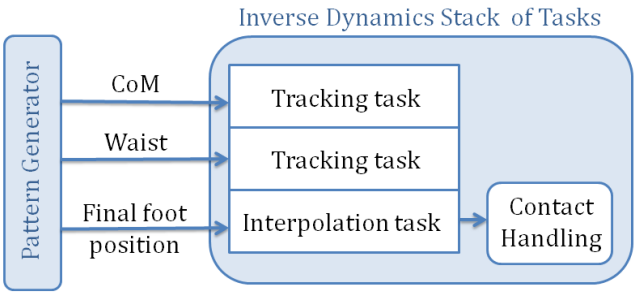


Fig. 2: Scheme of the usage of the pattern generator with the inverse dynamics stack of tasks.

whose solution leads to quadratic velocity and cubic position trajectories given by:

$$\dot{x}(t) = \frac{\ddot{x}_f - \ddot{x}_0}{2T}(t - t_0)^2 + \ddot{x}_0(t - t_0) + \dot{x}_0 \tag{22}$$

$$x(t) = \frac{\ddot{x}_f - \ddot{x}_0}{6T}(t - t_0)^3 + \frac{\ddot{x}_0}{2}(t - t_0)^2 + \dot{x}_0(t - t_0) + x_0 \tag{23}$$

The unknown parameters $\ddot{x}_0$ and $\ddot{x}_f$ can be obtained evaluating (22) and (23) at time t_f. However, as time passes by, the initial position and velocity as well as the time remaining for t_f are updated, and a new acceleration trajectory (21) is computed. Then, since (21) is permanently updated, only the initial acceleration $\ddot{x}_0$ is needed (accelerations at the following instants of time are of no interest), and the dynamic task in (11) is simply defined as $\ddot{e}^* = \ddot{x}_0$, which can be obtained by solving the system (22) and (23) at time t_f resulting:

$$\ddot{e}^* = \ddot{x}_0 = \frac{6}{T^2}(x_f - x_0) - \frac{2}{T}(\dot{x}_f + 2\dot{x}_0) \tag{24}$$

It should be noted that T becomes smaller after each iteration and is eventually close to zero as the initial time reaches the final time. To handle this situation, when $T < t_h$, where t_h is a threshold, the previous value of $\ddot{x}_0$ is kept, which in practice gives good results.

B. Handling 1 and 2 contact points

One of the main objectives of this control scheme is to explicitly handle the case of a rigid body with only 2 or 1 contact point(s) with the environment. In this situation, X in (12) is not full-column rank, and the reductions where J_c could be used instead of J_f do not necessarily hold since both matrices do not have the same null space anymore.

In this case, the equivalence $J_f = XJ_c$ has to be used and the computation of V will be explicitly $ker(XJ_cB)$. The relation between the derivative of both Jacobians is $\dot{J}_f = X\dot{J}_c + \dot{X}J_c$, but the second term is null provided that $\dot{X}$ is null (the contact points do not move) leading to $\dot{J}_f = X\dot{J}_c$. Then, for the contact drift, $\delta_c = -(XJ_cB)^+ X\dot{J}_c\dot{q}$.

C. Walking generation scheme

In order to make the robot walk, the output of the pattern generator is used as input to the inverse dynamics solver as

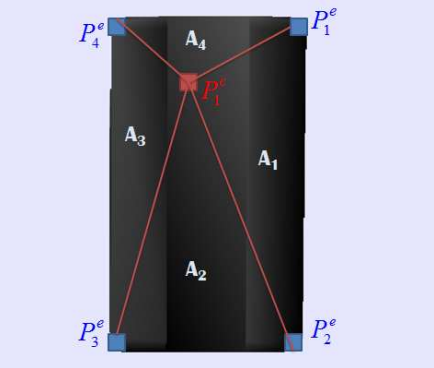
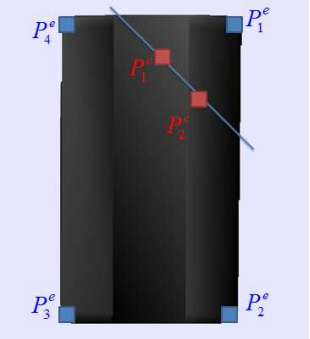
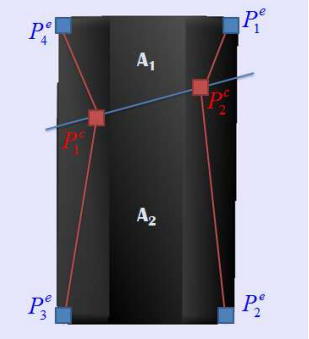


