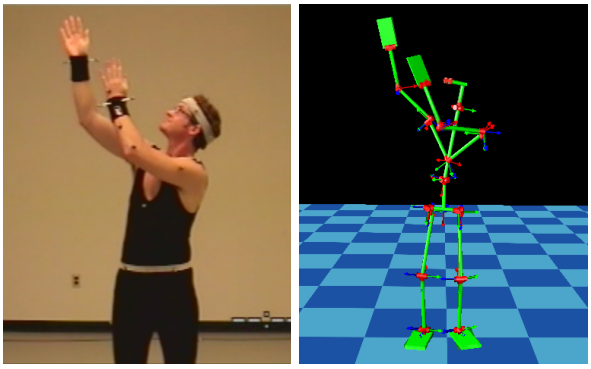
2013 13th IEEE-RAS International Conference on
Humanoid Robots (Humanoids).
October 15 - 17, 2013. Atlanta, GA

Human Motion Tracking Control with Strict Contact Force Constraints for Floating-Base Humanoid Robots

Yu Zheng, Katsu Yamane

***Abstract*— This paper presents a tracking controller considering contact force constraints for floating-base humanoid robots. The goal of the tracking controller is to compute the joint torques such that the robot can imitate given reference motions obtained from, for example, human motion capture data. The technical challenge is that the robot motion depends not only on joint torques but also on contact forces from the environment, which depend on the joint torques and are subject to constraints on friction forces. Therefore, computing the joint torques and associated contact forces typically results in a nonlinear optimization problem with complex constraints. We solve this issue by taking advantage of the property that the motion of the floating base is only affected by contact forces. We can then compute the contact forces and joint torques separately by solving two simple sequential optimization problems. Through dynamics simulations, we demonstrate that the proposed tracking controller successfully enables a humanoid robot to reproduce different human motions, including those with contact state changes.**



(a) (b)

Fig. 1. Example of (a) the original human motion and (b) the simulated robot motion.

I. Introduction

Humanoid robots are expected to work with humans in home and office environments. It would be more desirable for such robots to have human-like motions so that human co-workers can easily infer their intention and predict future movements for safe and smooth interactions.

However, programming humanoid robots is not straightforward because they tend to have complex structures consisting of many joints. A possible solution is to teach the motions through human demonstration as often referred to as learning from demonstration [1] or imitation learning [2]. This approach allows a programmer to simply demonstrate the motion while the robot observes the motion. A learning algorithm then makes adjustments to the motion so that the robot can achieve the task using its own body.

Unfortunately, most of the work based on this approach only considers the kinematics of motions and therefore cannot be directly applied to robots and motions that require balancing, such as standing and walking motions of floating-base humanoid robots. Considering the dynamics in such motions is essential because the six degrees of freedom (DOF) of the translation and rotation of the floating base are not directly actuated. Instead, the corresponding generalized force is provided by contact forces that are subject to inequality constraints on the friction.

Yu Zheng is with the Department of Computer Science, University of North Carolina at Chapel Hill, NC 27599, USA. This work was done when he was with Disney Research Pittsburgh, PA 15213, USA. yuzheng001@gmail.com

Katsu Yamane is with Disney Research Pittsburgh, PA 15213, USA. kyamane@disneyresearch.com

In this paper, we present a controller for floating-base humanoid robots that can track motion capture data while maintaining balance (Fig. 1). Unlike previous work with similar goals, our controller does not include a balance controller based on simplified models.

Our controller consists of two components. The first component is a standard proportional-derivative (PD) controller that computes the desired acceleration to track the given reference trajectory at every DOF, including the six unactuated ones of the floating base.

The second component, which is the main contribution of this paper, computes the optimal joint torques and contact forces to realize the desired accelerations given by the first component, considering the full-body dynamics of the robot and the constraints on contact forces. The desired acclerations may not be feasible for the robot due to the limits in normal contact forces and friction. We decouple the optimization problem into two simple sub-problems by taking advantage of the property that the joint torques do not contribute to the six DOF of the floating base, which allows us to solve the optimization with strict contact force constraints in real time.

We demonstrate the usefulness of our tracking controller in full-body dynamics simulation with two settings. In the first setting, we require a humanoid robot to track choreographic human motions while maintaining both feet on the ground, while the feet in the motion capture data are not perfectly still due to errors in motion capturing and differences between the kinematics of the human subject and the robot. In the second, the robot is required to follow human stepping motions where the two feet lift up and touch down alternately. In all these

978-1-4799-2618-3/13/$31.00 ©2013 IEEE

cases, the robot can successfully track the motion capture data by using the proposed tracking controller.

The rest of this paper is organized as follows. Section II reviews the previous work. Section III gives the basic equations of the full-body dynamics of floating-base robots. The proposed tracking controller is discussed in detail in Section IV. Section V shows the simulation results, followed by concluding remarks and future work in Section VI.

II. Related Work

One of the popular methods for generating humanoid robot motions, especially locomotion, is to first determine a zero moment point (ZMP) trajectory based on given footprints and then compute a physically consistent center of mass (COM) trajectory using a simplified dynamics model, typically an inverted pendulum [3], [4], [5]. The joint trajectories can then be computed by inverse kinematics using the footprints and COM trajectory. However, motions generated by this approach usually do not look human-like, though some work tried to mimic human walking motion by adding single toe support phase and characterized swing leg motions [6]. Another issue is that the ZMP feasibility criterion is only applicable to horizontal terrains. Hirukawa *et al.* [7] and Ott *et al.* [8] used a more general criterion based on feasible contact forces for a walking pattern generator and a balance controller, respectively.

