Generation of Dynamic Humanoid Behaviors Through Task-Space Control with Conic Optimization

Patrick M. Wensing, David E. Orin

***Abstract*— This paper presents a new formulation of prioritized task-space control for humanoids that is used to develop a dynamic kick and dynamic jump in a 26 degree of freedom simulated system. The demonstrated motions are controlled through a real-time conic optimization scheme that selects appropriate joint torques and contact forces. More specifically, motions are characterized in appropriate task spaces, and the real-time optimizer solves the task-space control problem while accounting for user-defined priorities between the tasks. In contrast to previous solutions of the Prioritized Task-Space Control (PTSC) problem for humanoids, the solution presented here satisfies the ZMP constraint and ground friction limitations at all levels of priority, and is general to periods of flight as well as support. All generated motions include control of the system's centroidal angular momentum, which leads to emergent whole-body behaviors, such as arm-swing, that are not specified by the designer. In addition, compared to a previous quadratic programming solution of the PTSC problem, our approach gains a factor of 2 speedup in its required computational time. This speedup allows the control approach to operate at real-time rates of approximately 200 Hz.**

I. Introduction

While the field of humanoid robots has received much attention over the past decade, control of dynamic whole-body humanoid behaviors remains a difficult objective. Although challenging, the proper coordination of humanoids' many degrees of freedom (DOF) offers many benefits such as in the recovery from disturbances through arm-swinging [1] or to redirect the torso during a fall through managing inertial couplings from the legs [2]. While these additional DOFs provide more flexibility to whole-body motion, they also add complexity to select appropriate joint torques in real time. Additionally, the large number of DOFs make it difficult to author motions by hand, which has lead to widespread use of motion capture techniques for whole-body motion generation [1], [3], [4], [5].

Task-space (or operational-space) control [6], [7] provides a framework that significantly eases the burden of authoring whole-body behaviors. Task-space trajectories can be designed in intuitive motion spaces. For instance, walking motions can be specified through design of foot and center of mass trajectories. Without the need for any inverse kinematics, online task-space controllers provide an elegant solution to generate whole-body walking behaviors [8].

In this paper, Prioritized Task-Space Control (PTSC) is used to generate dynamic behaviors such as the kick and jump shown in Fig. 1. The common control structure used to create these behaviors is shown in Fig. 2. A high-level state machine is used to manage the desired task behaviors, while a new conic optimization formulation of PTSC is used to select joint torques at every control step. Through the formulation of PTSC described here, the control loop is able to be closed at rates of approximately 200 Hz on a commodity laptop, which is approximately double that of previous formulations. Generated motions automatically satisfy the ZMP constraint on level terrain and are general to produce balanced motions on uneven surfaces. Additionally,

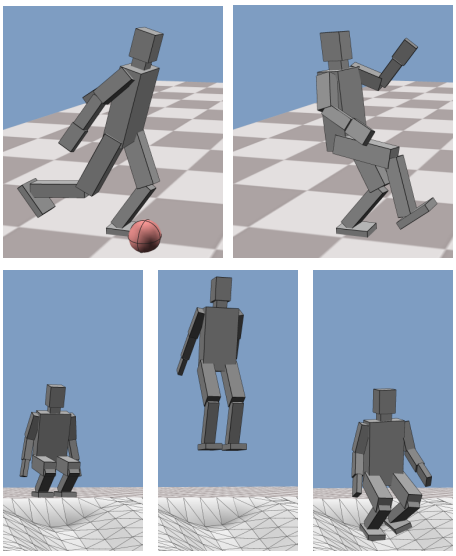

Fig. 1. A new formulation of the Prioritized Task-Space Control problem allows us to control dynamic behaviors such as a kick and a jump at real-time rates and in challenging environments. The use of Centroidal Momentum control results in rich emergent arm motions to maintain balance without any upper-body motion authoring.

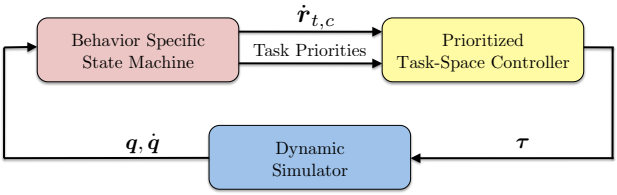

Fig. 2. Block diagram for behavioral controllers used in this work. A high-level state machine manages the commanded task dynamics $\dot{r}_{t,c}$ and priorities at each instant. A prioritized task-space controller then finds joint torques τ which keep feet planted and prevent slipping while realizing the desired task dynamics. This control loop is closed at real-time rates of approximately 200 Hz.

P. Wensing is a PhD student in the Department of Electrical and Computer Engineering, The Ohio State University: wensing.2@osu.edu

D. E. Orin is a Professor Emeritus in the Department of Electrical and Computer Engineering, The Ohio State University: orin.1@osu.edu

centroidal momentum control [9] is used to alleviate the need to design specific upper-body trajectories, which instead emerge from our control strategies.

Related Work in Whole-Body Humanoid Control

While we adopt a task-space approach here, many previous approaches assume access to pre-designed joint trajectories and apply an inverse dynamics controller to select joint torques. Under the assumption of sufficient friction on the feet, Mistry et al. [10] use a projection method to select appropriate torques and to resolve over-actuation in double support. Improvements to their approach [11] attempt to push ground reaction forces inside their frictional boundaries. Still, no motion modification is applied, which can lead to balance failure when aggressive joint accelerations are desired. This problem can be accounted for by the application of motion filtering techniques that modify a motion to remain balanced [5], [12]. Park et al. formulate a stringent motion filter as a conic optimization problem [13] which forces the feet to remain in full support. Other motion filtering techniques incorporate long-term balance strategies based on simple models while trying to follow motion [3]. Again, all these approaches require whole-body joint trajectories which are costly to plan in advance and even more difficult to modify online.

