

Balancing experiments on a torque-controlled humanoid with hierarchical inverse dynamics

Alexander Herzog¹, Ludovic Righetti^{1,2}, Felix Grimmering¹, Peter Pastor² and Stefan Schaal^{1,2}

Abstract—Recently several hierarchical inverse dynamics controllers based on cascades of quadratic programs have been proposed for application on torque controlled robots. They have important theoretical benefits but have never been implemented on a torque controlled robot where model inaccuracies and real-time computation requirements can be problematic. In this contribution we present an experimental evaluation of these algorithms in the context of balance control for a humanoid robot. The presented experiments demonstrate the applicability of the approach under real robot conditions (i.e. model uncertainty, estimation errors, etc). We propose a simplification of the optimization problem that allows us to decrease computation time enough to implement it in a fast torque control loop. We implement a momentum-based balance controller which shows robust performance in face of unknown disturbances, even when the robot is standing on only one foot. In a second experiment, a tracking task is evaluated to demonstrate the performance of the controller with more complicated hierarchies. Our results show that hierarchical inverse dynamics controllers can be used for feedback control of humanoid robots and that momentum-based balance control can be efficiently implemented on a real robot.

I. INTRODUCTION

We expect autonomous legged robots to perform complex tasks in persistent interaction with an uncertain and changing environment (e.g. in a disaster relief scenario). Therefore, we need to design algorithms that can generate precise but compliant motions while optimizing the interactions with the environment. In this context, the choice of a control strategy for legged robots is of primary importance as it can drastically improve performance in face of unexpected disturbances and therefore open the way for agile robots, whether they are locomoting or performing manipulation tasks.

Robots with torque control capabilities [1], [2], including humanoids [3], [4], are becoming increasingly available and torque control algorithms are therefore necessary to fully exploit their capabilities. Indeed, such algorithms often offer high performance for motion control while guaranteeing a certain level of compliance [2], [5], [6], [7]. In addition, they also allow for the direct control of contact interactions with the environment [1], [8], [9], which is required during operation in dynamic and uncertain environments. Recent contributions have demonstrated the relevance of torque control approaches for humanoid robots, for example for balancing capabilities [10], [11], [4], [12]. All these

approaches try to regulate the position of the center of mass (CoM) of the robot to ensure that the robot does not fall while guaranteeing that contact forces are physically admissible. We can essentially distinguish two approaches.

Passivity-based approaches on humanoids were originally proposed in [10] and recently extended in [4]. They compute admissible contact forces and control commands under quasi-static assumptions. The great advantage of such approaches is that they do not require a precise dynamic model of the robot. Moreover, robustness is generically guaranteed due to the passivity property of the controllers. However, the quasi-static assumption can be a limitation for dynamic motions.

On the other hand, controllers based on the *full dynamic model* of the robot have also been successfully implemented on legged robots [1], [9], [12]. These methods essentially perform a form of inverse dynamics. The advantage of such approaches is that they are in theory well suited for very dynamic motions. However, the need for a precise dynamic model, sensor noise (particularly in the velocities) and limited torque bandwidth makes them more challenging to implement. Moreover, it is generally required to simplify the optimization process to meet time requirements of fast control loops (typically 1 kHz on modern torque controlled robots). Practical evaluations of both approaches are still rare due to the lack of torque controlled humanoid platforms and the complexity in conducting such robot experiments.

Recent approaches for controlling hierarchies of tasks using the full dynamics of the robot are also very promising [13], [6], [14]. Their problem formulations are more general than approaches based on pseudo-inverses as they naturally allow the inclusion of arbitrary types of tasks including inequalities. The resulting optimization problems are phrased as cascades of quadratic programs (QPs). Evaluation of their applicability was done in simulation and it has been argued that these algorithms can be implemented fast enough and that they can work on robots with model-uncertainty, sensor noise and limited torque bandwidth. But to the best of our knowledge, these controllers have never been used as feedback-controllers on real torque controlled humanoids. In [14], the trajectories computed in simulation are replayed on a real robot using joint space position control but the method is not used for feedback control on the robot. It is worth mentioning that [1] recently successfully implemented a controller using the full dynamics of the robot and task hierarchies on a torque controlled quadruped robot. The approach is based on pseudo-inverses and not QPs which makes it potentially inefficient to handle inequalities.

In this contribution we evaluate the balancing performance of a humanoid robot running hierarchical inverse dynamics controllers phrased as cascades of QPs [13]. First we propose

This research was supported in part by National Science Foundation grants ECS-0326095, IIS-0535282, IIS-1017134, CNS-0619937, IIS-0917318, CBET-0922784, EECS-0926052, CNS-0960061, the DARPA program on Autonomous Robotic Manipulation, the Army Research Office, the Okawa Foundation, the ATR Computational Neuroscience Laboratories, and the Max-Planck-Society.

¹Autonomous Motion Department, Max Planck Institute for Intelligent Systems, Tübingen, Germany. first.lastname@tuebingen.mpg.de

²CLMC Lab, University of Southern California, Los Angeles, USA.

a simplification of the dynamic constraints that allow us to generally reduce the computational complexity of algorithms using inverse dynamics. It allows us to implement our controller in a 1 kHz real-time control loop. Our main focus is then on presenting various balancing experiments in order to demonstrate the applicability of the approach under real robot conditions, i.e. model uncertainty, sensor noise, state estimation errors and a limited control bandwidth. In one experiment, we implement a momentum-based balance controller as originally proposed by [15] and further developed in [11] to take into account dynamic constraints. This experiment demonstrates the capabilities of such momentum-based balance controllers on a torque controlled robot. The second experiment is a tracking task, demonstrating that tracking accuracy in operational space can still be achieved. It is, to the best of our knowledge, the first demonstration of the applicability of the methods proposed in [13] or [6] as feedback controllers on torque controlled humanoids (i.e. without joint space PD control).

II. HIERARCHICAL INVERSE DYNAMICS

For our experiments we write hierarchical inverse dynamics controllers and implement the solver proposed by [13] to perform real-time feedback control. In the following we give a short summary on how tasks can be formulated and revisit the original solver formulation. In Sect. II-C we then propose a simplification to reduce the complexity of the original formulation. The simplification is also applicable to any other inverse dynamics formulations.

