

Whole-body Motion Integrating the Capture Point in the Operational Space Inverse Dynamics Control

Oscar E. Ramos, Nicolas Mansard, Philippe Souères

Abstract—It is important for a humanoid robot to be able to move its body without falling down even if the target motion takes its center of mass to the limits of the support polygon. Usually the center of mass is overconstrained to keep balance, but this can make fast motion of the robot upper body or tasks that are far away from the reachable space unfeasible. To achieve these tasks that challenge the robot balance, this paper proposes the integration of the capture point in the operational-space inverse dynamics control framework so that, if balance is about to be lost a good place to step on will be determined preventing the robot from falling down. Moreover, the control of the capture point as a task (or constraint) will guarantee that it does not move out exponentially, allowing the foot to have time to safely step on it before the robot falls. An advantage over other methods is the transparent integration of the capture point letting the robot be able to simultaneously move its whole body satisfying other tasks. The method has been tested in simulation using the dynamic model of the HRP-2 robot.

I. INTRODUCTION

Whole-body control of humanoid robots is a complicated task since balance has to be maintained while generating motion for the robot body, which is highly redundant, underactuated, and presents an unstable nature in a vertical posture. One approach is to use motion planning with optimal control [1], [2], but this can be computationally expensive. Another approach uses inverse kinematics introducing constraints with priority order through projections onto the null space of prior constraints [3], [4]. An extension, based on quadratic programs (QPs) and able to deal with inequality constraints, has been proposed in [5] for kinematic control, and [6] improved the resolution methodology introducing a computationally efficient solver. Inverse-dynamics control is another approach that generates dynamically feasible motion and successive projections onto nullspaces can be used for a task hierarchy [7] or a more powerful QP-based solution integrating inequalities can be formulated [8].

Balance control is also an active research topic and aims at preventing the robot from falling down. For static balance, the Center of Mass (CoM) needs to lie inside the support polygon at all times [9], and for dynamic balance, the Zero-Moment Point (ZMP) [10] needs to be inside the support polygon. Using the relation between the CoM and the ZMP, and relying on simplified models, several schemes for walking have been successfully used for humanoid robots such as walking pattern generators [11], [12], including the integration of constraints on the ZMP and footstep adaptation

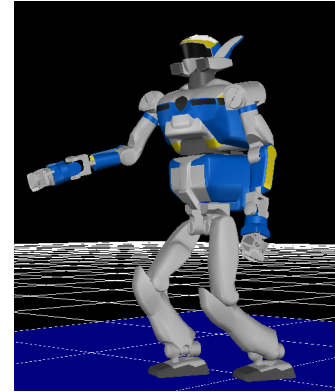


Fig. 1: HRP-2 after having stepped in order not to fall when trying to reach a far position with its right hand

[13]. But, it has been shown that a small horizontal distance between the CoM projection on the ground and the ZMP can produce a moment that destabilizes the robot [14]. Thus, these quantities cannot reveal how stable the robot is in a given configuration, and more general stability margins have been proposed [15]. As a result, the capture point (CP) [16], [14] was introduced as a criteria that can be used for balance control, and it can be easily computed when a simplified model of the robot is used.

The approach presented in this paper uses an inverse dynamics control framework based on the task-function approach for motion generation, and includes an inequality task for the CP in order to guarantee robot balance. There are situations where the robot needs to move fast or needs to move farther than its static limits (e.g. reaching an object that is relatively far, as Fig. 1 shows), and this can generate a high velocity on the CoM leading to a possible loss of balance. Introducing the CP in the control loop enables the robot to perform very fast whole-body movements without falling. That is, the robot will maintain its balance as much as possible and if the required task would lead to falling, the robot will make a step to keep its balance. It is this step that will be guided by the CP, which will be constrained to remain within the foot reachable limits.

The paper is organized as follows. Section II presents a brief summary of the inverse dynamics control model using the so called dynamic Stack of Tasks (SoT) formalism. In Section III the linear inverted pendulum model is recalled as well as the mathematical formulation of the CP and its dynamics. The CP task used to keep balance, as well as its usage within the stack of tasks is presented in Section IV.

Finally, the results obtained in simulation for the HRP-2 humanoid robot trying to move its arm far away from its reachability space are presented in Section V.