Task-space control provides an alternative to the aforementioned approaches, and can be solved in a variety of ways. Projection methods of Park and Khatib [7] and Sentis et al. [14] allow task-priorities to be enforced and deal well with support foot constraints as long as the desired task dynamics lead to balanced motion. Just as a motion filter can be applied to balance pre-designed joint trajectories that violate contact force constraints (such as ZMP), similar filters can be applied to physically unrealizable task-space trajectories. For example de Lasa et al. [8] and Salini et al. [15] both use a series of quadratic programs (QPs) to solve PTSC while satisfying ground reaction force constraints. This paper is inspired by their work and reformulates their QPs into a conic optimization problem where PTSC can be solved at real-time rates. This improvement allows high-bandwidth control of highly dynamic movements such as the kick and jump showcased here.

II. Task-Space Control for Humanoids

A. Notation

This section introduces notation and summarizes the previous approaches that are used to solve the PTSC problem for humanoids. Given an n degree-of-freedom (DOF) floating-base humanoid, as shown in Fig. 3, the system's configuration can be described by:

$$\boldsymbol{q} = \begin{bmatrix} \boldsymbol{q}_b^T & \boldsymbol{q}_a^T \end{bmatrix}^T. \tag{1}$$

Here $\boldsymbol{q}_b \in SE(3)$ is the unactuated position and orientation of the system's floating base, described in Fig. 3 and $\boldsymbol{q}_a$ denotes the actuated joints' configurations. The system's joint rate $\dot{\boldsymbol{q}} \in \mathbb{R}^{n+6}$ and acceleration vectors and $\ddot{\boldsymbol{q}} \in \mathbb{R}^{n+6}$

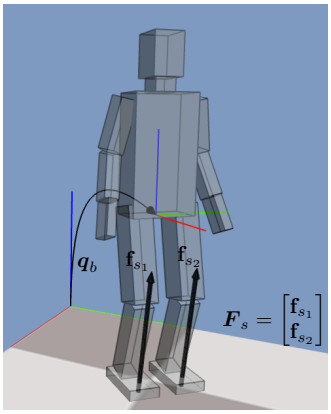

Fig. 3. The floating-base configuration $\boldsymbol{q}_b \in SE(3)$ describes the position and orientation of the floating-base fixed coordinate system with respect to an earth fixed frame. The combined contact force vector $\boldsymbol{F}_s$ is comprised of local contact wrenches (force and moment) $\mathbf{f}_{s_1} \in \mathbb{R}^6$ and $\mathbf{f}_{s_2} \in \mathbb{R}^6$ which act on bodies in support.

are partitioned similarly. The standard dynamic equations of motion are:

$$\boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{S}_a^T \boldsymbol{\tau} + \boldsymbol{J}_s(\boldsymbol{q})^T \boldsymbol{F}_s \tag{2}$$

where $\boldsymbol{H}$, $\boldsymbol{C}\dot{\boldsymbol{q}}$, and $\boldsymbol{G}$ are the familiar mass matrix, velocity product terms, and gravitational terms, respectively [16]. Here $\boldsymbol{F}_s$ collects ground reaction forces (GRFs) for appendages in support, as described in Fig. 3, and $\boldsymbol{J}_s$ is a combined support Jacobian. The matrix $\boldsymbol{S}_a = \begin{bmatrix} \mathbf{0}_{n\times 6} & \mathbf{1}_{n\times n} \end{bmatrix}$ is a selection matrix for the actuated joints.

In order to author whole-body behaviors, it is often convenient to characterize the system's desired dynamics in a task (or operational) space [6]. Task velocities $\dot{\boldsymbol{x}}_t$ are related to joint-rates $\dot{\boldsymbol{q}}$ by the standard relationship:

$$\dot{\boldsymbol{x}}_t = \boldsymbol{J}_t(\boldsymbol{q})\, \dot{\boldsymbol{q}} \tag{3}$$

where $\boldsymbol{J}_t(\boldsymbol{q})$ is a task Jacobian. Alternatively, our task may not arise from a Jacobian relationship, such as in the control of the system's net angular momentum. To accommodate such a generalization, we relax this definition of a task to any relationship of the form:

$$\boldsymbol{r}_t = \boldsymbol{A}_t(\boldsymbol{q})\, \dot{\boldsymbol{q}}\,. \tag{4}$$

Given commanded instantaneous task dynamics $\dot{\boldsymbol{r}}_{t,c}$ the task-space control problem is to find joint torques $\boldsymbol{\tau}$ that result in joint accelerations $\ddot{\boldsymbol{q}}$ with:

$$\boldsymbol{A}_t\ddot{\boldsymbol{q}} + \dot{\boldsymbol{A}}_t\dot{\boldsymbol{q}} = \dot{\boldsymbol{r}}_t \tag{5}$$

such that $\dot{\boldsymbol{r}}_t$ most closely matches $\dot{\boldsymbol{r}}_{t,c}$. Depending on the choice of $\dot{\boldsymbol{r}}_{t,c}$ additional freedoms may be used to match lower-priority task dynamics. For systems in support, there are a variety of ways to solve this problem, which differ in how they account for the presence of external forces $\boldsymbol{F}_s$ in (2). These different methods are discussed in the following subsections.

B. Projection Methods for Task-Space Control During Support

In periods of support, the equations of motion (2) are often considered under the constraint of zero acceleration at the support bodies:

$$\ddot{\boldsymbol{x}}_s = \boldsymbol{J}_s\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_s\dot{\boldsymbol{q}} = \boldsymbol{0}\,. \tag{6}$$

The support forces $\boldsymbol{F}_s$ required to ensure (6) can be substituted into (2) to produce a set of constrained dynamic equations of motion:

$$\boldsymbol{H}\ddot{\boldsymbol{q}} + \boldsymbol{N}_s^T\left(\boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{G}\right) + \boldsymbol{\gamma}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{N}_s^T\boldsymbol{S}_a^T\boldsymbol{\tau} \tag{7}$$

where $\boldsymbol{\gamma} = \boldsymbol{J}_s^T(\boldsymbol{J}_s\boldsymbol{H}^{-1}\boldsymbol{J}_s^T)^\dagger\dot{\boldsymbol{J}}_s\dot{\boldsymbol{q}}$ is a velocity dependent term due to the constraints and

$$\boldsymbol{N}_s^T = \boldsymbol{1} - \boldsymbol{J}_s^T\left(\boldsymbol{J}_s\boldsymbol{H}^{-1}\boldsymbol{J}_s^T\right)^\dagger\boldsymbol{J}_s\boldsymbol{H}^{-1} \tag{8}$$

is the dynamically-consistent null-space projector for support [7]. The symbol $(\cdot)^\dagger$ denotes the Moore-Penrose pseudo-inverse of the enclosed matrix. In a rough sense, these equations of motion are a projection of the original equations of motion onto the subset of the configuration space that is consistent with the support constraint. Since no constraints are placed on the GRFs, these dynamic equations are in fact not correct if torques are supplied that would lift the foot off the ground, or cause the foot to slip.