A. Modelling Assumptions and Problem Formulation

Assuming rigid-body dynamics, we can write the equations of motion of a robot as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{N}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{S}^T \boldsymbol{\tau} + \mathbf{J}_c^T \boldsymbol{\lambda} \quad (1)$$

where $\mathbf{q} = [\mathbf{q}_j^T \mathbf{x}^T]^T$ denotes the configuration of the robot. $\mathbf{q}_j \in \mathbb{R}^n$ is the vector of joint positions and $\mathbf{x} \in \text{SE}(3)$ denotes the position and orientation of a frame fixed to the robot with respect to an inertial frame (the floating base). $\mathbf{M}(\mathbf{q})$ is the inertia matrix, $\mathbf{N}(\mathbf{q}, \dot{\mathbf{q}})$ is the vector of all forces (Coriolis, centrifugal, gravity, friction, etc...), $\mathbf{S} = [\mathbf{I}_{n \times n} \mathbf{0}]$ represents the underactuation, $\boldsymbol{\tau}$ is the commanded joint torques, \mathbf{J}_c is the Jacobian of the contact constraints and $\boldsymbol{\lambda}$ are the generalized contact forces.

Endeffectors in contact need to stay stationary. We express the constraint that the feet (or hands) in contact with the environment do not move ($\mathbf{x}_c = \text{const}$) by differentiating it twice and using the fact that $\dot{\mathbf{x}}_c = \mathbf{J}_c \dot{\mathbf{q}}$. We get the following equality constraint

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

For the dynamics to be consistent, the centers of pressure (CoPs) at the endeffectors need to reside in the interior of each endeffector support polygon. This can be expressed as a linear inequality by expressing the ground reaction force at the zero moment point. For the feet not to slip we constraint the ground reaction forces (GRFs) to stay inside the friction cones. In our case, we approximate the cones by pyramids to have linear inequality constraints. Especially important for

generating control commands that are valid on a robot, is to take into account actuation limits $\tau_{min} \leq \tau \leq \tau_{max}$. The same is true for joint limits, which can be written as $\ddot{\mathbf{q}}_{min} \leq \ddot{\mathbf{q}} \leq \ddot{\mathbf{q}}_{max}$, where $\ddot{\mathbf{q}}_{min}, \ddot{\mathbf{q}}_{max}$ are computed from the distance of \mathbf{q} to the joint limits at each instant of time.

Motion controllers can be phrased as $\ddot{\mathbf{x}}_{ref} = \mathbf{J}_x \ddot{\mathbf{q}} + \dot{\mathbf{J}}_x \dot{\mathbf{q}}$, where \mathbf{J}_x is the task Jacobian and $\ddot{\mathbf{x}}_{ref}$ is a reference task acceleration (e.g. obtained from a PD-controller). Desired contact forces can be directly expressed as equalities on $\boldsymbol{\lambda}$. In general, we assume that each control objective can be expressed as a linear combination of $\ddot{\mathbf{q}}, \boldsymbol{\lambda}$ and $\boldsymbol{\tau}$.

At every control cycle, the equations of motion, Eq. (1), the constraints for physical consistency (torque saturation, CoP constraints, etc...) and our control objectives will all be expressed as affine equations of the variables $\ddot{\mathbf{q}}, \boldsymbol{\lambda}, \boldsymbol{\tau}$. Tasks of the same priority can then be stacked vertically into the form

$$\mathbf{A}\mathbf{y} + \mathbf{a} \leq 0, \quad (3)$$

$$\mathbf{B}\mathbf{y} + \mathbf{b} = 0, \quad (4)$$

where $\mathbf{y} = [\ddot{\mathbf{q}}^T \boldsymbol{\lambda}^T \boldsymbol{\tau}^T]^T$, $\mathbf{A} \in \mathbb{R}^{m \times (2n+6+6c)}$, $\mathbf{a} \in \mathbb{R}^m$, $\mathbf{B} \in \mathbb{R}^{k \times (2n+6+6c)}$, $\mathbf{b} \in \mathbb{R}^k$ and $m, k \in \mathbb{N}$ the overall task dimensions and $n \in \mathbb{N}$ the number of robot DoFs. $c \in \mathbb{N}$ is the number of constrained endeffectors.

The goal of the controller is to satisfy these objectives as well as possible. Objectives will be stacked into different priorities, with the highest priority in the hierarchy given to physical consistency. In a lower priority we will express balancing and motion tracking tasks and we will put tasks for redundancy resolution in the lowest priorities.

B. Hierarchical Inverse Dynamics Solver

The control objectives and constraints in Eqs. (3),(4) might not have a common solution, but need to be traded off against each other. In case of a push, for instance, the objective to decelerate the CoG might conflict with a swing foot task. A tradeoff can be expressed in form of slacks on the expressions in Eqs. (3),(4). The slacks then are minimized in a quadratic program

$$\min_{\mathbf{y}, \mathbf{v}, \mathbf{w}} \|\mathbf{v}\|^2 + \|\mathbf{w}\|^2 \quad (5)$$

$$\text{s.t.} \quad \mathbf{V}(\mathbf{A}\mathbf{y} + \mathbf{a}) \leq \mathbf{v}, \quad (6)$$

$$\mathbf{W}(\mathbf{B}\mathbf{y} + \mathbf{b}) = \mathbf{w}, \quad (7)$$

where matrices $\mathbf{V} \in \mathbb{R}^{m \times m}$, $\mathbf{W} \in \mathbb{R}^{k \times k}$ weigh the cost of constraints against each other and $\mathbf{v} \in \mathbb{R}^m$, $\mathbf{w} \in \mathbb{R}^k$ are slack variables. Note that \mathbf{v}, \mathbf{w} are not predefined, but part of the optimization variables. In the remainder we write the weighted tasks as $\tilde{\mathbf{A}} = \mathbf{V}\mathbf{A}$, $\tilde{\mathbf{a}} = \mathbf{V}\mathbf{a}$, $\tilde{\mathbf{B}} = \mathbf{W}\mathbf{B}$, $\tilde{\mathbf{b}} = \mathbf{W}\mathbf{b}$. Although, \mathbf{W}, \mathbf{V} allow us to trade-off control objectives against each other, strict prioritization cannot be guaranteed with the formulation in Eq. (5). For instance, we might want to trade off tracking performance of tasks against each other, but we do not want to sacrifice physical consistency of a solution at any cost. In order to guarantee prioritization, we solve a sequence of QPs, where a QP with lower priority tasks is optimized over the set of optimal solutions of higher priority tasks as suggested by [13]. Given one solution