II. INVERSE DYNAMICS CONTROL

Motion generation for redundant robots such as humanoids, typically considers more than one objective at the same time. These objectives, referred to as tasks, can be handled using weighted schemes or prioritized schemes. This work, based on the task-function approach, uses a prioritized scheme that satisfies dynamic constraints generating feasible motion. This section presents a summary of the approach, and more details can be found in [8].

A. Generic Dynamic Model

Let the humanoid robot configuration be represented by the generalized coordinates $q = (q_a, x)$, where q_a represents the n actuated degrees of freedom, and x the pose of the the robot free-flyer or base. The contact points of the robot with the environment will be represented by x_c and the forces acting at these points by f_c . The dynamic model of the robot in contact with the environment is given by:

$$A\ddot{q} + b + J_c^T f_c = S^T \tau \quad (1)$$

where A is the whole-body generalized inertia matrix, b is a vector that includes the Coriolis, centrifugal and gravity effects, $J_c = \frac{\partial x_c}{\partial q}$ is the Jacobian of the contact points relating them to the robot configuration, $S^T = [0 \ I]^T$ is a matrix that selects the actuated torques, and τ are the torques applied at the joints.

Let S^n , with components $s_{ij}^n = \delta(3i - j)$ where δ represents the Kronecker delta function, be a matrix that selects only the $3n^{th}$ components of a vector. Thus, $x_c^\perp = S^n x_c$ and $f_c^\perp = S^n f_c$ are vectors containing only the normal components of the contact position and force, respectively (assuming that the normal to the planar contact surface is in the z direction). The complementarity conditions to avoid interpenetration (e.g. between the foot and the ground) and to keep the contact are $x_c^\perp = 0$ and $f_c^\perp \geq 0$, with $x_c^\perp f_c^\perp = 0$. Since by definition the contact Jacobian J_c gives the differential relation $\dot{x}_c = J_c \dot{q}$, the contact condition can also be written as is

$$\ddot{x}_c = J_c \ddot{q} + \dot{J}_c \dot{q} = 0 \quad (2)$$

and the force condition is

$$f_c^\perp \geq 0. \quad (3)$$

The unidirectionality of the perpendicular contact forces, f_c^\perp , has been shown to be equivalent to the ZMP criterion for coplanar contacts [8], implying that if (3) holds then the ZMP is guaranteed to lie inside the support polygon.

B. Inverse-Dynamics Stack of Tasks

The Stack of Tasks (SoT) formalism generates motion for the robot using a series of tasks, each one specific to a particular objective. Using the task function approach [17], a task i can be defined without loss of generality in terms of the

error between the current specification s_i and the desired one s_i^* as $e_i = s_i - s_i^*$. The specification s_i , and thus the task e_i , is generic and can represent different things like a position, an orientation, a posture, a visual feature, among others. Since the robot dynamic model considers joint accelerations, each task must be specified in terms of \ddot{q} . For the i^{th} task, the relation at the acceleration level is achieved using the task Jacobian $J_i = \frac{\partial e_i}{\partial \ddot{q}}$ as:

$$\ddot{e}_i = \dot{J}_i \dot{q} + J_i \ddot{q}. \quad (4)$$

For instance, exponential error regulation for equality tasks can be obtained using a PD control law as $\ddot{e}^* = -k_p e_i - k_v \dot{e}_i$, where k_p is the proportional gain and k_v the derivative gain. These gains are chosen in such a way that $k_p > 0$ and $k_v = 2\sqrt{k_p}$.

The inverse dynamics control finds the proper joint accelerations (\ddot{q}), joint torques (τ), and contact forces (f_c) that are feasible and satisfy the task(s) specification(s) as closely as possible within the hierarchy. Hence, the optimization variable is defined as (\ddot{q}, τ, f_c) , and the solution to the inverse dynamics problem is found using *Hierarchical Quadratic Programming* (HQP) [6]. This approach consists in a cascade of QPs that can take into account equalities and inequalities at any level of the hierarchy. To specify the priority, the lexicographic operator \prec is introduced so that $(a) \prec (b)$ translates to (a) having a higher priority than (b) . With this notation, and considering m tasks, the dynamic SoT is represented as the following HQP problem: $(1) \prec (2) \prec (3) \prec (4)_1 \prec \dots \prec (4)_m$, where $(4)_i$ is the i^{th} task represented by (4). For a faster computation, it is possible to explicitly decouple the optimization space (\ddot{q}, τ, f_c) into two components: motion and actuation, which have a lower dimension than the original variables [18]. Then, without loss of generality, all the elements that constitute the SoT can be expressed in terms of the decoupled variables.