Starting from (7) the works of Park and Khatib [7] and Sentis et al. [14] extend the traditional prioritized operational-space control framework for manipulators to the case of humanoids in support. Just as in the above derivation, all their control laws are predicated on the assumption that given a joint input $\boldsymbol{\tau}$, the ground is capable of supplying support forces $\boldsymbol{F}_s$ to ensure the foot remains stationary. This assumption is powerful, and often valid on level terrain since GRFs are dominated by a gravity compensation force. Still, for highly dynamic motions, additional care has to be taken to ensure that the commanded task dynamics do not require GRFs outside their unidirectional or frictional boundaries. This becomes increasingly difficult on more challenging terrains. Their approach uses a nested series of task nullspace projectors to enforce task priority [14].

C. Ground Reaction Force Constraint Modeling

To account for the true constraints on ground reaction forces (GRFs) during the selection of joint torques, others ([8], [15], [17], [18]) have proposed the use of constrained quadratic programming to solve the PTSC problem. These approaches treat $\boldsymbol{F}_s$ as a control variable and optimize the selection of $\boldsymbol{\tau}$ and $\boldsymbol{F}_s$ under appropriate constraints. In order to approximate the constraints on each net foot wrench $\mathbf{f}_{s_i}$, it is customary to represent each wrench as a combination of pure forces that act at the corners of the feet, as shown in Fig. 4. Unidirectional and frictional constraints can then easily be enforced on the individual vertex forces $\boldsymbol{f}_{s_{ij}} \in \mathbb{R}^3$. Given a coefficient of friction μ_i for foot i, these forces must reside inside a friction cone:

$$\boldsymbol{\mathcal{C}}_i := \left\{(f_x, f_y, f_z) \in \mathbb{R}^3 \;\middle|\; \sqrt{f_x^2 + f_y^2} \le \mu_i f_z\right\}. \tag{9}$$

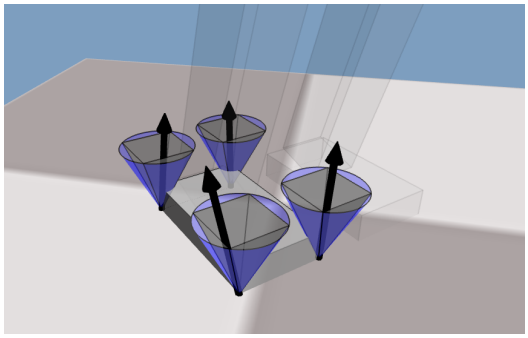

Fig. 4. The contact wrench $\mathbf{f}_{s_i} \in \mathbb{R}^6$ on foot i is approximated by a distribution of pure forces (no moment) to a lattice of contact points underneath the feet. These forces are denoted as $\boldsymbol{f}_{s_{ij}} \in \mathbb{R}^3$ in the text. Forces are constrained to lie inside the friction cone $\boldsymbol{\mathcal{C}}_i \subset \mathbb{R}^3$ (show in blue), or in a more restrictive friction pyramid $\boldsymbol{\mathcal{P}}_i \subset \mathbb{R}^3$ (shown in gray) as in previous work.

To simplify optimization, previous work has approximated these cones by a friction pyramid $\boldsymbol{\mathcal{P}}_i \subset \boldsymbol{\mathcal{C}}_i$ as shown in Fig. 4. To aid further development, we define N_S to be the number of feet in support, and N_{P_i} as the number of local contact points for foot i.

D. Quadratic Programming Methods for Task-Space Control During Support

Previous work [8], [18] formulates a quadratic program (QP) to find contact forces, joint accelerations $\ddot{\boldsymbol{q}}$, and joint torques $\boldsymbol{\tau}$ that are consistent with the dynamic equations of motion, and most closely match the commanded task dynamics. An example of such a QP is given below.

$$\min_{\ddot{\boldsymbol{q}},\boldsymbol{\tau},\boldsymbol{f}_{s_{ij}}} \quad \frac{1}{2}||\boldsymbol{A}_t\,\ddot{\boldsymbol{q}} + \dot{\boldsymbol{A}}_t\,\dot{\boldsymbol{q}} - \dot{\boldsymbol{r}}_{t,c}||^2 \tag{10}$$

$$\text{subject to } \boldsymbol{H}\,\ddot{\boldsymbol{q}} + \boldsymbol{C}\,\dot{\boldsymbol{q}} + \boldsymbol{G} = \boldsymbol{S}_a^T\boldsymbol{\tau} + \sum_{i=1}^{N_S}\sum_{j=1}^{N_{P_i}}\boldsymbol{J}_{s_{ij}}^T\boldsymbol{f}_{s_{ij}}$$

$$\boldsymbol{f}_{s_{ij}} \in \boldsymbol{\mathcal{P}}_i \qquad \forall i \in \{1,\ldots,N_S\},\ \forall j \in \{1,\ldots,N_{P_i}\} \tag{11}$$

$$\boldsymbol{J}_s\,\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_s\dot{\boldsymbol{q}} = \boldsymbol{0} \tag{12}$$

$$\underline{\boldsymbol{\tau}} \le \boldsymbol{\tau} \le \overline{\boldsymbol{\tau}}.$$

Here each $\boldsymbol{J}_{s_{ij}} \in \mathbb{R}^{3\times(n+6)}$ is the Jacobian for the contact point where $\boldsymbol{f}_{s_{ij}}$ acts. Joint torque limits are described by the vectors $\underline{\boldsymbol{\tau}}$ and $\overline{\boldsymbol{\tau}}$. The unidirectional constraints also imposed on the GRFs through (11) combined with the support acceleration constraint (12) assure that the optimized $\ddot{\boldsymbol{q}}$ satisfy the ZMP constraint.