$(\mathbf{y}_r^*, \mathbf{v}_r^*)$ for the QP of priority r , all remaining optimal solutions \mathbf{y} in that QP are expressed by the equations

$$\mathbf{y} = \mathbf{y}_r^* + \mathbf{Z}_r \mathbf{u}_{r+1}, \quad (8)$$

$$\bar{\mathbf{A}}_r \mathbf{y} + \bar{\mathbf{a}}_r \leq \mathbf{v}_r^*, \quad (9)$$

...

$$\bar{\mathbf{A}}_1 \mathbf{y} + \bar{\mathbf{a}}_1 \leq \mathbf{v}_1^*,$$

where $\mathbf{Z}_r \in \mathbb{R}^{(2n+6+6c) \times z_r}$ represents a surjective mapping into the nullspace of all previous equalities $\bar{\mathbf{B}}_r, \dots, \bar{\mathbf{B}}_1$ and $\mathbf{u}_r \in \mathbb{R}^{z_r}$ is a variable that parameterizes that nullspace. In order to compute \mathbf{Z}_r a Singular Value Decomposition (SVD) can be performed. In our implementation it is performed in parallel with the QP at priority level $r-1$ and rarely finishes after the QP, such that it adds only a negligible overhead. Now, we can express a QP of the next lower priority level $r+1$ and additionally impose the constraints in Eqs. (8), (9), s.t. we can optimize over \mathbf{y} without violating optimality of higher priority QPs:

$$\min_{\mathbf{u}_{r+1}, \mathbf{v}_{r+1}} \|\mathbf{v}_{r+1}\| + \|\bar{\mathbf{B}}_{r+1}(\mathbf{y}_r^* + \mathbf{Z}_r \mathbf{u}_{r+1}) + \bar{\mathbf{b}}_{r+1}\| \quad (10)$$

$$\text{s.t.} \quad \bar{\mathbf{A}}_{r+1}(\mathbf{y}_r^* + \mathbf{Z}_r \mathbf{u}_{r+1},) + \bar{\mathbf{a}}_{r+1} \leq \mathbf{v}_{r+1}, \quad (11)$$

$$\bar{\mathbf{A}}_r(\mathbf{y}_r^* + \mathbf{Z}_r \mathbf{u}_{r+1},) + \bar{\mathbf{a}}_r \leq \mathbf{v}_r^*,$$

...

$$\bar{\mathbf{A}}_1(\mathbf{y}_r^* + \mathbf{Z}_r \mathbf{u}_{r+1},) + \bar{\mathbf{a}}_1 \leq \mathbf{v}_1^*,$$

where we wrote the QP as in Eq. (5) and substituted \mathbf{w} into the objective function. In order to ensure that we optimize over the optimal solutions of higher priority tasks, we added Eq. (9) as an additional constraint and substituted Eq (8) into Eqs. (10)-(11). This allows us to solve a stack of hierarchical tasks recursively as originally proposed by [13]. Note that this optimization algorithm is guaranteed to find the optimal solution in a least-squares sense while satisfying priorities.

C. Decomposition of Equations of Motion

Hierarchical inverse dynamics approaches usually have in common that consistency of the variables with physics, i.e. the equations of motion, need to be ensured. In [13] these constraints are expressed as equality constraints (with slacks) resulting in an optimization problem over all variables $\ddot{\mathbf{q}}, \boldsymbol{\tau}, \boldsymbol{\lambda}$. In [14] a mapping into the nullspace of Eq. (1) is obtained from a SVD on Eq. (1). In both cases complexity can be reduced as we will show in the following. We decompose the equations of motion as

$$\mathbf{M}_u(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{N}_u(\mathbf{q}, \dot{\mathbf{q}}) = \boldsymbol{\tau} + \mathbf{J}_{c,u}^T \boldsymbol{\lambda}, \quad (12)$$

$$\mathbf{M}_l(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{N}_l(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{J}_{c,l}^T \boldsymbol{\lambda} \quad (13)$$

where Eq. (12) is just the first n equations of Eq. (1) and eq. (13) is the last 6 equations related to the floating base. The latter can then be interpreted as the Newton-Euler equations of the whole system [16]. They express the change of momentum of the robot as a function of external forces. A remarkable feature of the decomposition in Eqs. (12), (13) is that the torques $\boldsymbol{\tau}$ only occur in Eq. (12) and are exactly determined by $\ddot{\mathbf{q}}, \boldsymbol{\lambda}$ in the form

$$\boldsymbol{\tau} = \mathbf{M}_u(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{N}_u(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{J}_{c,u}^T \boldsymbol{\lambda} \quad (14)$$



Fig. 1. The lower part of the Sarcos Humanoid.

Since $\boldsymbol{\tau}$ is linearly dependent on $\ddot{\mathbf{q}}, \boldsymbol{\lambda}$, for any combination of accelerations and contact forces there always exist a solution for $\boldsymbol{\tau}$. It is given by Eq. (12). Therefore, it is only necessary to use Eq. (13) as a constraint for the equations of motion during the optimization (i.e. the evolution of momentum is the only constraint).

Because of the linear dependence, all occurrences of $\boldsymbol{\tau}$ in the problem formulation (i.e. in Eqs. (3)-(4)) can be substituted by Eq. (14). This reduces the number of variables in the optimization from $(2n + 6 + 6c)$ to $(n + 6 + 6c)$. This decomposition thus saves as many variables as there are DoFs on the robot. This simplification is crucial to reduce the time taken by the optimizer and allowed us to implement the controller in a 1 kHz feedback control loop.

Remark The simplification that we propose¹ can appear trivial at first sight. However, it is worth mentioning that such a decomposition is always ignored in related work despite the need for computationally fast algorithms [13], [14], [12].

III. EXPERIMENTAL SETUP

We now detail the experimental setup, the low-level feedback torque control and the limitations of the hardware.