III. CAPTURE POINT

A. Linear Inverted Pendulum Dynamics

The approach followed here will use the 3D linear inverted pendulum (LIP) [19] as an approximation to the dynamics of a humanoid robot. The assumptions of this model are the following:

- The robot is represented by a punctual mass m located at its CoM.
- The robot legs are massless and the extreme in contact with the ground can be freely moved (it is not externally actuated).
- The height of the CoM is kept constant throughout the motion.

Consider the 3D LIP model shown in Fig. 2. Let the CoM be represented by $r_c = (x_c, y_c, z_c)$, the point of the pendulum in contact with the ground by $r_z = (x_z, y_z, 0)$, and the force that acts along the pendulum by $\mathcal{F} = (f_x, f_y, f_z)$. For a robot with feet, r_z in the model is equivalent to the robot ZMP. With this notation, the equations of motion for

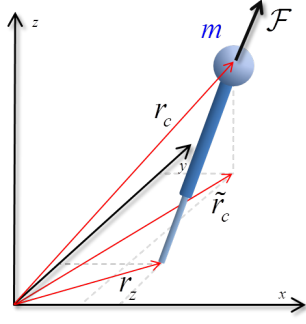


Fig. 2: 3D Linear Inverted Pendulum (LIP)

the system are:

$$m\ddot{r}_c = \mathcal{F} + m\mathcal{G} \quad (5)$$

$$(r_c - r_z) \times \mathcal{F} = 0 \quad (6)$$

where $\mathcal{G} = (0, 0, -g)$ is the gravity vector. Since one of the assumptions of the model implies a constant height for the CoM ($\dot{z}_c = 0$), the CoM acceleration will be given by $\ddot{r}_c = (\ddot{x}_c, \ddot{y}_c, 0)$. With this consideration, the last component of (5) establishes a vertical equilibrium as $f_z = mg$. Using this equivalence of f_z to solve for the horizontal force components f_x, f_y in (6), and replacing the expression of the force \mathcal{F} in (5) leads to the dynamics of the CoM:

$$\ddot{\tilde{r}}_c = \omega^2(\tilde{r}_c - \tilde{r}_z) \quad (7)$$

where $\tilde{r}_c = (x_c, y_c)$ and $\tilde{r}_z = (x_z, y_z)$ are the horizontal components of the CoM and the ZMP, respectively, and $\omega = \sqrt{\frac{g}{z_c}}$ is the eigenfrequency of the pendulum.

The solution to the differential equation representing the LIP dynamics (7) leads to the temporal evolution of the CoM horizontal components as:

$$\tilde{r}_c(t) = \frac{1}{2} \left(\tilde{r}_{c_0} - \tilde{r}_z + \frac{\dot{\tilde{r}}_{c_0}}{\omega} \right) e^{\omega t} + \frac{1}{2} \left(\tilde{r}_{c_0} - \tilde{r}_z - \frac{\dot{\tilde{r}}_{c_0}}{\omega} \right) e^{-\omega t} + \tilde{r}_z \quad (8)$$

where $\tilde{r}_{c_0} = \tilde{r}_c(0)$ and $\dot{\tilde{r}}_{c_0} = \dot{\tilde{r}}_c(0)$ are the initial CoM position and velocity. It can be seen that in general the trajectory for the CoM diverges due to the exponential term.

B. Capture Point Derivation and Dynamics

The Capture Point (CP), introduced in [15] and also referred to as the Extrapolated Center of Mass [20], is the point $\xi = (\xi_x, \xi_y)$ on the ground where the robot should step on to be able to come to a complete rest. Modeling the robot as a 3D LIP, its contact with the ground \tilde{r}_z is not fixed but can move. Moreover, with a robot that has feet, the point \tilde{r}_z corresponding to the ZMP can move inside the polygon defined by the supporting foot. If the CoM is moving, the only way for it to stop is by achieving a constant value for $\tilde{r}_c(t)$ as time approaches infinity. In (8), as $t \rightarrow \infty$, there is a divergence of $\tilde{r}_c(t)$ since $e^{\omega t} \rightarrow \infty$. Then, the condition

to avoid divergence is that the coefficient of the exponential be null, that is:

$$\tilde{r}_{c_0} + \frac{\dot{\tilde{r}}_{c_0}}{\omega} = \tilde{r}_z. \quad (9)$$