To optimize for lower priority tasks, the QP may be modified and additional constraints included to ensure that high-priority task dynamics are not corrupted. This nested series of QPs (or stack of QPs) replaces the nested series of nullspace projectors that are required for projection based methods. In the work of Mansard [17], an elimination of variables is employed that reduces the size of the above QP at each step, but this reduction is specific to periods of

support. In the next section, we propose a reduction that is general to periods of flight as well as support.

E. Net Momentum Control

As a quick aside, we note that the previous formulation is general to control features that cannot be described with a Jacobian relationship. Most notably, this generality can be exploited to control the system's centroidal momentum $\boldsymbol{h}_G$ [9]. The centroidal momentum $\boldsymbol{h}_G = [\boldsymbol{k}_G^T,\ \boldsymbol{l}_G^T]^T$ is a spatial momentum comprised of the system's net linear momentum $\boldsymbol{l}_G \in \mathbb{R}^3$ and angular momentum $\boldsymbol{k}_G \in \mathbb{R}^3$ about the Center of Mass (CoM). The centroidal momentum is given by the relationship

$$\boldsymbol{h}_G = \boldsymbol{A}_G(\boldsymbol{q})\,\dot{\boldsymbol{q}} \tag{13}$$

where $\boldsymbol{A}_G$ is known as the Centroidal Momentum Matrix [9]. While this looks like a Jacobian relationship, the system's net *angular* momentum does not admit any function $\boldsymbol{g}(\boldsymbol{q})$ with Jacobian $\boldsymbol{J}_g = \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{q}}$ such that $\boldsymbol{k}_G = \boldsymbol{J}_g\,\dot{\boldsymbol{q}}$. This result is due to the fact that the conservation of angular momentum imposes a nonholonomic constraint [19]. Methods to compute $\boldsymbol{A}_G$ and $\dot{\boldsymbol{A}}_G\dot{\boldsymbol{q}}$ are given in [1] and [9].

III. Conic Formulation of Prioritized Optimization

This section provides a reformulation of the PTSC problem as a conic optimization problem. This reformulation results in a reduced number of variables, reduced number of constraints, and is general to periods of flight as well as support. This reduction allows our reformulation to be solved about twice as fast as (10) for the examples considered. Suppose that the task matrix $\boldsymbol{A}_t$ contains all possible tasks regardless of priority. The error in achieving a commanded task dynamics is given by:

$$\boldsymbol{e} = \dot{\boldsymbol{r}}_{t,c} - \dot{\boldsymbol{r}}_t \tag{14}$$

$$= \dot{\boldsymbol{r}}_{t,c} - \boldsymbol{A}_t\ddot{\boldsymbol{q}} - \dot{\boldsymbol{A}}_t\dot{\boldsymbol{q}}\ . \tag{15}$$

Solving (2) for $\ddot{\boldsymbol{q}}$, the task dynamics error $\boldsymbol{e}$ obeys:

$$\boldsymbol{b}_t = \boldsymbol{A}_t\boldsymbol{H}^{-1}\boldsymbol{S}_a^T\boldsymbol{\tau} + \boldsymbol{A}_t\boldsymbol{H}^{-1}\boldsymbol{J}_{sp}^T\boldsymbol{F}_{sp} + \boldsymbol{e} \tag{16}$$

where:

$$\boldsymbol{b}_t = \dot{\boldsymbol{r}}_{t,c} + \boldsymbol{A}_t\boldsymbol{H}^{-1}(\boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{G}) - \dot{\boldsymbol{A}}_t\dot{\boldsymbol{q}}\,. \tag{17}$$

In (16), $\boldsymbol{F}_{sp}$ collects all support forces $\boldsymbol{f}_{s_{ij}}$ and $\boldsymbol{J}_{sp}$ is the combined support point Jacobian. This support point Jacobian relates $\dot{\boldsymbol{q}}$ to the linear velocity of all the support vertices and differs from the support Jacobian $\boldsymbol{J}_s$ which related $\dot{\boldsymbol{q}}$ to the linear and angular velocity of all support bodies. We define:

$$\boldsymbol{\Lambda}_{t\tau}^{-1} = \boldsymbol{A}_t\boldsymbol{H}^{-1}\boldsymbol{S}_a^T \quad \text{and} \tag{18}$$

$$\boldsymbol{\Lambda}_{ts}^{-1} = \boldsymbol{A}_t\boldsymbol{H}^{-1}\boldsymbol{J}_{sp}^T \tag{19}$$

with symbols $\boldsymbol{\Lambda}^{-1}$ since each of these is a cross-coupling inverse operational-space (task-space) inertia matrix [20].

Here, the PTSC problem with K levels of prioritization is solved with a series of K conic optimization problems. At each level $k \in \{1, \ldots, K\}$ it is assumed that the subset of the tasks to be optimized is encoded in a task optimization selector matrix $\boldsymbol{S}_{o,k}$. Each row of $\boldsymbol{S}_{o,k}$ is a unit vector that selects a single task error, and the number of rows corresponds to the number of tasks concurrently optimized at level k. The optimal error from the previous level is defined as $\boldsymbol{e}_{k-1}^*$. A similar task constraint selector $\boldsymbol{S}_{c,k}$ selects those tasks from all previous levels, as well as any additional hard constraints. The problem for priority level k is then:

$$\min_{\boldsymbol{\tau},\boldsymbol{F}_{sp},\boldsymbol{e}_k,z} \quad z \tag{20}$$

$$\text{s.t.} \quad \boldsymbol{\Lambda}_{t\tau}^{-1}\,\boldsymbol{\tau} + \boldsymbol{\Lambda}_{ts}^{-1}\,\boldsymbol{F}_{sp} + \boldsymbol{e}_k = \boldsymbol{b}_t$$

$$\|\boldsymbol{S}_{o,k}\,\boldsymbol{e}_k\| \le z \tag{21}$$

$$\boldsymbol{S}_{c,k}\,\boldsymbol{e}_k = \boldsymbol{S}_{c,k}\,\boldsymbol{e}_{k-1}^* \tag{22}$$

$$\boldsymbol{f}_{s_{ij}} \in \boldsymbol{\mathcal{C}}_i \tag{23}$$

$$\underline{\boldsymbol{\tau}} \le \boldsymbol{\tau} \le \overline{\boldsymbol{\tau}}\,.$$