A. Sarcos Humanoid Robot

The experiments were done on the lower part of the Sarcos Humanoid Robot [3] (Fig. 1). It consists of two legs and a torso. The legs have 7 DOFs each and the torso has 3 DOFs. Given that the torso supports a negligible mass because it is not connected to the upper body of the robot and its motion does not change much the dynamics, we froze these DOFs during the experiments. The legs of the robot are 0.82m high. Each foot is 0.09m wide and 0.25m long, which is rather small for a biped. Note also that the front of the foot is made of a passive joint that is rather stiff, located 10cm before the tip of the foot. The total robot mass is 51kg. The robot is actuated with hydraulics and each joint consists of a Moog Series 30 flow control servo valve with a piston on which a load cell is attached to measure the force at the piston. A position sensor is also located at the joint. Each foot has a 6-axis force sensor and we mounted an IMU on the pelvis of

¹We originally proposed the simplification in this technical note [17].

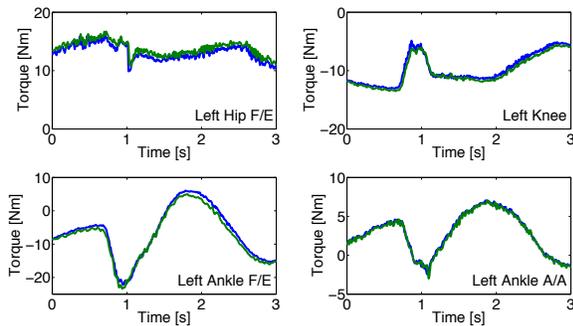


Fig. 2. Example of torque tracking performance during a balancing experiment. The left hip flexion/extension, left knee and left ankle flexion/extension and adduction/abduction joints are shown. Both desired (blue) and actual (green) torques are shown.

the robot from which we measure angular velocities, linear accelerations and the orientation of the robot in an inertial frame. An offboard computer sends control commands to the robot at 1 kHz. The control commands consist of the desired current applied to each valve.

B. Low-level torque control

For each actuator, we implemented a torque feedback controller that ensures that each joint produces the desired force generated by the balance controller. The controller essentially computes desired valve commands given a desired torque, the valve commands representing a desired flow. The controller we implemented is very much inspired from the work in [2], [18], with the difference that we implemented a simpler version where piston velocity feedback has a constant gain. The control law is

$$v = PID(F_{des}, F) + K\dot{x}_{piston} + d \quad (15)$$

where v is the valve command, PID is a PID control according to desired force command and force measured from the load cells, K is a positive gain, \dot{x}_{piston} is the piston velocity (computed from the joint velocity) and d is a constant bias.

This controller design allowed us to achieve good torque tracking performance. It is important to note that it was necessary to achieve good performance in the hierarchical inverse dynamics controller. Fig. 2 illustrates the torque tracking performance during the balancing experiment.

C. Limitations

During the experiments, we suffered from two important limitations. First, our dynamic model is based on the CAD model of the robot. It means that it is not very accurate as it does not take into account the contribution of the hydraulic hoses, the electronics nor any type of friction in the model. Second, we relied on very simple state estimation from the floating base. We just integrated the acceleration readings to get the base velocity and position, using a decay term to damp velocities.

However, these limitations are not too problematic as we show in the experiments. Furthermore, they can be overcome by using system identification [19] and filtering [20] methods that have proven to work well on legged robots.

IV. EXPERIMENTS

We formulated balancing and motion tracking tasks in the framework discussed in Sect. II and evaluated them on the Sarcos Humanoid described in Sect. III. The performance of the controller was evaluated in three different scenarios: balancing experiments in single and double support and a tracking task. A summary of the experiments is shown in the attached movie ².

For all the experiments, we run the hierarchical inverse dynamics controller as a feedback controller that computes directly the desired torque commands. We do not use any joint PD controller for stabilization (i.e. feedback control is only done in task space).

A. Processing Time

The computation time of our solver mainly depends on a) the DoFs of the robot b) the number of contact constraints and c) the composed tasks. All experiments were performed on an Intel Core i7-2600 CPU with 3.40GHz. In our experiments we will use the 14 DoFs lower part of a humanoid to perform several tasks in a 1 kHz control loop. In the following, however, we construct a more complicated stepping task for the full 25 DoFs robot in order to evaluate the speedup from the simplification in Sect. II-C. As sum-

Priority	Nr. of eq(uality) and ineq(uality) constraints	Constraint/Task
1	25 eq	Eq. (12) (not required for simplified problem)
2	6 eq	Newton Euler Eq. (13)
	2×25 ineq	torque limits
	$c \times 6$ eq	kinematic contact constraint
	$c \times 4$ ineq	CoPs reside in sup. polygons
3	$c \times 4$ ineq	GRFs reside in friction cones
	2×25 ineq	joint acceleration limits
	3 eq	PD control on CoM
4	$(2 - c) \times 6$	PD control on swing foot
5	$25 + 6$ eq	PD control on posture
	$c \times 6$ eq	regularizer on GRFs
DoFs: 25		max. time: 5 ms / 3 ms

TABLE I

FULL HUMANOID STEPPING TASK FOR SPEED COMPARISON

marized in Table I the highest two priorities satisfy hardware limitations and dynamic constraints, the third priority tracks the task relevant center of mass and swing foot motion and the remaining priorities resolve redundancies on motion and forces. The problem size changes depending on the number of contacts c ($c = 2$ in double support and $c = 1$ in single support). The proposed decomposition removed 25 equality constraints and 25 optimization variables. We measured the computation time of both versions of the hierarchical solver, one with the full EoM and one with the proposed reduction as plotted in Fig. 3. Looking at the worst case (as this is significant for execution in a time critical control loop) we can drop computation time by 40%. In our experiments with a 14 DoF robot, this speedup allows us to run a 1 kHz control-loop as we will demonstrate in the following sections. It would not have been possible without the simplification. In our speed comparison in Fig. 3 one can see that computation time varies with the number of constrained endeffectors,

²An extended version is available on <http://youtu.be/RIV0aW4vPU8>

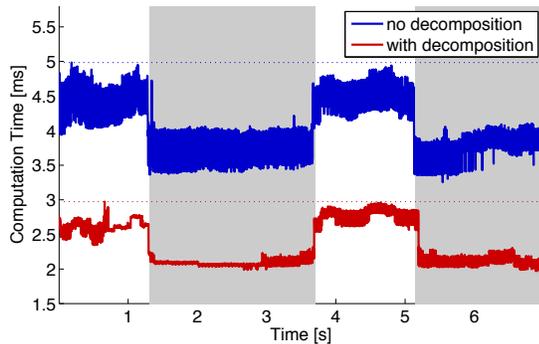


Fig. 3. Processing time of a stepping task (see Table I) using the decomposition proposed in Sect. II-C (red) and the same task performed without the decomposition (blue). The dotted line represents the maximum computation per control cycle respectively. Intervals shaded in gray show the robot in single support phase. In the remaining time the robot is in double support. With the proposed decomposition we decreased the computation time by approximately 40%.

which can be problematic if the number of contacts increases too much (e.g. when using both hands and feet).