Fig. 3: A single contact point on the sole of the robot.



(a) Case 1



(b) Case 2

Fig. 4: Two contact points on the sole of the robot.

shown in Figure 2. The dynamic solver (section II-E) considers the following three tasks:

- Task to track the trajectory of the CoM. The pattern generator gives the desired trajectory for the center of mass assuming a constant height. This task controls only the x and y components of the CoM.
- Task to partially track the waist trajectory. This task aims to control the height (position in z) of the waist at a certain constant value (since this is the basic premise of the pattern generator), as well as the orientation in x, y. It is partial because not all the six degrees of freedom are controlled.
- Interpolation tasks on the swinging foot. The pattern generator gives the next final position of the foot on the ground. This task takes the swinging foot from its initial position to its desired final position. There are two interpolation steps: the first task takes the foot from the initial position to an intermediate position that lies halfway between the initial and final positions with a predetermined height. The second interpolation takes the foot from the intermediate to the final position, if possible. If a contact is detected before arriving to the final position, the collision is handled as described in the following section. The interpolation task (section IV-A) guarantees that the time constraint is fulfilled according to the pattern generator.

D. Non-horizontal foot contacts handling

The second interpolation for the foot, described in the previous section, would ideally end without interruption if the ground is flat. However, if there is a non-planar surface, generated by the own irregular ground or by a small obstacle (whose size is assumed to be smaller than the step height), the interpolation will not end but will be interrupted as soon as some point(s) on the foot sole touch(es) a rigid object. In this case, there are three possibilities for the contact situation:

- There is only one contact point
- There are two contact points
- There are three or more contact points

If there are more than three contact points, the foot can step on those points, which will generate the support polygon. When there are less than 3 contact points, the foot still needs to move to find at least one more contact. In this case, if the foot is left uncontrolled after the first contact(s), the robot dynamics might take the foot to an unstable position. In order to avoid these instabilities, the foot extremes have to be controlled so that the maximum support polygon is obtained.

If there is only one contact point, the foot is in the situation shown by figure 3, where the single contact point is p_1^c and the foot extremes are denoted by p_1^e, p_2^e, p_3^e and p_4^e. Four triangles are formed by joining the contact point to two consecutive foot extremes. The area of the triangle formed by the consecutive foot extremes p_i^e and p_{i+1}^e is given by

$$A_i = 0.5 \parallel (p_1^c - p_i^e) \times (p_1^c - p_{i+1}^e) \parallel \tag{25}$$

The triangle with the greatest area will contain the extremes of the foot that are farther from the contact point, and thus, it is desirable to take those extremes to the ground so that the largest support polygon is obtained. To that end, a task is assigned to each of these extremes controlling only the vertical z position so that they go to the ground. While these points are moving, if a contact point is detected, it is added as a contact to the solver and its position is checked against the points that were going to the ground. The task for the closest extreme is removed. The foot continues its rotation until another contact point is detected, in which case, the remaining extreme task is removed.