Minimization of the scalar z in (20) results in a minimization of error for the current task dynamics to be optimized due to constraint (21). The incorporation of z also provides a linear objective, which is required for the conic solver used here. The constraint (22) ensures that the optimal errors for the higher priority tasks are not corrupted. Table I describes the entire algorithm for this series of optimization problems in more explicit detail. Note that for application to position controlled systems, an optimal $\ddot{\boldsymbol{q}}$ can be obtained by solving (2) or through a forward dynamics computation.

Inputs:

Task Dynamics Descriptors: $\boldsymbol{A}_t,\ \dot{\boldsymbol{A}}_t\dot{\boldsymbol{q}},\ \dot{\boldsymbol{r}}_{t,c}$
System Dynamics Descriptors: $\boldsymbol{H},\ \boldsymbol{C}\dot{\boldsymbol{q}},\ \boldsymbol{G},\ \boldsymbol{S}_a,\ \boldsymbol{J}_{sp}$
Task Priority Descriptors: $\boldsymbol{e}_0^*,\ \boldsymbol{S}_{c,1},\ K$
$\boldsymbol{S}_{o,k} \quad \forall\, k \in \{1, \ldots, K\}$

Algorithm:

$\boldsymbol{\Lambda}_{t\tau}^{-1} := \boldsymbol{A}_t\boldsymbol{H}^{-1}\boldsymbol{S}_a^T$
$\boldsymbol{\Lambda}_{ts}^{-1} := \boldsymbol{A}_t\boldsymbol{H}^{-1}\boldsymbol{J}_{sp}^T$
$\boldsymbol{b}_t := \dot{\boldsymbol{r}}_{t,c} + \boldsymbol{A}_t\boldsymbol{H}^{-1}(\boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{G}) - \dot{\boldsymbol{A}}_t\dot{\boldsymbol{q}}$
for $k = 1$ **to** K **do**
  $(\boldsymbol{\tau}^*, \boldsymbol{F}_{sp}^*, \boldsymbol{e}_k^*, z^*) := \arg\min_{\boldsymbol{\tau},\boldsymbol{F}_{sp},\boldsymbol{e}_k,z}(20)$
  $\boldsymbol{S}_{c,(k+1)} := [\boldsymbol{S}_{c,k}^T\ \boldsymbol{S}_{o,k}^T]^T$
end for k

Output:

$\boldsymbol{\tau} := \boldsymbol{\tau}^*$

TABLE I
Prioritized Task-Space Control (PTSC) Algorithm

The advantages of this strategy over the QP in (10) is multi-pronged. First, the variables $\ddot{\boldsymbol{q}}$ are eliminated in favor of task error $\boldsymbol{e}$. Since at each level only a few tasks are optimized, this reduces the number of optimization variables, and simplifies the objective. Salini [18] eliminates $\ddot{\boldsymbol{q}}$ in an alternative QP formulation, but does not include $\boldsymbol{e}$, resulting in a dense objective function Hessian. As a result, the formulation of [18] and the QP here were found to perform comparably. Second, polygonal approximations to the friction cones, which grow in complexity as the fidelity of the approximation is increased, are replaced by a single

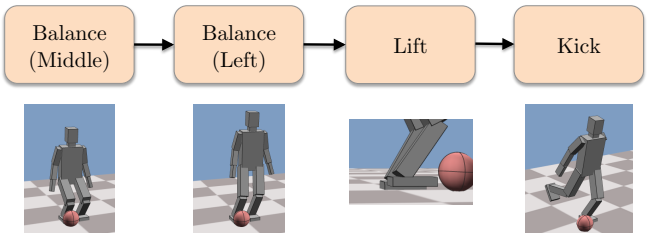

Fig. 5. State machine used for Kick control. All states use the task priorities (1) Feet, (2) Linear Momentum, (3) Pose & Centroidal Angular Momentum. All state transitions are based on time in this example.

constraint per force $\boldsymbol{f}_{s_{ij}}$. For a 4-sided polygonal approximation, the use of friction cones was not found to have any substantial effect on the computation times or simulation results. Beyond a 4-sided polygonal approximation, it was found to be advantageous to use cone constraints to enable faster solutions. Solution of all optimization formulations was provided by the interior point optimizer in MOSEK [21]. This optimizer employs a primal-dual method that handles the cone constraints (21) and (23) directly and efficiently.

IV. Application to the Control of Dynamic Behaviors

A. Model and Simulation

The model used in this work is a 26 DoF (20 actuated DoFs) humanoid as shown in Fig. 3. Spherical joints are modeled at the hips and shoulders, providing 6 DoFs in each leg, and 4 DoFs in each arm. Degrees of freedom in the hands, wrists, and head are not modeled. The mass distribution to each segment is modeled after a 50-th percentile male [22], with segment dimensions based on the model presented in [23]. Inertia tensors are estimated based on a simple equidensity mass distribution for each segment.

Full 3D simulation of the system is carried out using the DynaMechs [24] simulation package. This package provides an extremely efficient implementation of the articulated-body algorithm [25] with recursive steps of the algorithm optimized for each type of joint. Contact dynamics are simulated with a penalty-based spring damper model. No force feedback or information about the environment (e.g. surface normal or surface height) is provided to the controller.

B. Control of a Dynamic Kick

As a first example, PTSC is used along with a state machine, shown in Fig. 5, to control a dynamic kicking motion. During all states, the task prioritization (1) Feet, (2) Centroidal Linear Momentum, (3) Pose and Centroidal Angular Momentum is employed. Pose control helps to prevent configuration drift. The details of how the commanded task dynamics $\dot{\boldsymbol{r}}_{t,c}$ are specified is described below.