As humanoid robots have similar structures to humans, using human motion capture data to program humanoid robots seems to be an effective way to generate human-like motions. The work [9], [10] mapped human motions to fixed-base humanoid robots considering the kinematic constraints of the robot. Adapting human motion data to the dynamics of floating-base humanoid robots was discussed in [11], [12]. Methods for generating humanoid locomotion based on motion capture data were developed in [13], [14], which modify the extracted joint trajectories according to a replanned ZMP trajectory that ensures the dynamic consistency. Nakaoka *et al.* [15] proposed a method to convert human dancing motions to physically feasible motions for humanoid robots by manually segmenting a motion into motion primitives and designing a controller for each of them. However, these approaches are aimed at offline planning. Some methods can realize online tracking of upper-body motions in the double-support phase while using the lower body for balancing [16], [17]. Yamane and Hodgins [18], [19] presented controllers for humanoid robots to simultaneously track motion capture data and maintain balance.

Using human motion data to generate motion for humanoid characters has also been studied in computer graphics [20], [21], [22], [23], [24]. Nevertheless, those approaches usually employ an extensive optimization process and cannot be applied to realtime control of humanoid robots.

Similarly to the previous work [7], [8], we compute feasible contact forces without using the ZMP criterion for realizing motions on humanoid robots, but our goal is to track human motion capture data rather than walking pattern generation [7] or balance control [8]. Compared to the methods for mapping human motion data to humanoid robots [18], [19], the contribution of this paper is that we can enforce strict constraints on contact forces while achieving real time computation required for controlling robot hardware.

III. Basic Equations

If the robot has N_J actuated joints, the total DOF of the robot is $N_G = N_J + 6$ including the six unactuated DOF of the translation and rotation of the floating base. We let $\boldsymbol{q} \in \mathbb{R}^{N_G}$ denote the generalized coordinate that defines the robot configuration and assume that its first six components correspond to the translation and rotation of the floating base. Also let N_C denote the number of links in contact with the environment and $\boldsymbol{w}_i \in \mathbb{R}^6$ $(i = 1, 2, \ldots, N_C)$ the contact wrench (force and moment) applied to the i-th contact link by the environment.

The equation of motion of the robot can be written as

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{c} = \boldsymbol{N}^T\boldsymbol{\tau} + \boldsymbol{J}^T\boldsymbol{w} \tag{1}$$

where $\boldsymbol{M} \in \mathbb{R}^{N_G \times N_G}$ is the joint-space inertia matrix of the robot, $\boldsymbol{c} \in \mathbb{R}^{N_G}$ is the sum of Coriolis, centrifugal and gravity forces, and $\boldsymbol{w} = \begin{bmatrix}\boldsymbol{w}_1^T & \boldsymbol{w}_2^T & \ldots & \boldsymbol{w}_{N_C}^T\end{bmatrix}^T \in \mathbb{R}^{6N_C}$. Matrix $\boldsymbol{N} \in \mathbb{R}^{N_J \times N_G}$ maps the joint torques into the generalized forces. Since the floating base is unactuated, $\boldsymbol{N}$ has the form

$$\boldsymbol{N} = \begin{bmatrix}\boldsymbol{0}_{N_J \times 6} & \boldsymbol{I}_{N_J \times N_J}\end{bmatrix}. \tag{2}$$

Matrix $\boldsymbol{J} \in \mathbb{R}^{6N_C \times N_G}$ is the contact Jacobian matrix whose transpose maps the contact wrenches into the generalized forces and has the form

$$\boldsymbol{J} = \begin{bmatrix}\boldsymbol{J}_1^T & \boldsymbol{J}_2^T & \ldots & \boldsymbol{J}_{N_C}^T\end{bmatrix}^T \tag{3}$$

where $\boldsymbol{J}_i \in \mathbb{R}^{6 \times N_G}$ is the Jacobian matrix of the i-th contact link's position and orientation with respect to the generalized coordinates. Let $\dot{\boldsymbol{r}}_i \in \mathbb{R}^6$ denote the linear and angular velocities of the i-th contact link and $\ddot{\boldsymbol{r}}_i$ the accelerations. Then, the relationship between $\dot{\boldsymbol{r}}_i$ and the generalized velocity $\dot{\boldsymbol{q}}$ can be written as

$$\dot{\boldsymbol{r}}_i = \boldsymbol{J}_i\dot{\boldsymbol{q}}. \tag{4}$$

Differentiating (4), we obtain the relationship between the joint and Cartesian accelerations as

$$\ddot{\boldsymbol{r}}_i = \boldsymbol{J}_i\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_i\dot{\boldsymbol{q}}. \tag{5}$$

IV. Motion Tracking Control

A. Overview

Figure 2 shows an overview of our motion tracking controller consisting of two main steps in the dashed box. The first step computes the contact wrenches via the computation of contact forces that satisfy the friction constraint and realize the desired reference motion of the floating base as much as possible. The contact points are determined using the desired contact state from the reference motion and the current contact link positions of the actual robot. Based on the computed contact wrenches, the second step determines the joint torques that realize all joint motions and especially the desired contact link motion.

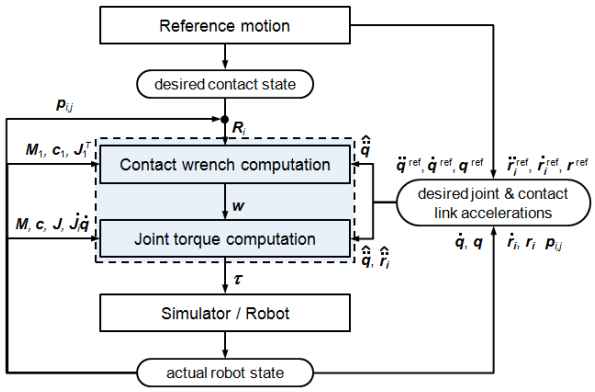


Fig. 2. Overview of the motion tracking controller consisting of two main steps in the dashed box: 1) computation of the contact wrenches to realize the desired floating-base motion, and 2) computation of the joint torques to realize the desired full-body motion. The symbols are defined in the text.