B. Balance Control Experiments

Our first experiment is the implementation of a momentum balance controller as originally proposed in [15] and then extend in [11]. The idea is to regulate both the linear and angular momentum of the robot to keep balance. In this implementation, the physical constraints are put in the highest priority. In the second priority, we put the balancing task as well as the force regularization and postural control. The task composition is summarized in Table II.

In [21] a linear mapping

$$\mathbf{H}_G(\mathbf{q})\dot{\mathbf{q}} = \mathbf{h} \quad (16)$$

was derived that maps joint velocities to $\mathbf{h} = [\mathbf{h}_{lin}^T \ \mathbf{h}_{ang}^T]^T$, the system linear and angular momentum expressed at the CoM. The matrix \mathbf{H}_G is called the centroidal momentum matrix. It was applied in [11] to regulate the momentum by computing a desired momentum rate of change via PD control

$$\dot{\mathbf{h}}_{ref} = \mathbf{P} \begin{bmatrix} m(\mathbf{x}_{cog,des} - \mathbf{x}_{cog}) \\ \mathbf{0} \end{bmatrix} + \mathbf{D}(\mathbf{h}_{des} - \mathbf{h}) + \dot{\mathbf{h}}_{des} \quad (17)$$

where \mathbf{P} and \mathbf{D} are positive-definite gain matrices. Typically, this can be used to regulate the position of the center of mass while damping its linear velocity and the angular momentum. The derivative of Eq. (16) allows us to express a controller on the system momentum

$$\dot{\mathbf{h}} = \mathbf{H}_G\ddot{\mathbf{q}} + \dot{\mathbf{H}}_G\dot{\mathbf{q}} \quad (18)$$

$$= \begin{bmatrix} \mathbf{I}_{3 \times 3} & \mathbf{0}_{3 \times 3} & \cdots \\ [\mathbf{x}_{cog} - \mathbf{x}_i]_{\times} & \mathbf{I}_{3 \times 3} & \cdots \end{bmatrix} \boldsymbol{\lambda} + \begin{bmatrix} m\mathbf{g} \\ \mathbf{0} \end{bmatrix}, \quad (19)$$

where $m\mathbf{g}$ is the gravitational force applied at the CoM and $[\cdot]_{\times}$ maps a vector to a skew symmetric matrix, s.t. $[\mathbf{x}]_{\times}\boldsymbol{\lambda} = \mathbf{x} \times \boldsymbol{\lambda}$. Eq. (19) is the general formula for the *change of momentum* of a rigid multi-body system. We see that we can express the rate of momentum change either as a function of $\ddot{\mathbf{q}}$ as in Eq. (18) or as a function of $\boldsymbol{\lambda}$ as in Eq. (19). We can express this task either as a force task or as a kinematic task (the matrix in front of $\ddot{\mathbf{q}}$ or $\boldsymbol{\lambda}$ being the Jacobian of the task).

Priority	Nr. of eq(uality) and ineq(uality) constraints	Constraint/Task
1	6 eq	Newton Euler Eq. (13)
	2×14 ineq	torque limits
	2×6 eq	kinematic contact constraint
	2×4 ineq	CoPs reside in sup. polygons
	2×4 ineq	GRFs reside in friction cones
2	2×14 ineq	joint acceleration limits
	6 eq	PD control on system momentum, Eq. (17)
	$14 + 6$ eq	PD control on posture
	2×6 eq	regularizer on GRFs
DoFs: 14		max. time: 0.4 ms

TABLE II

HIERARCHY OF DOUBLE SUPPORT BALANCING TASK

In this experiment we use the kinematic representation and in the tracking experiment we use the force representation. We note that in general, expressing the momentum control with Eq. (19) might be better because we do not have to compute $\dot{\mathbf{H}}_G$, which usually is acquired through numerical derivation and might suffer from magnified noise.

In our first experiment we pushed the robot impulsively with a stick at various contact points with different force directions. To ensure that the robot is really balancing, the same experiment was conducted when running a simple inverse dynamics algorithm with contact forces optimization [8]. As expected, the balance controller showed a better balancing performance and did not fall over as it was the case for [8] as can be seen in the video. When pushing the robot with a constant force at various parts, it stayed in balance and adapted its posture in a compliant manner. We also tested the controller when the feet were not co-planar, but one foot was put on top of a block as can be seen in the movie.

In the planar posture, even pushes with a rather high magnitude were absorbed and the robot kept standing. The change in momentum was damped out quickly and the CoM was tracked after an initial disturbance as can be seen in Fig. 4. The CoPs remained inside of the support polygons and were tracked well. We notice from Fig. 4 that the CoPs, as they were predicted from the dynamics model, are approximately correct (within 2cm error). However, one can expect that a higher precision might be needed to achieve dynamic motions which could be achieved with an inertial parameters estimation procedure [19].

For our next experiment we put the biped on a rolling platform and rotated and moved it with a rather fast change of directions. Although, the CoM was moving away from its desired position initially, the momentum change was damped out and the robot kept standing and recovered CoM tracking. The stationary feet indicated that forces were applied that were consistent with our CoP boundaries.