Using (9), it can be easily shown that the limits for the CoM position and velocity as time approaches infinity are $\lim_{t \rightarrow \infty} \{\tilde{r}_c(t)\} = \tilde{r}_z = \xi$ and $\lim_{t \rightarrow \infty} \{\dot{\tilde{r}}_c(t)\} = 0$ where ξ is called the Capture Point since it is the only point where the CoM can come to a rest. Considering this equivalence for ξ , the condition (9) for a general position of the CoM gives the expression for the instantaneous CP as:

$$\xi = \tilde{r}_c + \frac{\dot{\tilde{r}}_c}{\omega} \quad (10)$$

It is also possible to find the expression for the instantaneous CP by analyzing the orbital energy of the pendulum [16].

The CP velocity can be obtained by differentiating (10) and replacing both (7) and the expression for the velocity of the CoM from (10), and is given by

$$\dot{\xi} = \omega(\xi - \tilde{r}_z) \quad (11)$$

which represents the first order dynamics of the CP. Solving (11), the explicit formulation of the instantaneous CP trajectory as a function of time is

$$\xi(t) = (\xi_0 - \tilde{r}_z)e^{\omega t} + \tilde{r}_z \quad (12)$$

where ξ_0 is the CP ξ at the initial time. The computation of (12) allows the prediction of the future CP position.

IV. INTEGRATION OF THE CAPTURE POINT IN THE STACK OF TASKS

A. Capture Point and the Control of the Center of Mass

For a standing phase where both feet are on the ground and the rest of the body is moving with arbitrary motion, there are typically two ways to control the CoM. The first one is to fix it to a certain horizontal position, for instance the center of the support polygon, imposing an equality constraint. Although this approach ensures balance, it greatly affects the whole motion since it overconstrains the robot body and can lead feasible tasks to become unachievable. The other way is to let the CoM lie anywhere inside the support polygon imposing an inequality constraint. This control is less restrictive and allows for more variety of motion, but it presents the potential inconvenience that if the CoM reaches the border of the support polygon with a considerable high velocity, balance can be lost as the CoM will irreversibly leave the polygon. Thus, the CoM velocity is important and if not properly considered, the border of the support polygon can become a dangerous zone that can easily lead to a loss of balance. A naive way to overcome this difficulty is by restricting the CoM inside a polygon that is itself inside the real support polygon, giving some ‘‘security margins’’. However, the problem is again the imposition of unnecessary constraints to the motion, and the proper choice of those margins is not evident.

The problem with the CoM arises from the fact that inverse dynamics control considers only a linearization of the current

system dynamics, and thus there is very little it can do by itself to avoid overshoots due to large CoM velocities and accelerations. The control law cannot predict future states. To overcome these limitations, optimal control can be used, but it is currently computationally very expensive. The approach in this paper rather proposes to preview the CoM future through the Capture Point, and therefore to constrain the CoM velocity at the limit using the CP as a measurement of its future, since both quantities are related, as (10) shows. While the CP remains inside the support polygon, the robot is able to freely move its whole-body with the consideration that there will always exist the possibility to come to rest. That is, the robot keeps and will keep its balance. If the CP leaves the support polygon while performing some fast motion, it will not be possible for the robot to come to rest in double support and, unless a step is taken, balance will be lost. Then, a task to impose a constraint on the CP can be a good solution.