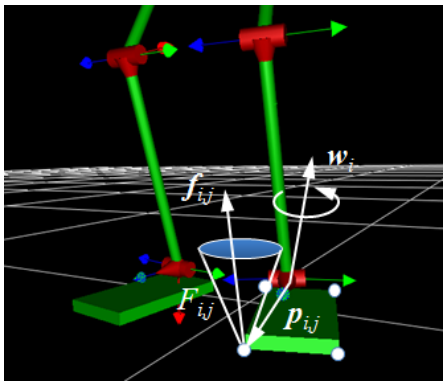


Fig. 3. Illustration of the relationship between the contact wrench applied to a contact link and the contact forces from the environment.

B. Formulation of Contact Wrenches

For a given reference motion, we identify the set of contact candidate links that includes the links that may be in contact with the environment during the motion. The set typically consists of both feet of the robot. At any time during the motion, a contact candidate link has one of the following contact states: face contact, edge contact, point contact, or no contact.

Let $\boldsymbol{p}_{i,j} \in \mathbb{R}^3$ $(j = 1, 2, \ldots, N_i)$ be the vertices of the contact area between the i-th contact link and the environment, and $\boldsymbol{f}_{i,j} \in \mathbb{R}^3$ the contact force at the j-th contact vertex, as depicted in Fig. 3. The number of contact vertices, N_j, is 0, 1, 2, or greater than or equal to 3 in no, point, edge, and face contact states respectively. We assume that the contacts are rigid subject to Coulomb friction model. Therefore, $\boldsymbol{f}_{i,j}$ is a pure force and can be decomposed into three components $f_{i,j1}, f_{i,j2}, f_{i,j3}$ along the normal and two tangential vectors at the contact vertex. The wrench $\boldsymbol{w}_i$ applied to the i-th contact link by the environment is the resultant force and moment from all contact forces $\boldsymbol{f}_{i,j}$ $(j = 1, 2, \ldots, N_i)$ and can be written as

$$\boldsymbol{w}_i = \sum_{j=1}^{N_i} \boldsymbol{R}_{i,j} \boldsymbol{f}_{i,j} = \boldsymbol{R}_i \boldsymbol{f}_i \tag{6}$$

where $\boldsymbol{R}_i = [\boldsymbol{R}_{i,1} \ \boldsymbol{R}_{i,2} \ \ldots \ \boldsymbol{R}_{i,N_i}] \in \mathbb{R}^{6\times 3N_i}$, $\boldsymbol{f}_i = \left[\boldsymbol{f}_{i,1}^T \ \boldsymbol{f}_{i,2}^T \ \ldots \ \boldsymbol{f}_{i,N_i}^T\right]^T \in \mathbb{R}^{3N_i}$, and $\boldsymbol{R}_{i,j} \in \mathbb{R}^{6\times 3}$ is the matrix that maps $\boldsymbol{f}_{i,j}$ into the force and moment around the local frame of the i-th contact link in which $\boldsymbol{w}_i$ is expressed. Here, we do not consider slipping contact, so each contact force $\boldsymbol{f}_{i,j}$ must satisfy the friction constraint given by

$$F_{i,j} = \left\{ \boldsymbol{f}_{i,j} \in \mathbb{R}^3 \mid f_{i,j1} \geq 0, \ \sqrt{f_{i,j2}^2 + f_{i,j3}^2} \leq \mu_i f_{i,j1} \right\}. \tag{7}$$

The controller requires the contact vertex positions and the contact states in order to compute $\boldsymbol{R}_i$. The actual contact vertex positions and states may be different from those in the reference motion. In our implementation, we use the *actual* contact vertex *positions* to compute feasible contact forces at the current pose, while using the contact *states* in the *reference* motion for the reason illustrated by the following example. Consider a case where the right foot is about to touch down. If we use the actual contact state, the optimized COP will stay in the left foot and therefore the right foot may not touch down unless the position tracking is perfect. If we use the contact state in the reference motion, on the other hand, the optimized COP will leave the left foot, which forces the right foot to touch down.

C. Desired Joint and Contact Link Accelerations

The desired accelerations of joints are calculated based on the reference and current positions and velocities as well as the reference accelerations as

$$\hat{\ddot{\boldsymbol{q}}} = \ddot{\boldsymbol{q}}^{ref} + k_d(\dot{\boldsymbol{q}}^{ref} - \dot{\boldsymbol{q}}) + k_p(\boldsymbol{q}^{ref} - \boldsymbol{q}) \tag{8}$$

where $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ are the current joint angle and velocity, $\boldsymbol{q}^{ref}$, $\dot{\boldsymbol{q}}^{ref}$, $\ddot{\boldsymbol{q}}^{ref}$ are the reference joint angle, velocity, and acceleration, and k_p, k_d are proportional and derivative gains.

The desired acceleration of contact candidate link i, $\hat{\ddot{\boldsymbol{r}}}_i$, is determined depending on its desired contact state. If the desired contact state is face contact, $\hat{\ddot{\boldsymbol{r}}}_i = \boldsymbol{0}$. If the desired contact state is no contact, $\hat{\ddot{\boldsymbol{r}}}_i$ is determined using the same control law as $\hat{\ddot{\boldsymbol{q}}}$:

$$\hat{\ddot{\boldsymbol{r}}}_i = \ddot{\boldsymbol{r}}_i^{ref} + k_{dc}(\dot{\boldsymbol{r}}_i^{ref} - \dot{\boldsymbol{r}}_i) + k_{pc}(\boldsymbol{r}_i^{ref} - \boldsymbol{r}_i). \tag{9}$$

where k_{pc} and k_{dc} are the proportional and derivative gains.