In an additional scenario the biped was standing on a balancing board. We ran the experiment with two configurations for the robot: in one case the robot is standing such that the board motion happens in the sagittal plane and in the other case the motion happens in the lateral plane. Fig. 5 shows results when the motion happens in the lateral plane. In this case, the slope was varied in a range of $[-9.5^\circ; 9.5^\circ]$. Even for quite rapid changes in the slope, the feet remained flat on the ground. Compared to the push experiment the CoPs were moving in a wider range, but still remained in the interior

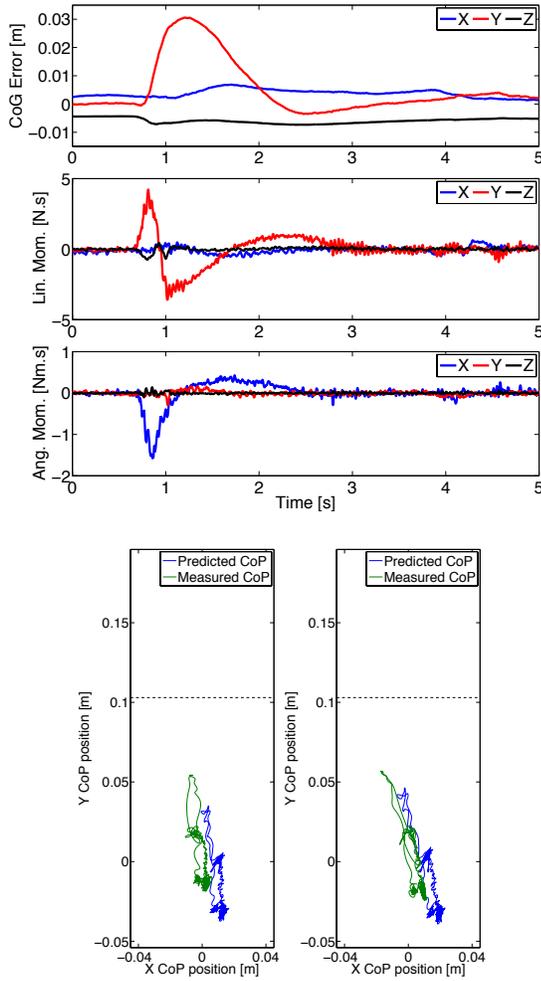


Fig. 4. This figure shows the recovery of desired CoG position, linear and angular momentum after a push from the front. X points in the robot right direction, Y in the forward direction and Z in the vertical direction. The change in CoP for each leg is also plotted in the lower graph. The plot has the same ratio as the feet and the horizontal dashed line shows the position of the passive joint at the front of the foot.

of the foot soles with a margin. In this case, we notice a discrepancy in the predicted contact forces and the real ones, making the case for a better dynamics model. Yet, the robot was still able to balance.

When we increased the pushes on the robot, eventually the momentum could not be damped out fast enough anymore and the robot reached a situation where the optimization could not find solutions anymore and the biped fell. In these cases the constraint that the feet have to be stationary was too restrictive. A higher level controller that takes into account stepping [22], [23] becomes necessary to increase the stability margin. As the balancing task required only two hierarchies, it took approximately 0.4 ms to solve the optimization problem at each control cycle.

C. Tracking Experiments

We also performed a tracking experiment. The goal was to track a desired CoM motion while satisfying the physical constraints. The lower priority tasks consist of joint posture tracking and contact forces regularization (i.e. we minimize tangential contact forces). We wrote the CoM tracking task

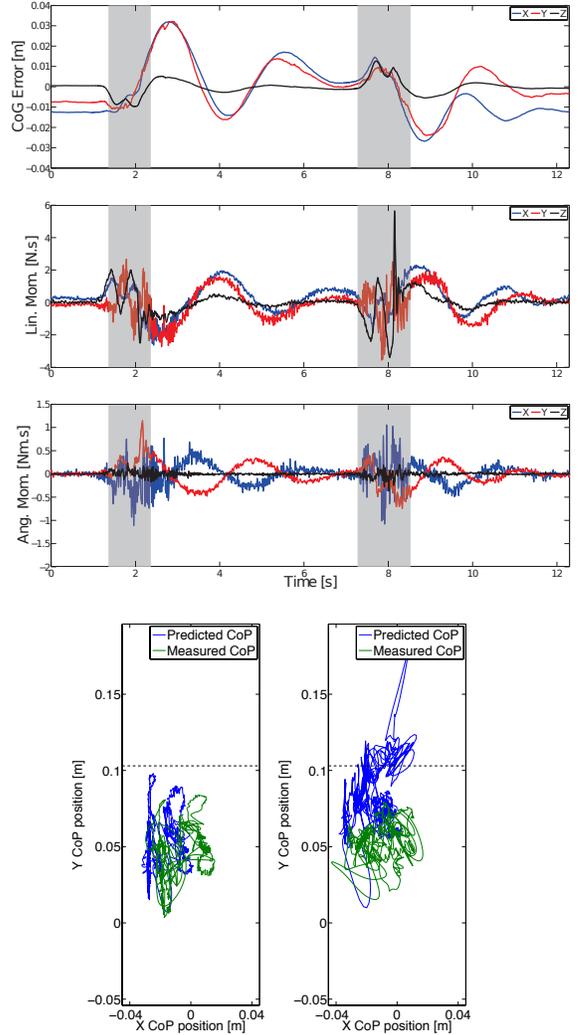


Fig. 5. This figure shows the recovery of desired CoG position, linear and angular momentum during the balance board experiment. X points in the robot right direction, Y in the forward direction and Z in the vertical direction. The grey shades are the moments when the board is moving, first up, then down. The total angle motion is from -9.5° to 9.5° with respect to horizontal.

as a force task, using Eq. (19) where the desired CoM acceleration is computed using a PD controller. The full hierarchy of tasks used in this experiment is shown in Table III.

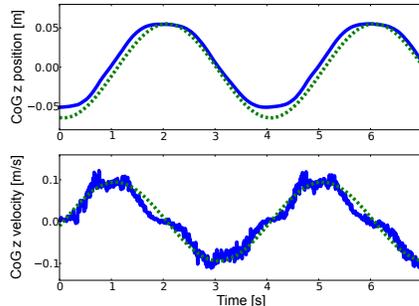
The CoM was tracking sine waves of various forms. The first one was a 0.3 Hz sine wave with 0.02 m amplitude in the forward direction and 0.03 m in the vertical direction. The second one had a larger amplitude of 0.06 m only in the vertical direction. Fig. 6 shows the typical tracking performance during these tasks. During the tracking experiments the robot was still able to handle a certain level of disturbances, as can be seen in the attached video but not as much as when the angular momentum was also regulated.

In this experiment we use much more hierarchies than in the previous one. In this case, the controller was able to compute the control commands in an average of 0.9 ms (standard deviation of 0.045). We see that in the current state, the hierarchical inverse dynamics is at the limit of the number of hierarchies it can solve in the 1 kHz control cycle.