B. Capture Point Task

The CP task is proposed as an inequality task, or constraint, that aims at keeping the CP inside the support polygon at all times. Moreover, this task can be used with a “larger” polygon, in which case, a step will be required, but the CP position will remain bounded making the step feasible. Let the lower and upper bounds of a polygon (the support polygon or some larger polygon) be given by \underline{r}_p and \bar{r}_p , respectively. The CP task constrains the CP to lie within the limits of this polygon as $\underline{r}_p \leq \xi \leq \bar{r}_p$, or equivalently:

$$\underline{r}_p \leq \tilde{r}_c + \frac{\dot{\tilde{r}}_c}{\omega} \leq \bar{r}_p. \quad (13)$$

From (13) it is evident that the CP can be bounded by indirectly controlling the CoM position and velocity. Then, the task to effectively control the CP can be defined in terms of the CoM. Since tasks are integrated in a dynamic control scheme, the CP task needs to be formulated using the acceleration as in (4); hence, $\ddot{\tilde{r}}_c$ and $\dot{\tilde{r}}_c$ in (13) need to be related to the CoM acceleration. Using a sampling time Δt , and a second order approximation around the point \tilde{r}_{c_i} , the position of the CoM can be expressed as

$$\tilde{r}_c \approx \tilde{r}_{c_i} + \dot{\tilde{r}}_{c_i} \Delta t + \ddot{\tilde{r}}_{c_i} \frac{\Delta t^2}{2} \quad (14)$$

and its velocity as

$$\dot{\tilde{r}}_c \approx \dot{\tilde{r}}_{c_i} + \ddot{\tilde{r}}_{c_i} \Delta t. \quad (15)$$

Replacing (14) and (15) in (13), and letting the task reference behavior be $\ddot{e}_{cp}^* = \ddot{\tilde{r}}_c$, the Capture Point task respecting the limits within the polygon is expressed as a function of the CoM as:

$$k_a \ddot{\underline{r}}_c \leq \ddot{e}_{cp}^* \leq k_a \ddot{\bar{r}}_c \quad (16)$$

where k_a is a constant given by

$$k_a = \frac{2\omega}{(\omega\Delta t + 2)\Delta t}$$

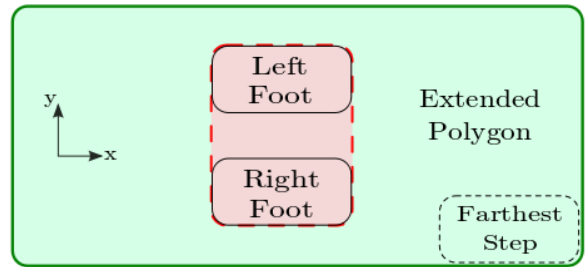


Fig. 3: The red polygon defines the real support polygon in the double support phase. The green polygon defines an ‘extended’ polygon used to limit the position of the capture point. This polygon is defined so that the farthest feasible step lies within it

and the CoM acceleration limits are

$$\begin{aligned} \ddot{\underline{r}}_c &= \underline{r}_p - \tilde{r}_{c_i} - \dot{\tilde{r}}_{c_i} \left(\Delta t + \frac{1}{\omega} \right) \\ \ddot{\bar{r}}_c &= \bar{r}_p - \tilde{r}_{c_i} - \dot{\tilde{r}}_{c_i} \left(\Delta t + \frac{1}{\omega} \right) \end{aligned}$$

The CP task given by (16) can be directly included in the inverse dynamics SoT. The fact of controlling the CP as a task implies that it will not escape the pre-defined polygon implicitly constraining the whole motion of the robot. The implicit constraints appear as motion of parts of the robot for which there is no explicit task (for instance, an arm or the chest) in an attempt to compensate the otherwise fast falling motion.

C. Scheme for the Capture Point Control

The first use of the CP task is to directly keep the CP inside the real support polygon. This is done by defining the limits of the task \underline{r}_p and \bar{r}_p in terms of the support polygon. The task (16) is then added to the inverse dynamics SoT (Section II-B) as an inequality task with a priority higher than the rest of the tasks, so that it is satisfied in all the cases. This leads to a more restrictive control than simply constraining the CoM within the support polygon, but it has the advantage that the CoM velocity is also controlled. This implies that the CoM at the borders of the polygon will not present high velocities that make it irreversibly exit the polygon. The reason is that by definition, if the CP is always kept inside the polygon, the CoM will be able to come to a rest within it in finite time. In other terms, the robot will be able to move without falling.