If the desired contact state is edge or point contact, we first compute the temporary desired link acceleration $\hat{\ddot{\boldsymbol{r}}}_{i0}$ by (9). We then project $\hat{\ddot{\boldsymbol{r}}}_{i0}$ onto the subspace of link acceleration that satisfies the kinematic constraints of edge or point contact to obtain the desired link acceleration $\hat{\ddot{\boldsymbol{r}}}_{i0}$. If the contact link rotates around an edge, $\ddot{\boldsymbol{r}}_i$ should have the form

$$\ddot{\boldsymbol{r}}_i = \begin{bmatrix} [\boldsymbol{p}_{i,1}\times] \, (\boldsymbol{p}_{i,1} - \boldsymbol{p}_{i,2}) \\ \boldsymbol{p}_{i,1} - \boldsymbol{p}_{i,2} \end{bmatrix} \dot{\omega} \tag{10}$$

where $[\boldsymbol{p}_{i,1}\times] \in \mathbb{R}^{3\times 3}$ is the skew-symmetric matrix representing the cross product of $\boldsymbol{p}_{i,1}$ with another vector and $\dot{\omega} \in \mathbb{R}$ is the angular acceleration of the contact link about the edge. If the contact link rotates about a vertex, $\ddot{\boldsymbol{r}}_i$ should have the form

$$\ddot{\boldsymbol{r}}_i = \begin{bmatrix} [\boldsymbol{p}_{i,1}\times] \\ \boldsymbol{I}_{3\times 3} \end{bmatrix} \dot{\boldsymbol{\omega}} \tag{11}$$

where $\dot{\boldsymbol{\omega}} \in \mathbb{R}^3$ consists of the angular accelerations about the vertex. We project $\hat{\dot{\boldsymbol{r}}}_{i0}$ onto the subspace represented by either (10) or (11) depending on whether the desired contact state is edge or point to obtain $\hat{\dot{\boldsymbol{r}}}_i$.

D. Optimizing the Contact Forces and Joint Torques

The role of the tracking controller is to determine the joint torques such that the robot can replay a given reference motion that may or may not be physically feasible for the robot. In floating-base humanoid robots, the motion depends not only on the joint torques but also the reaction contact forces from the environment, which are also affected by the joint torques. Furthermore, contact forces are subject to the nonlinear friction constraints described by (7). Solving for the contact forces and joint torques simultaneously is computationally expensive and not suitable for realtime control.

In our tracking controller, we decouple the contact forces from the joint torques by taking advantage of the property that joint torques do not affect the motion of the floating base. The tracking controller therefore consists of two steps: 1) optimize the contact forces considering the friction constraints, and 2) optimize the joint torques considering the contact link acceleration constraint.

1) Step 1—Computing contact forces: The first six equations in the full-body motion equation (1) describe the motion of the floating base. From (2) we see that the six equations do not contain joint torques, which corresponds to the fact that the total linear and angular momenta are affected only by external forces. Extracting the first six equations, we obtain

$$\boldsymbol{M}_1 \ddot{\boldsymbol{q}} + \boldsymbol{c}_1 = \boldsymbol{J}_1^T \boldsymbol{w} \tag{12}$$

where $\boldsymbol{M}_1 \in \mathbb{R}^{6\times N_G}$ and $\boldsymbol{J}_1^T \in \mathbb{R}^{6\times 6N_C}$ consist of the first six rows of $\boldsymbol{M}$ and $\boldsymbol{J}^T$, respectively, and $\boldsymbol{c}_1$ comprises the first six components of $\boldsymbol{c}$. Substituting (6) into (12) yields

$$\boldsymbol{M}_1 \ddot{\boldsymbol{q}} + \boldsymbol{c}_1 = \boldsymbol{G}\boldsymbol{f} \tag{13}$$

where $\boldsymbol{G} = [\boldsymbol{G}_1 \ \boldsymbol{G}_2 \ \cdots \ \boldsymbol{G}_{N_C}] \in \mathbb{R}^{6\times 3\sum_{i=1}^{N_C} N_i}$ with $\boldsymbol{G}_i = \boldsymbol{J}_{1i}^T \boldsymbol{R}_i \in \mathbb{R}^{6\times 3N_i}$ and $\boldsymbol{f} = \begin{bmatrix}\boldsymbol{f}_1^T & \boldsymbol{f}_2^T & \cdots & \boldsymbol{f}_{N_C}^T\end{bmatrix}^T = \begin{bmatrix}\boldsymbol{f}_{1,1}^T & \cdots & \boldsymbol{f}_{i,j}^T & \cdots & \boldsymbol{f}_{N_C,N_{N_C}}^T\end{bmatrix}^T \in \mathbb{R}^{3\sum_{i=1}^{N_C} N_i}$. The right-hand side of (13) gives the wrench that is applied to the floating base by the contact forces.

The desired joint acceleration $\hat{\ddot{\boldsymbol{q}}}$ given by (8) may not be feasible due to the limitation of contact forces. In other words, after replacing $\ddot{\boldsymbol{q}}$ with $\hat{\ddot{\boldsymbol{q}}}$ in (13), there may not exist contact forces $\boldsymbol{f}_{i,j} \in F_{i,j}$ that satisfy (13). In order to obtain joint accelerations that are as close as possible to $\hat{\ddot{\boldsymbol{q}}}$, we formulate an optimization problem to compute the contact forces $\boldsymbol{f}_{i,j}$ described by

$$\begin{cases} \text{minimize} & \|\boldsymbol{M}_1 \hat{\ddot{\boldsymbol{q}}} + \boldsymbol{c}_1 - \boldsymbol{G}\boldsymbol{f}\|^2 \\ \text{subject to} & \boldsymbol{f}_{i,j} \in F_{i,j} \text{ for } \forall\ i \text{ and } j \end{cases} \tag{14}$$

where $\|\cdot\|$ denotes the Euclidean norm. The problem (14) has the following geometric meaning. Let V be the set of all wrenches that can be applied to the floating base by the contact forces satisfying the friction constraint, i.e., $V = \{\boldsymbol{G}\boldsymbol{f} \in \mathbb{R}^6 \mid \boldsymbol{f}_{i,j} \in F_{i,j} \text{ for } \forall\ i \text{ and } j\}$. Geometrically, each friction cone $F_{i,j}$ given by (7) is a convex cone. Then, it can be proved that the set V is a convex cone in $\mathbb{R}^6$. The optimization (14) is to compute the minimum Euclidean distance between V and $\boldsymbol{M}_1 \hat{\ddot{\boldsymbol{q}}} + \boldsymbol{c}_1$ in $\mathbb{R}^6$, which can be calculated by the algorithm described in [25]. Also, (14) is a quadratic program with conic constraints, for which some efficient algorithms are available. After solving (14) for the optimized contact forces $\boldsymbol{f}^*$, we can compute the corresponding contact wrenches, $\boldsymbol{w}^*$, using (6).