If there are only two contact points, there are two possibilities shown in figure 4. To obtain the case, a line L is passed through both contact points and the extremes of the foot are determined to lie in one side or on the other side of the line. Let the contact points be p_1^c and p_2^c, the vector joining these points be $v = (v_x, v_y) = p_1^c - p_2^c$, and the vector joining one point with one extreme of the foot be $v_i = (v_{ix}, v_{iy}) = p_i^e - p_2^c$. The idea is to rotate the points so that the line L is aligned with the vertical line. Then, the sign of the arctangent can be used to determine the side of the line in which a point lies. The angle that line L must rotate to be aligned with the vertical is $\theta = atan2(v_y, v_x)$. Then, each v_i is rotated by the angle θ as:

$$v_{ix}^f = v_{ix}cos(-\theta) - v_{iy}sin(-\theta) \tag{26}$$

$$v_{iy}^f = v_{ix}sin(-\theta) + v_{iy}cos(-\theta) \tag{27}$$

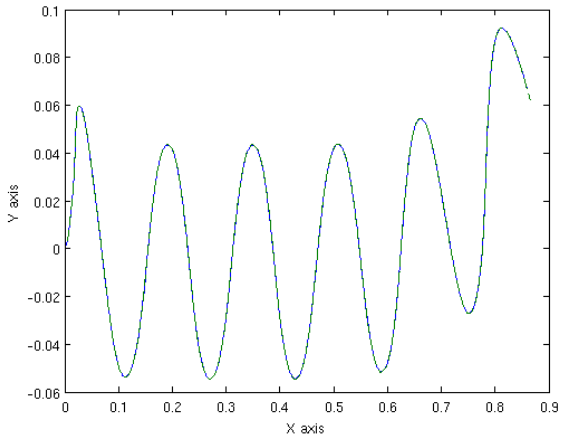


Fig. 6: Evolution of the center of mass in the XY (horizontal) plane. The dashed line shows the reference of the pattern generator and the solid line shows the tracking result of the controller.

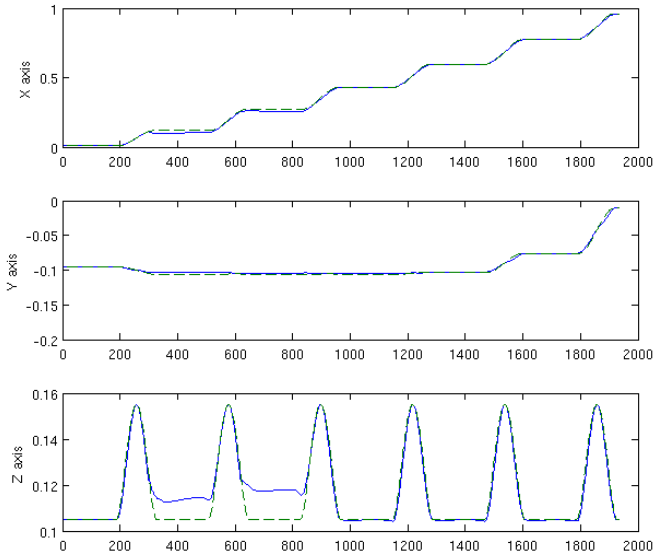


Fig. 7: Trajectory of the right foot in xyz. The dashed line shows the trajectory of the pattern generator and the solid line shows the trajectory generated by the inverse dynamics controller.

After this rotation, the angle to the line is determined as $\phi = atan2(v_{iy}^f, v_{ix}^f)$, the sign of ϕ indicating whether the point is on one side or the other of line L. If only one point is on one side and three are on the other side (Figure 4a), the three points are taken to the ground. If two points are on each side (Figure 4b), then, the area of the quadrilateral formed by the contact points and the extremes on each side is computed and the largest area indicates that those extremes are farther and must be taken to the ground. As in the case of a single contact, as soon as a new contact appears, it is added as a contact to the solver, and the task for the closest extreme is removed.