However, there are situations where the fulfillment of a task might require to move the CoM away from the support polygon (for example, if the robot has to reach an object that is farther than roughly the length of its arm), which would cause a loss of balance. For these cases, the strict balance condition can be relaxed by defining a polygon outside the limits of the real support polygon, but within an area that is reachable by a step, as Fig. 3 shows. Then, this ‘extended’ polygon is used to limit the position of the CP. Provided that double support balance is lost, a step towards the CP will

have to be taken to recover balance; otherwise the robot will fall. This scheme is summarized as follows:

- *Definition of the CP task.* An ‘extended’ polygon covering the area that is reachable by the robot foot is defined as limit for the CP task. This constrains the CP to always lie within the polygon and avoids the problem of its exponential increase, which would prevent the foot to step over it. Then, other tasks are added to the inverse dynamics SoT to generate whole-body motion.
- *Need for a step detection.* If at some moment of the motion the CP exits the real support polygon, the robot will not be able to recover its balance if it keeps its current support polygon. The support needs to be extended to cover the CP position. To this end, a step must be executed.
- *Beginning the step.* To perform a step, the foot that is closer to the current CP position is selected. As soon as the CP leaves the real support polygon, the foot leaves the ground towards an intermediate position. This position is horizontally located at the current position of the CP, and the step height is pre-defined (assuming a flat horizontal ground). The horizontal position is chosen to be the current position of the CP since the final position of the CP is unknown and the motion needs to be fast (defining a position in the midway between the current foot position and the CP position reduces the capability of reaching the CP at the final stage). The desired intermediate position will change continuously as the CP moves farther. Because of this change, only the step height is considered as criteria to finish this stage: as soon as the desired step height is achieved, the foot will move towards its final position.
- *Ending the step.* After the foot completed the intermediate position, its final position is defined as the current CP position. Since the CP is still moving, the task objective will also be time-varying. However, the CP task defined at the beginning of the scheme is controlling the CP to remain inside a greater polygon preventing it from moving to unreachable positions. If no control was applied on the CP, it would exponentially move away and the foot would never be able to meet it, causing the robot to fall down on the attempt to step over the CP; thereof the importance of the CP task. When the foot reaches the CP on the ground, it again enters a double support phase which will now contain the current capture point. Then, the CoM is ensured to be able to come to a full stop.

V. SIMULATION RESULTS

The described framework was implemented and tested using the dynamic model of the HRP-2 robot. The desired objective was for the right hand to reach the position specified by the ball in front of the robot (Fig. 4a). It can be foreseen that when the robot tries to reach that position with the right hand, there are two possibilities for the CoM: (i) if the CoM is controlled to lie inside the support polygon, the robot will not achieve the goal keeping double support

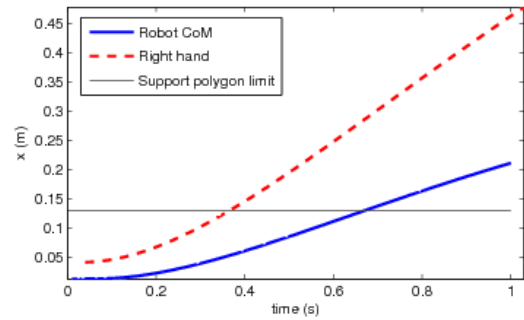


Fig. 5: Time evolution in the forward direction (x) when there is no control of the CoM or CP

since the target is far away from the reachable limits; (ii) if the CoM is not controlled, the robot will fall down while trying to reach that point, unless it performs a step. The latter case without an additional step is shown in Fig. 5, where the right hand task drives the CoM far away from the limits of the support polygon making the robot fall. This scenario describes a typical application of the proposed approach. To overcome the problem, an extended support polygon was defined as in Fig. 3 covering the space that is reachable by the foot. Then, the CP task was added to the SoT having this polygon as limit.

Following the control scheme of section IV-C it is observed in Fig. 4 that the robot starts moving the arm towards the ball. However, since the target is far, this implies an initial relatively fast motion of the arm, which generates a dangerous velocity for the CoM. This is detected by the capture point leaving the real support polygon as Fig. 6 shows (at $t = 0.27$ s), which acts as a preview control for the CoM. It is in this moment that the right foot leaves the ground and moves towards the CP, which is constrained not to exponentially increase but to lie within some bounded region, as verified by Fig. 6 (at $t = 1$ s the CP is bounded). When the foot finally reaches the CP, the robot enters a new double support phase, which now contains the CoM. After the foot reaches the ground (at $t = 1.10$ s), the CoM still presents a forward motion, and therefore continues moving forwards, but eventually it is able to come to rest since the CP is now contained inside the new support polygon. At the end of the motion, the right hand achieves the far target (the ball, in this case), and both the CoM and the CP converge to the same position provided that the CoM velocity is null. Fig. 6 also shows that it is the usage of the CP that acts as a predictor for the CoM motion and allows the foot to start the step as a reaction to the velocity of the CoM, before it has escaped the support polygon and before a recovery is too late.