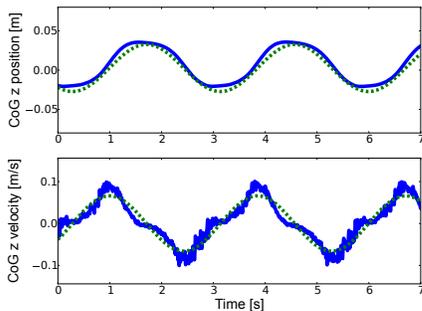
Priority	Nr. of eq(uality) and ineq(uality) constraints	Constraint/Task
1	6 eq	Newton Euler Eq. (13)
2	2×14 ineq	torque limits
	2×6 eq	kinematic contact constraint
	2×4 ineq	CoPs reside in sup. polygons
	2×4 ineq	GRFs reside in friction cones
3	2×14 ineq	joint acceleration limits
	3 eq	PD control on CoG
4	$14 + 6$ eq	PD control on posture
5	2×6 eq	regularizer on GRFs
DoFs: 14		max. time: 0.9 ms

TABLE III

HIERARCHY OF TASKS USED IN THE TRACKING EXPERIMENT



(a) 0.25 Hz high amplitude tracking task



(b) 0.3 Hz low amplitude tracking task

Fig. 6. Plot of the vertical CoM tracking performance for two tracking tasks. Both the positions (upper graphs) and velocities (lower graph) are shown in the plots. We note that both the desired positions and velocities (green dashed line) match very well the actual (blue plain line). The position RMSE is 0.0094m for the upper plot and 0.0058 for the lower plot.

D. Balance in Single Support

The experiments in the previous sections were done while the robot remained in double support. The goal of this experiment is to show that the controller can handle more complicated tasks involving contact switching and that the robot is able to balance on a single foot in face of disturbances.

The robot moves from a double support position to a single support position where the swing foot is lifted 10 cm above the ground while balancing. This motion consists of 3 phases. First, the robot moves its CoM towards the center of the stance foot. Then an unloading phase occurs during which the contact force regularization enforces a zero contact force to guarantee a continuous transition when the double support constraint is removed. In the final phase, a task controlling the motion of the swing foot is added to the hierarchy.

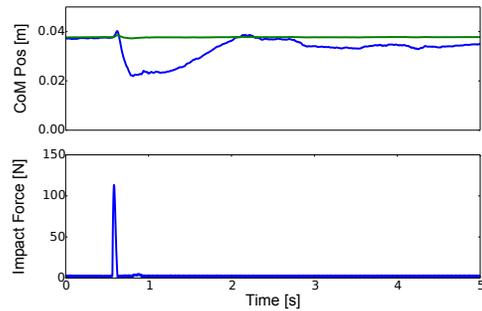


Fig. 7. This plot shows the CoM tracking (upper graph) and the force exerted at the robot (lower graph) when it was balancing on one foot. The force was measured with a load cell that was attached to our push-stick. After the impulsive push the CoM (blue) deviated from the desired (green), but tracking was established again after a short duration.

Priority	Nr. of eq(uality) and ineq(uality) constraints	Constraint/Task
1	6 eq	Newton Euler Eq. (13)
2	2×14 ineq	torque limits
	2×4 ineq	CoPs reside in sup. polygons
	2×4 ineq	GRFs reside in friction cones
	2×14 ineq	joint acceleration limits
3	6 eq.	Linear and angular momentum control
	12/6 eq.	kinematic contact constraint
4	0/6 eq.	Cartesian foot motion (swing)
	14 eq.	PD control on posture
	$2 \times 6/1 \times 6$ eq.	regularizer on GRFs
DoFs: 14		max. time: 1.05 ms

TABLE IV

HIERARCHY FOR SINGLE SUPPORT BALANCING EXPERIMENT

Our contact switching strategy is simple but guarantees that continuous control commands were sent to the robot. For more complicated tasks, such guarantees can always be met by using automatic task transitions such as in [24]. The composition of hierarchies is summarized in Table IV. Concerning computation time, the controller computes a solution in average well below 1ms but a maximum at 1.05ms is reached a few times during the unloading phase due to many constraints getting active at the same time.

Once in single support, we pushed the robot with a stick that has a force sensor attached to its tip allowing to measure the magnitude of the push. The robot was pushed impulsively with peak forces up to 150N and impulses between 4.5Ns and 5.8Ns, which compares nicely with other balance experiments [4]. Fig. 7 shows the CoM tracking and exerted push force during a typical push. We can notice the fast recovery of the CoM position and the damped behavior which allows the CoM to stay above support polygon. It is worth mentioning again that the foot size of the robot is rather small compared to other humanoids.

V. DISCUSSION

In the following we discuss the results we presented and how they relate to other approaches.

A. Relation to other balancing approaches

The balance controller presented in the first experiment is a slight reformulation of the momentum-based controller presented by Lee and Goswami [11], [25]. Our formulation has the great advantage to solve a single optimization problem instead of several ones and can therefore guarantee that

the control law will be consistent with all the constraints (joint limits, accelerations, torque saturation, CoP limits and contact force limitations). Furthermore, we search over the full set of possible solutions and thus we are guaranteed to find the optimum, where [11], [25] optimize over sub-parts of the variables sequentially. It is also different from the work of [15] because we explicitly take into account contact forces in the optimization and not purely kinematics.

The balance controller is very much related to the work of Stephens et al. [12]. In [12], the authors write the whole optimization procedure using Eq. (1) with constraints similar to what we have. However, the optimization problem being complicated, they actually solve a simpler problem where the contact forces are first determined and then desired accelerations and torques are computed through a least-square solution. From that point of view, torque saturation and limits on accelerations are not accounted for. In our experiments, no tradeoff is necessary and we solve for all the constraints exactly. Further, the capability of strict task prioritization makes the design of more complicated tasks like balancing on one foot easier.

B. Relations to other hierarchical inverse dynamics solvers

In the hierarchical task solver that was used in this paper, we exploited the fact that the null space mapping of one priority could be computed in parallel with solving the QP of the same priority. On the other hand [14] suggest a more efficient way of dealing with inequalities by spending more computation time in constructing the problem. It is not clear which approach would be more efficient and a speed comparison would be very interesting. Their approach can also profit from the decomposition proposed in this paper, then it will not be required to compute an SVD of the full equations of motion, but only of the last six rows. In this paper our focus was the experimental evaluation of the discussed problem formulation, but as tasks become more complex and we will use the full robot, more efficient solvers will become necessary.