Foot control commands linear and angular acceleration of the foot: $\dot{\boldsymbol{r}}_c = [\dot{\boldsymbol{\omega}}_c^T \ \ddot{\boldsymbol{p}}_c^T]^T$. This command is set to zero for feet in support. When the foot is in the air, these rates are selected based on a position/orientation PD scheme:

$$\dot{\boldsymbol{\omega}}_c = \dot{\boldsymbol{\omega}}_d + \boldsymbol{K}_{D,\omega}(\boldsymbol{\omega}_d - \boldsymbol{\omega}) + \boldsymbol{K}_{P,\omega}\boldsymbol{e}_\theta \tag{24}$$

$$\ddot{\boldsymbol{p}}_c = \ddot{\boldsymbol{p}}_d + \boldsymbol{K}_{D,p}(\dot{\boldsymbol{p}}_d - \dot{\boldsymbol{p}}) + \boldsymbol{K}_{P,p}(\boldsymbol{p}_d - \boldsymbol{p}) \tag{25}$$

where $\boldsymbol{e}_\theta \in \mathbb{R}^3$ is an angle-axis representation of error between a desired and actual orientation, as used in [4]. Desired positions and orientations are derived from hand-authored motion, such as the kick trajectory in the kick state.

For Centroidal Momentum, a rate of change $\dot{\boldsymbol{r}}_c = [\dot{\boldsymbol{k}}_{G,c}^T \ \dot{\boldsymbol{l}}_{G,c}^T]^T$ is commanded separately for linear and angular momentum. For linear momentum, this command is from PD control on the desired CoM (G):

$$\dot{\boldsymbol{l}}_{G,c} = m[\ddot{\boldsymbol{p}}_{G,d} + \boldsymbol{K}_{D,\ell}(\dot{\boldsymbol{p}}_{G,d} - \dot{\boldsymbol{p}}_G) + \boldsymbol{K}_{P,\ell}(\boldsymbol{p}_{G,d} - \boldsymbol{p}_G)] \tag{26}$$

where $\boldsymbol{p}_G$ is the CoM position and m is the total mass of the system. The commanded rate of change in angular momentum takes a simpler form:

$$\dot{\boldsymbol{k}}_{G,c} = \dot{\boldsymbol{k}}_{G,d} + \boldsymbol{K}_{D,k}(\boldsymbol{k}_{G,d} - \boldsymbol{k}_G)\,. \tag{27}$$

All of the balance states and the lift state employ $\dot{\boldsymbol{k}}_{G,d} = \boldsymbol{k}_{G,d} = \boldsymbol{0}$ which provides a dampening of any excess angular momentum.

To achieve pose control, joint accelerations are commanded for actuated joints and the torso orientation. For all examples, this commanded acceleration takes the form of a PD law to a static nominal pose. For revolute joints:

$$\ddot{q}_{i,c} = \ddot{q}_{i,d} + K_{D,i}(\dot{q}_{i,d} - \dot{q}_i) + K_{P,i}(q_{i,d} - q_i),$$

where $\dot{q}_{i,d} = \ddot{q}_{i,d} = 0$ in all the examples here. For spherical joints and orientation of the torso, the law (24) is employed. Since the pose task is optimized at the last level of control, it is generally not possible to fulfill all the desired pose dynamics. Weighting factors are employed that promote closer tracking on certain joints than others. For instance, the shoulder and elbow are given lower weight to promote arm action in the resultant motion. These weights can be incorporated by replacing $\boldsymbol{e}$ with $\boldsymbol{W}\boldsymbol{e}$ in (16) for an appropriate diagonal weighting matrix $\boldsymbol{W}$.

Simple spline trajectories are used throughout to generate the desired dynamic motions. Cubic spline trajectories on the CoM and right foot are used to generate the desired motions for the Balance and Lift states. In the kick state, the foot is commanded to move in an arc centered at the initial right hip position. The orientation of the right foot is commanded to remain tangent to this arc. A series of cubic splines on the desired right virtual leg angle θ_d are used to produce the desired 3D accelerations of the right foot using standard formula that relate accelerations in polar coordinates to cartesian coordinates.

During a kick, the system predominantly rotates about the stance hip, resulting in non-zero centroidal angular momentum. For the example shown, the inertial z-axis is opposite gravity and the y-axis is perpendicular to the sagittal plane. To promote a whole-body rotation about the y-axis, the system's net moment of inertia about the y-axis $I_{yy,0}$ is recorded at the beginning of the kick motion and desired centroidal angular momentum and rates are selected as:

$$\boldsymbol{k}_{G,des} = [0,\ \gamma\, I_{yy,0}\, \dot{\theta}_d, 0]^T,$$
$$\dot{\boldsymbol{k}}_{G,des} = [0,\ \gamma\, I_{yy,0}\, \ddot{\theta}_d,\ 0]^T.$$

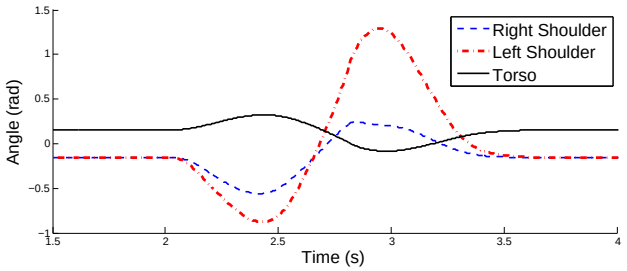

Fig. 6. Shoulder and torso angles during the Kick state of the kick motion. Shoulder angles are measured in the sagittal plane and are taken relative to the torso. The torso angle is measured in the sagittal plane and relative to the inertial coordinate system.