It should be noted that the right hand continues its motion towards the ball at all times. By the end of the motion, Fig. 4 shows a natural movement of the left hand backwards to compensate for the motion of the right hand, even though there is no specific task controlling the left arm. This is a consequence of the inverse dynamics SoT control and

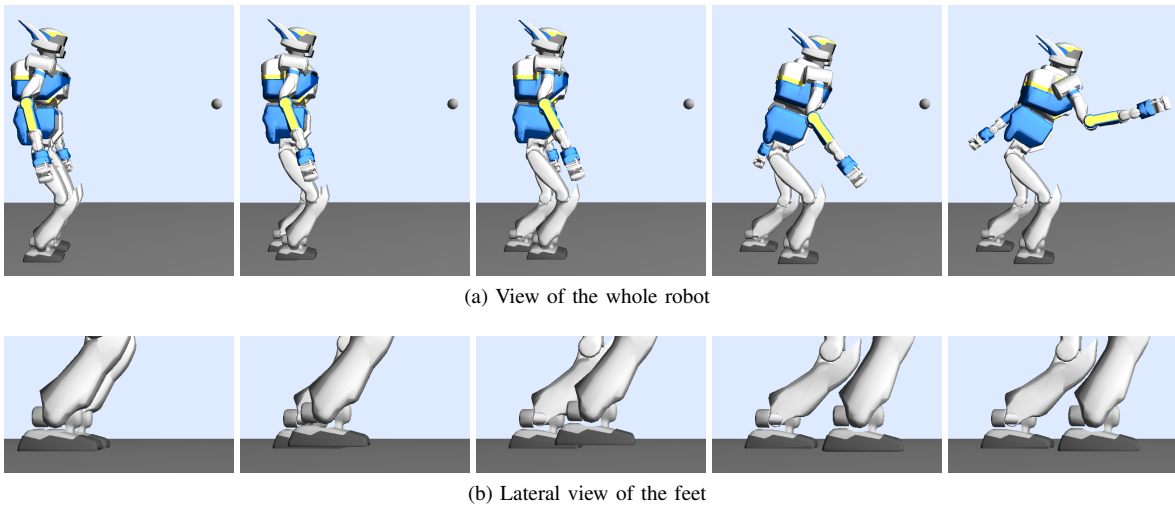


Fig. 4: Snapshots for the robot trying to reach an object (ball) which is far from the arm reachable space unless a step is performed. The robot starts in double support, then as the right arm moves towards the ball the right foot automatically starts a step to follow the CP. After the step finishes, the robot continues its motion to reach the target with the right hand.

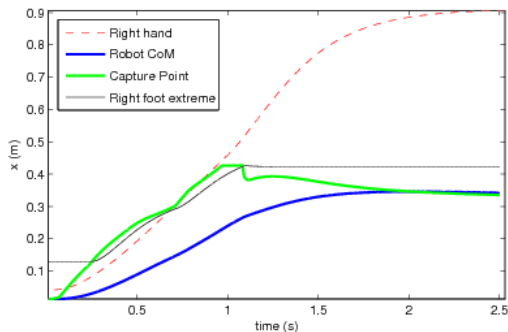


Fig. 6: Time evolution in the forward direction ('x') when the proposed control approach is used. The black line shows the frontal extreme of the right foot which determines the frontal limit of the support polygon when the foot is on the ground (as at the beginning and end of the motion)

resembles the way humans move (the effect of the control resembling human motion was previously noted in [21]). It is also important to point out that the step parameters (when to start the step and where to step to) have not been previously precomputed but have resulted as a natural consequence of the control framework in response to the loss of balance.