2) Step 2—Computing joint torques: After computing and substituting the optimized contact wrenches $\boldsymbol{w}^*$ into (1), we determine the joint torques $\boldsymbol{\tau}$. We formulate this step as the following quadratic program

$$\begin{cases} \text{minimize} & \frac{1}{2}(\ddot{\boldsymbol{q}} - \hat{\ddot{\boldsymbol{q}}})^T \boldsymbol{W}_q (\ddot{\boldsymbol{q}} - \hat{\ddot{\boldsymbol{q}}}) + \frac{1}{2}\boldsymbol{\tau}^T \boldsymbol{W}_\tau \boldsymbol{\tau} \\ \text{subject to} & \boldsymbol{M}\ddot{\boldsymbol{q}} - \boldsymbol{N}^T\boldsymbol{\tau} = \boldsymbol{J}^T\boldsymbol{w}^* - \boldsymbol{c}, \\ & \boldsymbol{J}_i \ddot{\boldsymbol{q}} = \hat{\dot{\boldsymbol{r}}}_i - \dot{\boldsymbol{J}}_i \dot{\boldsymbol{q}} \text{ for } \forall i \end{cases} \tag{15}$$

where $\ddot{\boldsymbol{q}}$ and $\boldsymbol{\tau}$ are the $N_V = N_G + N_J$ unknown variables that need to be determined. The cost function comprises the error from the desired joint accelerations and the magnitude of joint torques. The first constraint is the full-body motion equation (1) of the robot, from which we obtain N_G linear equality constraints. After considering the full-body dynamics, therefore, N_J variables among $\ddot{\boldsymbol{q}}$ and $\boldsymbol{\tau}$ are free, which leaves us the freedom to choose or optimize the joint torques for realizing the desired full-body motion. Also, this allows us to add the second constraint, which requires that joint accelerations $\ddot{\boldsymbol{q}}$ and the desired contact link accelerations $\hat{\dot{\boldsymbol{r}}}_i$ to satisfy the relation (5), as long as $6N_C \leq N_J$. If $6N_C > N_J$, then the problem (15) will be overdetermined and may not have a feasible solution. In that case, we can convert the second constraint to a penalty term and add it to the cost function. Furthermore, when a contact candidate link is in the air, we may omit the second constraint for the link because we no longer have to constraint its motion.

We currently deal with the torque limit by adding it as a penalty term in the cost function of (15) rather than as inequality constraints, so that (15) is a quadratic optimization problem with only linear equality constraints and we can derive a closed-form solution. We give the derivation for the case of $6N_C \leq N_J$ as follows, while that for the case of $6N_C > N_J$ is similar.

The cost function of (15) can be rewritten as $\frac{1}{2}(\boldsymbol{x} - \hat{\boldsymbol{x}})^T \boldsymbol{W}(\boldsymbol{x} - \hat{\boldsymbol{x}})$, where $\boldsymbol{x} = \begin{bmatrix}\ddot{\boldsymbol{q}}^T & \boldsymbol{\tau}^T\end{bmatrix}^T \in \mathbb{R}^{N_V}$, $\hat{\boldsymbol{x}} = \begin{bmatrix}\hat{\ddot{\boldsymbol{q}}}^T & \boldsymbol{0}\end{bmatrix}^T \in \mathbb{R}^{N_V}$, and $\boldsymbol{W} = \mathrm{diag}(\boldsymbol{W}_q, \boldsymbol{W}_\tau) \in \mathbb{R}^{N_V \times N_V}$. Also, the constraints of (15) can be integrated as the following linear equations

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{16}$$

where

$$\boldsymbol{A} = \begin{bmatrix}\boldsymbol{M} & -\boldsymbol{N} \\ \boldsymbol{J} & \boldsymbol{0}\end{bmatrix} \in \mathbb{R}^{(N_G + 6N_C)\times N_V}$$

$$\boldsymbol{b} = \begin{bmatrix} \boldsymbol{J}^T\boldsymbol{w}^* - \boldsymbol{c} \\ \hat{\dot{\boldsymbol{r}}}_1 - \dot{\boldsymbol{J}}_1\dot{\boldsymbol{q}} \\ \vdots \\ \hat{\dot{\boldsymbol{r}}}_{N_C} - \dot{\boldsymbol{J}}_{N_C}\dot{\boldsymbol{q}} \end{bmatrix} \in \mathbb{R}^{N_G+6N_C}.$$

Let $\boldsymbol{y} = \boldsymbol{W}^{\frac{1}{2}}(\boldsymbol{x} - \hat{\boldsymbol{x}})$. Then, the cost function of (15) can be further reduced to $\frac{1}{2}\boldsymbol{y}^T\boldsymbol{y}$ and

$$\boldsymbol{x} = \boldsymbol{W}^{-\frac{1}{2}}\boldsymbol{y} + \hat{\boldsymbol{x}}. \tag{17}$$

Substituting (17) into (16) yields

$$\boldsymbol{A}\boldsymbol{W}^{-\frac{1}{2}}\boldsymbol{y} = \boldsymbol{b} - \boldsymbol{A}\hat{\boldsymbol{x}}. \tag{18}$$