V. Results

To validate the proposed scheme, a simulation of the robot HRP-2 considering its full dynamics was performed on a ground where random objects were presented. The height of the obstacles was always smaller than the step height. As described in the paper, only the CoM and the waist are tracked by the solver. The foot position are specified using two interpolation tasks. The robot first walks in a straight line and then slightly turns left. Some snapshots of the robot are shown in Figure 5, and a video of the robot walking can be found in homepages.laas.fr/oramos/humanoids2012/walking.avi. The snapshots show the right and left foot stepping over different obstacles in different conditions. Since there is no control on the arms, they move freely to help keep the CoM position.

Figure 6 shows the tracking of the center of mass using the inverse dynamics stack of tasks. It can be observed that the CoM is properly tracked by the system. Figure 7 shows the trajectory of the right foot. The dashed line shows the trajectory that is obtained from the pattern generator, which assumes a constant ground height. As stated before, the right foot does not follow this trajectory but only uses the final position of the foot. In this experience, the two first steps of the right foot find an obstacle on the ground. It can be observed in the curves with solid line that even though the foot is not

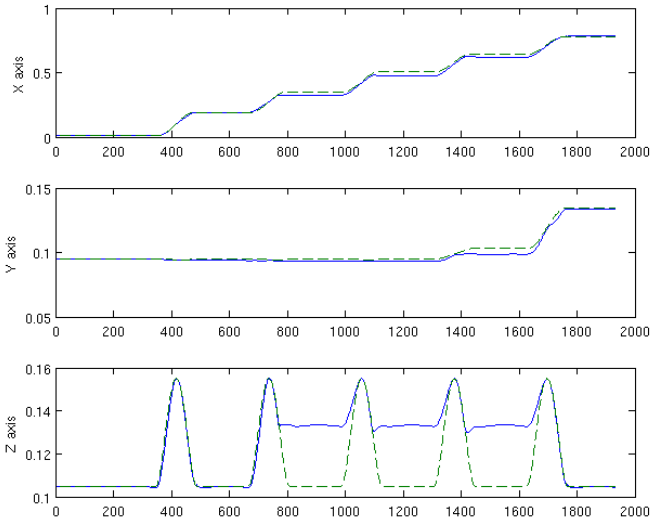
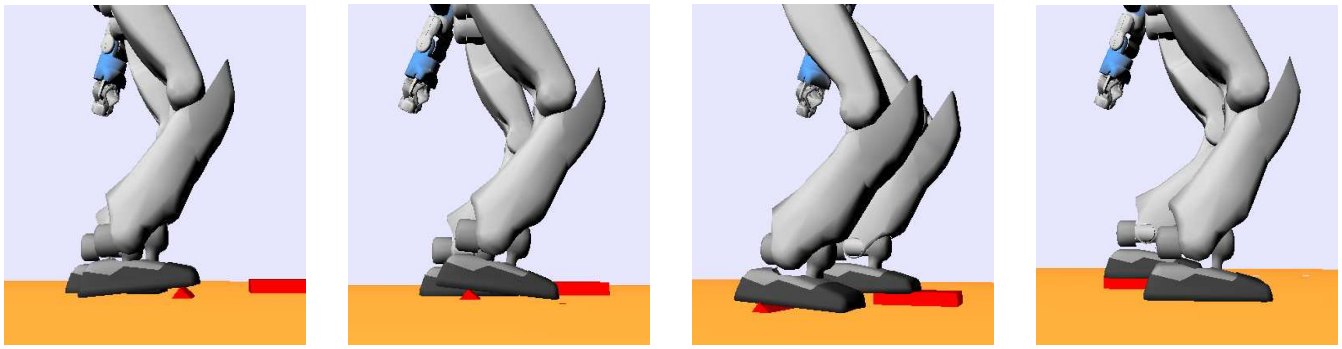


Fig. 8: Trajectory of the left foot in xyz. The dashed line shows the trajectory of the pattern generator and the solid line shows the trajectory generated by the inverse dynamics controller.

tracked, the interpolation task generates a trajectory similar to that of the pattern generator. The difference in the height z for the two first steps is due to the obstacle found by the foot, which is properly handled by the system without losing the dynamic balance. Figure 8 shows the same information for the left foot. In this case, the second, third and fourth steps found an obstacle on the ground and that is the reason why there is an observable difference in the z axis between the output of the pattern generator and the trajectory generated by the dynamic solver.