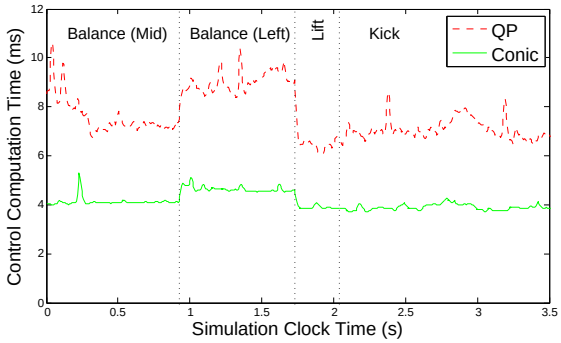

Fig. 7. Control computation times for PTSC applied to a kick motion with a QP versus a conic formulation. State transitions are shown by the vertical dotted lines, with the state denoted at the top of each corresponding section.

Here $\gamma = 0.8$ is a factor that accounts for the stance leg remaining stationary. While crude, these desired centroidal momentum dynamics are some of the first non-trivial ones to be designed in the literature and are sufficient to produce rich motion. For a video of the kick motion, please see the attachment to this paper or view it at the link below.

`http://www.go.osu.edu/Wensing_Orin_ICRA2013`

Without authoring any upper-body trajectories, the control approach yields complex upper-body motion. Shoulder and torso angles during the kick state are shown in Fig. 6. During the kick motion, the left shoulder shows a larger displacement. This behavior emerges from the controller to regulate the angular momentum about the global vertical z-axis in response to the right foot's kick trajectory.

The conic formulation of PTSC was found to be much faster than the QP formulation for this example. As shown in Fig. 7, the conic formulation (20) was able to be solved in nearly half the time as the QP formulation (10) (in 55% of the time on average). A 4-sided polygonal approximation to the friction cone was employed for the QP formulation. Thus, the majority of the improved speed results here can be attributed to the variable reductions and simplified objective due to the use of $\boldsymbol{e}$. The times shown in this graph include the computation time of all quantities required by each algorithm ($\boldsymbol{H}$, $\dot{\boldsymbol{A}}_t\dot{\boldsymbol{q}}$, $\boldsymbol{\Lambda}_{ts}^{-1}$, etc.) as well as the optimization time of the solver. All computational experiments were run on a 2.3 GHz Intel Core i5 MacBook Pro.

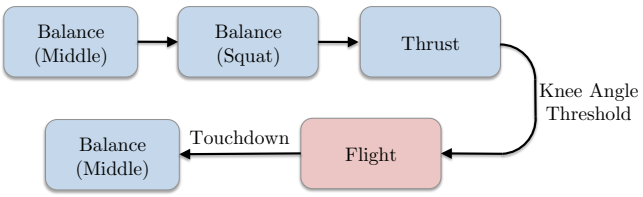

Fig. 8. State machine used for Jump control. States shown in blue use task priorities (1) Feet, (2) Centroidal Momentum, (3) Pose. The state in red (Flight) uses task priorities (1) Feet, (2) Pose. State transition criteria are noted on the transition arrows, where an omission of a criterion indicates a transition that takes place based on time.

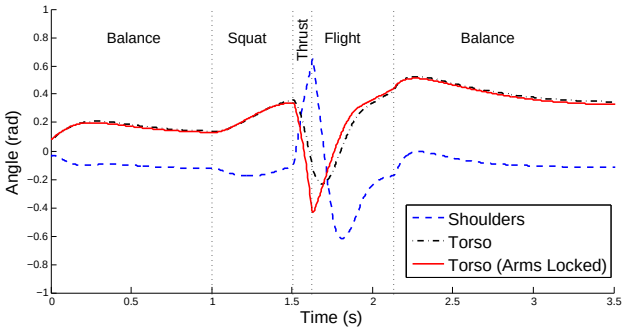

Fig. 9. Shoulder and torso angles during the jump motion. State transitions are shown by the vertical dotted lines, with the state denoted at the top of each corresponding section. Angles are measured the same as in Fig. 6. The angle of the torso is also shown with arms locked, and results in greater torso pitch during Thrust.

C. Control of a Standing Broad Jump

The PTSC framework was also applied to produce a standing broad jump using the state machine shown in Fig. 8. All states during stance use the task priorities (1) Feet, (2) Centroidal Momentum, (3) Pose. Centroidal momentum control is omitted during flight, as $\dot{\boldsymbol{h}}_G$ is equal to the net gravity force in the absence of GRFs.

The details of the task dynamics commanded by the jump state machine are very similar to those in [8]. During the squat state, the desired CoM is lowered though a cubic spline trajectory. At the beginning of the thrust state, the desired CoM position and velocity is discontinuously incremented in the desired direction of the jump. This results in the selection of joint torques that will result in an impulsive ground force on the system. As the legs approach full extension, a knee angle threshold triggers transition to the flight state. In flight, the foot trajectories commanded are largely ballistic, but modified smoothly to position the feet forward for landing. Space limitations prevent the precise details of these trajectories from being included. At touchdown, the PD gains on the CoM are softened to provide a smooth landing, with a desired CoM placed over the middle of the support.

The video attachment to this paper shows the use of the jump controller in a number of different environments. Note that all jumps display significant arm action forward and backward to maintain balance and to help position the feet before landing. This is a feature of biological long jumps [26] that was largely not present in previous work

where hand authored upper-body motions were used [27]. The torso and shoulder angles for the level terrain jump are shown in Fig. 9. During thrust, the arms are swung upwards to prevent the torso from pitching backwards. If the arms are forced to remain locked, the lack of this behavior results in excess torso pitch as shown.

The jump controller for level terrain is applied without modification for a jump onto uneven terrain. Once the system lands, no knowledge of the terrain is assumed, and the feet are simply commanded to not accelerate. Since the control approach used here always attempts to push off the ground, corners of the foot that are originally not in contact are quickly pushed into contact with the unknown terrain. Based on these experiments, it appears that force feedback may not be required for reasonably stiff terrain.

In a final video demonstration, we show the performance of the jump controller when landing on a slippery surface. In this demonstration, the controller assumes that the ground has a coefficient of friction of $\mu = 0.6$, but the simulation employs a coefficient of $\mu = 0.4$. While the system does experience foot slip due to this inaccuracy, the balance controller is robust to this disturbance and results in non-authored arm-windmilling to maintain balance. The desired CoM is constantly updated in the example to remain over the middle of the support polygon.