Then, the optimal value of $\boldsymbol{y}$ that satisfies (18) and minimizes $\frac{1}{2}\boldsymbol{y}^T\boldsymbol{y}$ can be calculated by

$$\boldsymbol{y}^* = \boldsymbol{W}^{-\frac{1}{2}}\boldsymbol{A}^T\left(\boldsymbol{A}\boldsymbol{W}^{-1}\boldsymbol{A}^T\right)^{-1}(\boldsymbol{b} - \boldsymbol{A}\hat{\boldsymbol{x}}). \tag{19}$$

Substituting (19) into (17), we finally obtain the closed-form solution to (15) as

$$\boldsymbol{x}^* = \boldsymbol{W}^{-1}\boldsymbol{A}^T\left(\boldsymbol{A}\boldsymbol{W}^{-1}\boldsymbol{A}^T\right)^{-1}(\boldsymbol{b} - \boldsymbol{A}\hat{\boldsymbol{x}}) + \hat{\boldsymbol{x}}. \tag{20}$$

V. Simulation Results

A. Simulation Setup

We use the dynamics simulator with rigid-contact model developed by University of Tokyo [26], [27] to conduct our experiments. The humanoid robot model used in the simulations has 25 joints and 31 DOFs including the translation and rotation of the floating base. Each leg has 7 joints (pitch, roll, yaw at both the hip and the ankle and pitch at the knee). We only consider 4 joints in each arm (pitch, roll, yaw at the shoulder and pitch at the elbow) and fix wrist joints. There are 3 joints in the torso. The robot model is about 1.7 meters tall and 65 kg in weight.

We implemented the controller in C++ on a laptop with an Intel Core i7 2.67GHz CPU and 3GB RAM. Average computation times of the whole controller in the following examples are in the range of 1.48–1.61 msec, which is fast enough for realtime control at 500 Hz. Contact force and joint torque optimizations take approximately 24–26% and 31–33% of the time respectively. The rest is spent for computing the other quantities such as the desired accelerations, mass matrix, and Jacobian matrices.

B. Tracking Human Motion Without Contact State Change

In this example, the robot tracks two motion capture clips chosen from CMU Motion Capture Data Library [28], where two actors perform nursery rhyme "I'm a little teapot" while maintaining their feet on the ground, as displayed in Figs. 4a and 5a. The simulated motions are shown in Figs. 4b and 5b and the accompanying video, which demonstrates that the proposed tracking controller enables the robot to reproduce the human motions without falling. For the second case (Fig. 5), we also plot the contact forces and moments on the left foot computed by the tracking controller and the simulated values in Fig. 6, where we can observe that the two values match well in all six components. While not shown, we see similar match at the right foot.

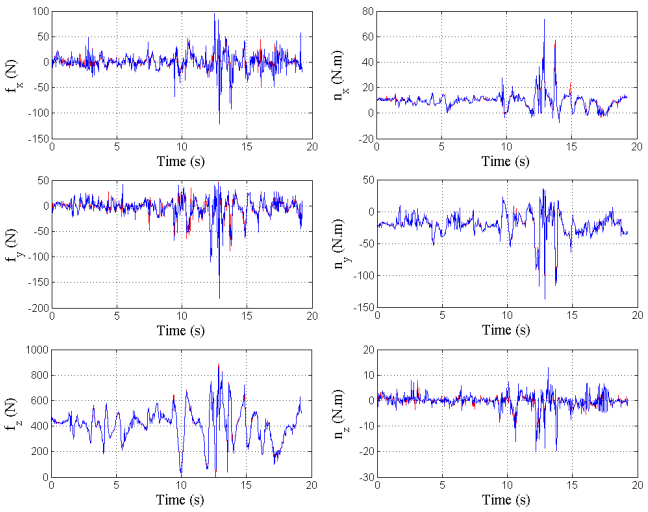


Fig. 6. Optimized (red) and actual (blue) contact forces and moments on the left foot.