VI. CONCLUSION

In this paper we have presented experimental results using a hierarchical inverse dynamics controller. To the best of our knowledge, it is the first implementation of such an approach on a torque controlled humanoid robot in a fast control loop. We have presented both balance and tracking experiments. Our results suggest that the use of complete dynamic models and hierarchies of tasks for feedback control is a feasible approach, despite the model inaccuracies and computational complexity.

ACKNOWLEDGMENT

We would like to thank Ambarish Goswami and Seungkook Yun for hosting us at the Honda Research Institute for one week and for their precious help in understanding the original momentum-based controller. We would also like to thank Ambarish Goswami and Sung-Hee Lee for giving us an early access to their publication. We are also grateful to Daniel Kappler for helping us with the videos, Nick Rotella for helping with the data acquisition from the IMU and Sean Mason for the help with the inertial parameters of the robot.

REFERENCES

- [1] M. Hutter, M. A. Hoepffinger, C. Gehring, M. Bloesch, C. D. Remy, and R. Siegwart, "Hybrid Operational Space Control for Compliant Legged Systems," in *R:SS*, 2012.
- [2] T. Boaventura, C. Semini, J. Buchli, M. Frigerio, M. Focchi, and D. Caldwell, "Dynamic Torque Control of a Hydraulic Quadruped Robot," in *ICRA*, 2012.
- [3] G. Cheng, H. Sang-Ho, A. Ude, J. Morimoto, J. G. Hale, J. Hart, J. Nakanishi, D. Bentivegna, J. Hodgins, C. Atkeson, M. Mistry, S. Schaal, and M. Kawato, "CB: Exploring Neuroscience with a Humanoid Research Platform," in *ICRA*, 2008.
- [4] C. Ott, M. A. Roa, and G. Hirzinger, "Posture and balance control for biped robots based on contact force optimization," in *Humanoids*, 2011.
- [5] M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal, "Learning, Planning, and Control for Quadruped Locomotion over Challenging Terrain," *IJRR*, vol. 30, no. 2, pp. 236–258, 2011.
- [6] L. Saab, O. Ramos, N. Mansard, P. Soueres, and J.-Y. Fourquet, "Generic Dynamic Motion Generation with Multiple Unilateral Constraints," in *IROS*, 2011.
- [7] J. Salini, V. Padois, and P. Bidaud, "Synthesis of Complex Humanoid Whole-Body Behavior: A Focus on Sequencing and Tasks Transitions," in *ICRA*, 2011.
- [8] L. Righetti and S. Schaal, "Quadratic programming for inverse dynamics with optimal distribution of contact forces," in *Humanoids*, 2012.
- [9] L. Righetti, J. Buchli, M. Mistry, M. Kalakrishnan, and S. Schaal, "Optimal distribution of contact forces with inverse-dynamics control," *IJRR*, vol. 32, no. 3, pp. 280–298, 2013.
- [10] S.-H. Hyon, J. G. Hale, and G. Cheng, "Full-Body Compliant Human-Humanoid Interaction: Balancing in the Presence of Unknown External Forces," *Transactions on Robotics*, vol. 23, no. 5, pp. 884–898, 2007.
- [11] S.-H. Lee and A. Goswami, "A momentum-based balance controller for humanoid robots on non-level and non-stationary ground," *Autonomous Robots*, vol. 33, pp. 399–414, 2012.
- [12] B. J. Stephens and C. G. Atkeson, "Dynamic Balance Force Control for compliant humanoid robots," in *IROS*, 2010.
- [13] M. de Lasa, I. Mordatch, and A. Hertzmann, "Feature-Based Locomotion Controllers," *ACM Transactions on Graphics*, vol. 29, no. 3, 2010.
- [14] N. Mansard, "A dedicated solver for fast operational-space inverse dynamics," in *ICRA*, 2012.
- [15] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Resolved momentum control: humanoid motion planning based on the linear and angular momentum," in *IROS*, 2003.
- [16] P. Wieber, "Holonomy and nonholonomy in the dynamics of articulated motion," *Fast motions in biomechanics and robotics*, pp. 411–425, 2006.
- [17] A. Herzog, L. Righetti, F. Grimmering, P. Pastor, and S. Schaal, "Momentum-based balance control for torque-controlled humanoids," vol. <http://arxiv.org/abs/1305.2042v1>, 2013.
- [18] T. Boaventura, M. Focchi, M. Frigerio, J. Buchli, C. Semini, G. A. Medrano-Cerda, and D. Caldwell, "On the role of load motion compensation in high-performance force control," in *IROS*, 2012.
- [19] M. Mistry, S. Schaal, and K. Yamane, "Inertial Parameter Estimation of Floating Base Humanoid Systems using Partial Force Sensing," in *Humanoids*, 2009.
- [20] M. Bloesch, M. Hutter, M. H. Hoepffinger, C. D. Remy, C. Gehring, and R. Siegwart, "State estimation for legged robots-consistent fusion of leg kinematics and IMU," in *R:SS*, 2012.
- [21] D. E. Orin and A. Goswami, "Centroidal Momentum Matrix of a humanoid robot: Structure and Properties," in *IROS*, 2008.
- [22] J. Pratt, T. Koolen, T. De Boer, J. Rebula, S. Cotton, J. Carff, M. Johnson, and P. Neuhau, "Capturability-based analysis and control of legged locomotion, Part 2: Application to M2V2, a lower-body humanoid," *IJRR*, vol. 31, no. 10, 2012.
- [23] J. Urata, K. Nshiwaki, Y. Nakanishi, K. Okada, S. Kagami, and M. Inaba, "Online Walking Pattern Generation for Push Recovery and Minimum Delay to Commanded Change of Direction and Speed," in *IROS*, 2012.
- [24] G. Jarquin, A. Escande, G. Archavaleta, T. Moulard, E. Yoshida, and V. Parra-Vega, "Real-time smooth task transitions for hierarchical inverse kinematics," in *Humanoids*, 2013.
- [25] S.-H. Lee and A. Goswami, "Ground reaction force control at each foot: A momentum-based humanoid balance controller for non-level and non-stationary ground," in *IROS*, 2010, pp. 3157–3162.