VI. CONCLUSIONS

The methodology proposed in this paper has shown the feasibility of incorporating the capture point inside the inverse dynamics control scheme in order to perform tasks that would otherwise make the robot fall down. Stepping on the CP is a good solution but presents the inconvenience that the CP can exponentially escape the reachable zone for the foot. The control of the CP guarantees that it remains inside the specified extended polygon by implicitly constraining the whole body motion within limits that will allow the robot to attain balance after a step. One of the advantages of

integrating the CP in the control scheme is the ability to determine the moment when the robot will lose its balance, and thus, when a step has to be started and where to step. Moreover, besides balance control, the whole-body is moved with an arbitrary task generating some useful motion. The extension to more steps follows the same approach; however, in this case making the robot walk towards the goal would be more efficient.

ACKNOWLEDGMENTS

This work was supported by the OSEO ROMEO-2 Project, and the FP7-ICT-2013-10/611909 KOROIBOT Project.

REFERENCES

- [1] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue, "Motion planning for humanoid robots," in *Robotics Research*, 2005, pp. 365–374.
- [2] A. El-Khoury, "Planning optimal motions for anthropomorphic systems," Ph.D. dissertation, LAAS-CNRS, Université de Toulouse (UPS), June 2013.
- [3] Y. Nakamura, *Advanced robotics: redundancy and optimization*. Addison-Wesley Longman Publishing Co., Inc., 1990.
- [4] B. Siciliano and J.-J. Slotine, "A general framework for managing multiple tasks in highly redundant robotic systems," in *Int. Conf. on Advanced Robotics (ICAR)*, 1991.
- [5] O. Kanoun, F. Lamiroux, P.-B. Wieber, F. Kanehiro, E. Yoshida, and J.-P. Laumond, "Prioritizing linear equality and inequality systems: application to local motion planning for redundant robots," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Kobe, Japan, 2009.
- [6] A. Escande, N. Mansard, and P.-B. Wieber, "Hierarchical quadratic programming," *The International Journal of Robotics Research (IJRR)*, March 2014, [in press].
- [7] L. Sentis and O. Khatib, "Control of free-floating humanoid robots through task prioritization," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Barcelona, Spain, 2005.
- [8] L. Saab, O. Ramos, N. Mansard, P. Souères, and J.-Y. Fourquet, "Dynamic whole-body motion generation under rigid contacts and other unilateral constraints," *IEEE Transactions on Robotics (TRO)*, vol. 29, no. 2, pp. 346–362, April 2013.
- [9] P.-B. Wieber, "On the stability of walking systems," in *International workshop on humanoid and human friendly robotics*, 2002.
- [10] M. Vukobratović and J. Stepanenko, "On the stability of anthropomorphic systems," *Mathematical Biosciences*, vol. 15, no. 1, pp. 1–37, 1972.

- [11] 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 Int. Conf. on Robotics and Automation (ICRA)*, vol. 2, 2003.
- [12] T. Sugihara, "Standing stabilizability and stepping maneuver in planar bipedalism based on the best com-zmp regulator," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2009.
- [13] P.-B. Wieber, "Trajectory free linear model predictive control for stable walking in the presence of strong perturbations," in *IEEE/RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2006.
- [14] A. L. Hof, "The 'extrapolated center of mass' concept suggests a simple control of balance in walking," *Human movement science*, vol. 27, no. 1, pp. 112–125, 2008.
- [15] J. E. Pratt and R. Tedrake, "Velocity-based stability margins for fast bipedal walking," in *Fast Motions in Biomechanics and Robotics*. Springer, 2006, pp. 299–324.
- [16] J. Pratt, J. Carff, S. Drakunov, and A. Goswami, "Capture point: A step toward humanoid push recovery," in *IEEE/RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2006.
- [17] C. Samson, B. Espiau, and M. Le Borgne, *Robot control: the task function approach*. Oxford University Press, 1991.
- [18] N. Mansard, "A dedicated solver for fast operational-space inverse dynamics," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, St Paul, MN, USA, 2012.
- [19] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3d linear inverted pendulum mode: A simple modeling for a biped walking pattern generation," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, vol. 1, 2001.
- [20] A. Hof, M. Gazendam, and W. Sinke, "The condition for dynamic stability," *Journal of biomechanics*, vol. 38, no. 1, pp. 1–8, 2005.
- [21] O. Ramos, N. Mansard, O. Stasse, and P. Soueres, "An advanced robotics motion generation framework for inferring the organization of human movements," *Computer Methods in Biomechanical Engineering (CMBEE)*, vol. 16, no. 1, September 2013.