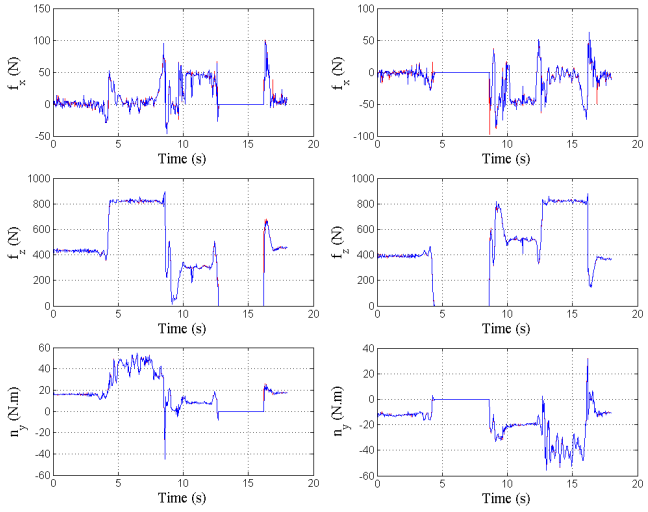


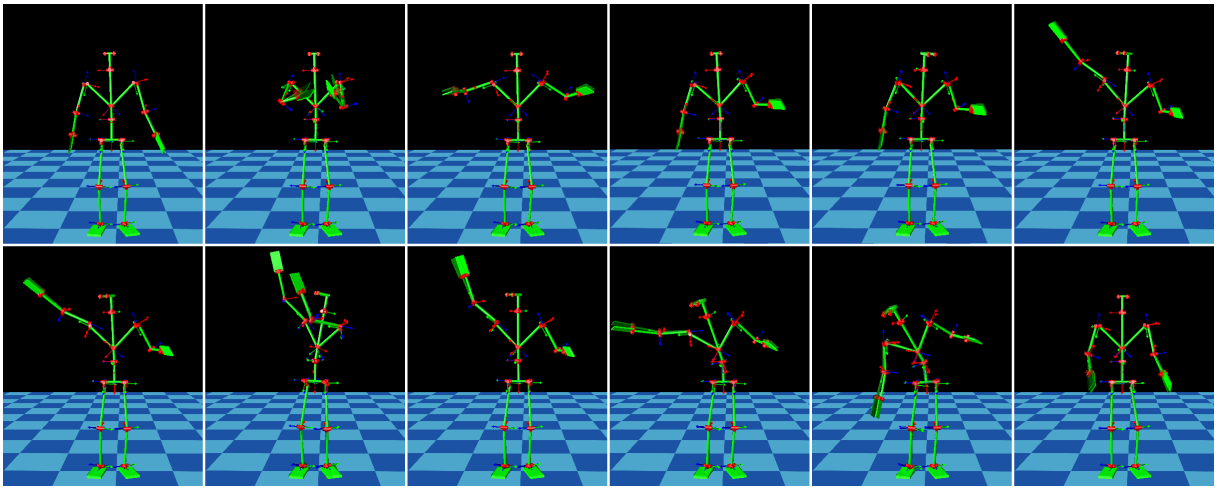
Fig. 9. Optimized (red) and actual (blue) contact forces in x- and z-directions and moments in y-direction on the left (left) and right (right) feet for tracking the stepping motion.

We do note the difference in tracking fidelity between the two examples due to the different styles of the motions. Because the first example is slower and smoother, the robot can track the motion more accurately. In the second example, the robot's pose can be significantly different from the subject's, especially when the subject makes rapid movements.

C. Tracking Human Motion With Contact State Change

In the second example we let the robot follow human's stepping motions. The simulated robot motions are shown in Figs. 7 and 8 and the accompanying video. By our tracking controller, the robot can preserve the style of original human motions. As the side stepping motion is performed in the xz plane of the global frame, the contact forces in x- and z-directions and the moment in y-direction on the feet play a decisive role in the motion, and the optimized and actual values on both feet are exhibited in Fig. 9.

(a)



(b)

Fig. 4. Example of (a) original and (b) simulated motion of "I'm a little teapot" performed by subject 1.

D. Prevent Falling due to Extreme Reference Motions

Only considering the contact force constraints does not generally guarantee that the robot does not tip over, especially if the reference motion is extremely difficult to track. A simple example is trying to maintain a stationary pose where the COM projection is outside of the contact area. In this case, our controller will result in a falling motion with COP in the contact area. This issue is not unique to our controller but applies to any realtime controller for interactive floating-base robots because we cannot predict future reference motion.

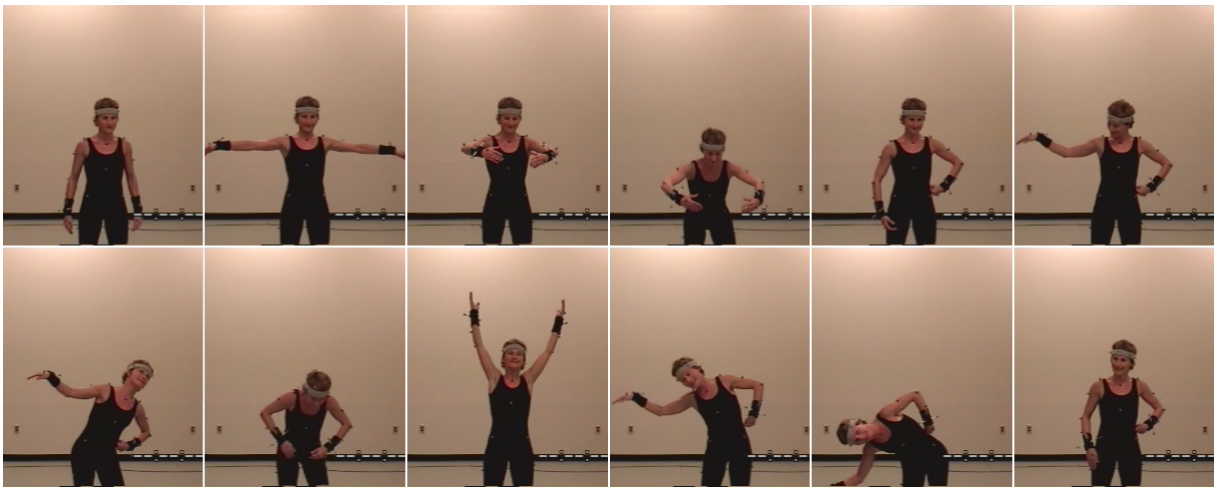
A solution to this issue is to switch or interpolate between two or more reference motions, one being a "safe" reference such as maintaining a static equilibrium pose. Similar idea has been used in an online walking pattern generator that uses joystick input to determine the walking direction [29].

We demonstrate this concept with a simple example using our tracking controller. The main reference motion is a static pose that is not a static equilibrium for robot's mass distribution, as depicted by the transparent model in Fig. 10a. Tracking this motion will eventually cause the robot to fall. We therefore use another reference motion, which is maintaining a static equilibrium pose shown by the opaque model in Fig. 10a. It is also the initial pose for the simulation.