V. Conclusion

Prioritized Task-Space Control provides a convenient framework in which to characterize dynamic behaviors for humanoid systems. Through consideration of the constraints on ground reaction forces, motions for level terrain are able to be easily adapted to uneven terrain scenarios. Reformulation of previous QP formulations for PTSC allows speed gains to be achieved while addressing friction constraints in their full complexity with conic optimization. This reformulation enables real-time control rates of 200 Hz. These algorithms are general to control quantities such as the system's net angular momentum, which leads to rich upper-body behaviors that are not authored by the designer. Future work will focus on addressing joint limits and self-collision as well as analysis of generated torque data for both QP and conic formulations. Additionally, the use of these methods to support higher-level motion planners provides an exciting direction of future work that may provide humanoids with a broader vocabulary of dynamic maneuverability.

VI. Acknowledgments

This work was supported by a National Science Foundation Graduate Research Fellowship to Patrick Wensing, and by Grant No. CNS-0960061 from the NSF with a subaward to The Ohio State University. Special thanks to Yiping Liu for assistance with the simulation videos. We also wish to thank the reviewers for their helpful comments.

References

[1] A. Macchietto, V. Zordan, and C. R. Shelton, "Momentum control for balance," *ACM Trans. Graph.*, vol. 28, pp. 80:1–80:8, July 2009.

[2] S.-H. Lee and A. Goswami, "Fall on backpack: Damage minimizing humanoid fall on targeted body segment using momentum control," in *ASME Int. Design Engineering Tech. Conf.*, pp. 47153:1–10, 2011.

[3] K. Yamane and J. Hodgins, "Control-aware mapping of human motion data with stepping for humanoid robots," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 726–733, Oct. 2010.

[4] M. Da Silva, Y. Abe, and J. Popović, "Simulation of human motion data using short-horizon model-predictive control," *Computer Graphics Forum*, vol. 27, no. 2, pp. 371–380, 2008.

[5] G. Bin Hammam, D. Orin, and B. Dariush, "Whole-body humanoid control from upper-body task specifications," in *IEEE Int. Conf. on Robotics and Automation*, pp. 3398–3405, May 2010.

[6] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal of Robotics and Automation*, vol. 3, pp. 43–53, Feb. 1987.

[7] J. Park and O. Khatib, "Contact consistent control framework for humanoid robots," in *Proc. of the IEEE Int. Conference on Robotics and Automation*, pp. 1963–1969, May 2006.

[8] M. de Lasa, I. Mordatch, and A. Hertzmann, "Feature-based locomotion controllers," in *ACM SIGGRAPH 2010 papers*, vol. 29, (New York, NY, USA), pp. 131:1–10, ACM, July 2010.

[9] D. E. Orin and A. Goswami, "Centroidal momentum matrix of a humanoid robot: Structure and properties," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 653–659, Sept. 2008.

[10] M. Mistry, J. Buchli, and S. Schaal, "Inverse dynamics control of floating base systems using orthogonal decomposition," in *IEEE Int. Conf. on Robotics and Automation*, pp. 3406–3412, May 2010.

[11] L. Righetti, J. Buchli, M. Mistry, and S. Schaal, "Control of legged robots with optimal distribution of contact forces," in *IEEE-RAS Int. Conf. on Humanoid Robots*, pp. 318–324, Oct. 2011.

[12] K. Yamane and Y. Nakamura, "Dynamics filter - concept and implementation of online motion generator for human figures," *IEEE Trans. on Robotics and Automation*, vol. 19, pp. 421–432, June 2003.

[13] J. Park, J. Haan, and F. Park, "Convex optimization algorithms for active balancing of humanoid robots," *IEEE Transactions on Robotics*, vol. 23, pp. 817–822, Aug. 2007.

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

[15] J. Salini, V. Padois, and P. Bidaud, "Synthesis of complex humanoid whole-body behavior: A focus on sequencing and tasks transitions," in *IEEE Int. Conf. on Rob. and Automation*, pp. 1283–1290, 2011.

[16] R. Featherstone and D. Orin, "Chapter 2: Dynamics," in *Springer Handbook of Robotics* (B. Siciliano and O. Khatib, eds.), New York: Springer, 2008.

[17] N. Mansard, "A dedicated solver for fast operational-space inverse dynamics," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 4943–4949, May 2012.

[18] J. Salini, *Dynamic control for the task/posture coordination of humanoids : toward synthesis of complex activities*. PhD thesis, Pierre and Marie Curie University, 2012.

[19] Y. Nakamura and R. Mukherjee, "Nonholonomic path planning of space robots," in *IEEE Int. Conf. on Robotics and Automation*, pp. 1050–1055, May 1989.

[20] P. Wensing, R. Featherstone, and D. Orin, "A reduced-order recursive algorithm for the computation of the operational-space inertia matrix," in *IEEE Int. Conf. on Rob. and Automation*, pp. 4911–4917, 2012.

[21] E. Andersen, C. Roos, and T. Terlaky, "On implementing a primal-dual interior-point method for conic quadratic optimization," *Mathematical Programming*, vol. 95, pp. 249–277, 2003.

[22] D. A. Winter, *Biomechanics and Motor Control of Human Movement*. New York: John Wiley & Sons, 2nd ed., 1990.

[23] S. H. Lee, *Biomechanical Modeling and Control of the Human Body for Computer Animation*. PhD thesis, U. of Cal. Los Angeles, 2008.

[24] S. McMillan, D. E. Orin, and R. B. McGhee, "DynaMechs: an object oriented software package for efficient dynamic simulation of underwater robotic vehicles," in *Underwater Robotic Vehicles: Design and Control*, pp. 73–98, Albuquerque, NM: TSI Press, 1995.

[25] R. Featherstone, *Rigid Body Dynamics Algorithms*. Springer, 2008.

[26] B. M. Ashby and S. L. Delp, "Optimal control simulations reveal mechanisms by which arm movement improves standing long jump performance," *J. of Biomech.*, vol. 39, no. 9, pp. 1726–1734, 2006.

[27] W. Wooten and J. Hodgins, "Simulating leaping, tumbling, landing and balancing humans," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, vol. 1, pp. 656–662, 2000.