(a) (b) (c) (d)

Fig. 5: Robot walking on a non-planar surface.

VI. Conclusion

The control scheme presented in this paper makes the humanoid robot able to walk on non-planar surfaces by detecting collision points and moving the foot properly to reach a larger support polygon. This approach is grounded on a mathematical reasoning and has been tested in a simulation environment where collision detection is available. The method is limited to horizontal surfaces and would fail if the ground has a large slope, in which case, a modification in the pattern generator would be needed. The implementation of the controller on the real robot requires a strategy to efficiently detect contact points. This constitutes our current work.

References

[1] J. Yamaguchi, A. Takanishi, and I. Kato, "Development of a biped walking robot adapting to a horizontally uneven surface," in *Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems' 94*, vol. 2. IEEE, 1994, pp. 1156–1163.

[2] K. Nishiwaki and S. Kagami, "High frequency walking pattern generation based on preview control of zmp," in *IEEE International Conference on Robotics and Automation, 2006. ICRA 2006*. IEEE, 2006, pp. 2667–2672.

[3] P.-B. Wieber, "Trajectory free linear model predictive control for stable walking in the presence of strong perturbations," in *IEEE International Conference on Humanoid Robotics*, Genova, Italy, October 2006, pp. 137–142.

[4] K. Hashimoto, Y. Sugahara, M. Kawase, A. Ohta, C. Tanaka, A. Hayashi, N. Endo, T. Sawato, H. Lim, and A. Takanashi, "Landing pattern modification method with predictive attitude and compliance control to deal with uneven terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2006, pp. 1755–1760.

[5] T. Takenaka, T. Matsumoto, T. Yoshiike, T. Hasegawa, S. Shirokura, H. Kaneko, and A. Orita, "Real time motion generation and control for biped robot-4th report: Integrated balance control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'09)*. IEEE, 2009, pp. 1601–1608.

[6] K. Nishiwaki and S. Kagami, "Strategies for adjusting the zmp reference trajectory for maintaining balance in humanoid walking," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2010, pp. 4230–4236.

[7] M. Morisawa, F. Kanehiro, K. Kaneko, S. Kajita, and K. Yokoi, "Reactive biped walking control for a collision of a swinging foot on uneven terrain," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids'11)*. IEEE, 2011, pp. 768–773.

[8] L. Saab, N. Mansard, F. Keith, J. Fourquet, and P. Soueres, "Generation of Dynamic Motion for Anthropomorphic Systems under Prioritized Equality and Inequality Constraints," in *IEEE International Conference on Robotics and Automation (ICRA'11)*, Shanghai, May 2011.

[9] L. Saab, O. Ramos, N. Mansard, P. Soueres, and J.-Y. Fourquet, "Generic Dynamic Motion Generation with Multiple Unilateral Constraints," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2011)*, San Francisco, USA, September 2011.

[10] N. Mansard, "A dedicated solver for fast operational-space inverse dynamics," in *IEEE International Conference on Robotics and Automation, ICRA'12*, Minnesota, USA, May 2012.

[11] R. Featherstone, *Rigid Body Dynamics Algorithms*. Springer Berlin, 2008, vol. 49.

[12] A. Escande, N. Mansard, and P. Wieber, "Fast resolution of hierarchized inverse kinematics with inequality constraints," in *IEEE International Conference on Robotics and Automation (ICRA'10)*. IEEE, 2010, pp. 3733–3738.

[13] 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 *IEEE International Conference on Robotics and Automation (ICRA'03)*, Taipei, Taiwan, September 2003.

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

[15] M. de Lasa, I. Mordatch, and A. Hertzmann, "Feature-Based Locomotion Controllers," *ACM Transactions on Graphics*, vol. 29, no. 3, 2010.