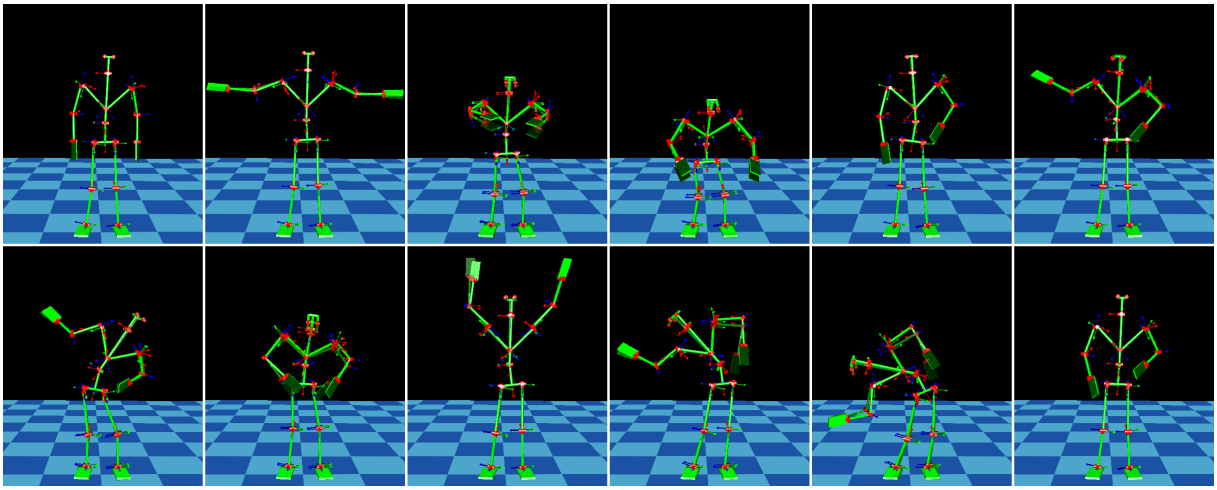
In this example, we interpolate the two motions with weights determined by the time at which the COM is expected to reach the support area boundary computed based on the current position and velocity of the COM. The shorter the time is, the larger the weight for the static equilibrium motion. If the time is below a threshold, we completely switch the reference pose to the static equilibrium one to prevent the robot from falling. In the simulation, the robot finally reaches and maintains the pose shown in Fig. 10b, which is between the two reference poses. Figure 10c shows that the final COP is at the edge of the support area (toe).

VI. Conclusion and Future Work

In this paper, we presented a controller for humanoid robots to track motion capture data. Given the desired accelerations at all DOF to track the reference motion, the proposed tracking controller computes the optimal joint



(a)



(b)

Fig. 5. Example of (a) original and (b) simulated motion of "I'm a little teapot" performed by subject 2.

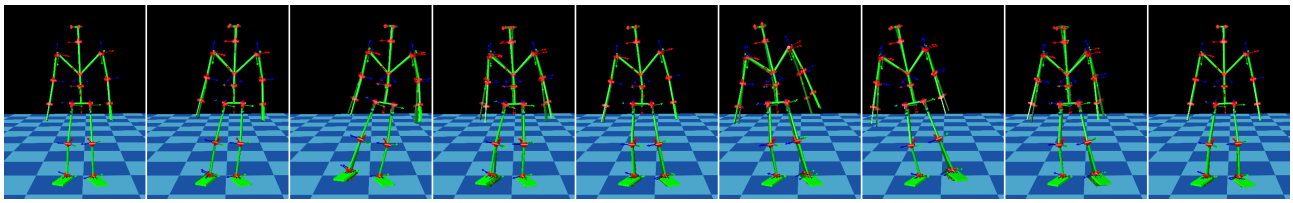


Fig. 7. Simulated side stepping motion.

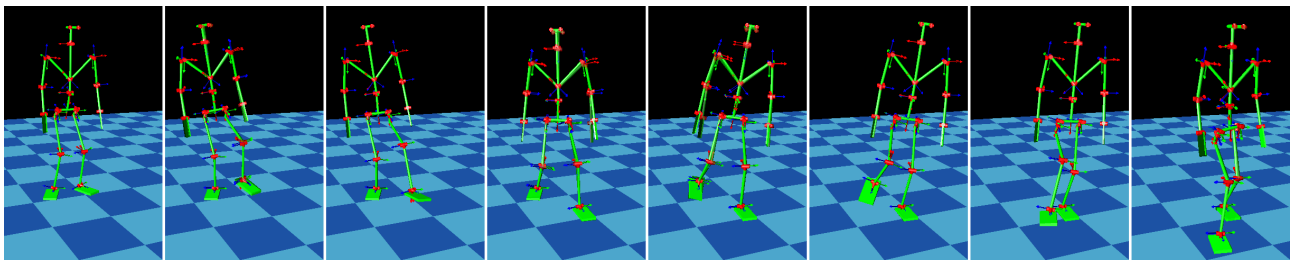
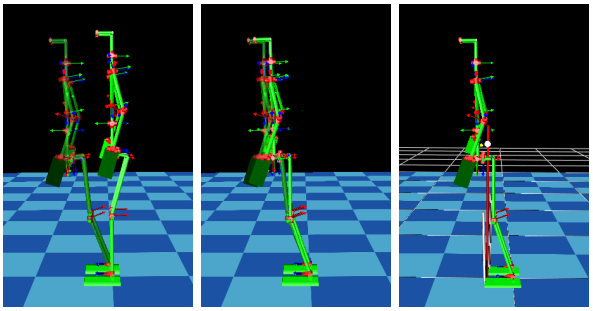


Fig. 8. Simulated forward stepping motion.

torques and contact forces to realize the desired accelerations considering the full-body dynamics of the robot and the constraints on the contact forces. The simulation results show that the tracking controller successfully makes a humanoid



(a) (b) (c)

Fig. 10. Example of preventing falling in tracking extreme motions. (a) Unbalanced reference pose for tracking (transparent) and static equilibrium pose for falling prevention (opaque). (b) Final pose of the robot, which is close to but does not reach the tracking reference pose. (c) The contact forces at the local COP on each foot (white lines) and the total contact force at the global COP (red line), which reaches the boundary of the support area.

robot track various human motions. We also illustrated a simple extension for preventing falling when an extreme motion was given as the reference.

This work can be extended in many directions. A straightforward extension is to motions involving contacts at hands or other links in addition to the feet, which will allow humanoid robots to track a much larger range of human motions. Another possible extension is to motions with more complex contact states and contact link motions. In addition to the rolling contacts formulated in Section IV-C, we can potentially extend the formulation to sliding contacts, where the contact force should be restricted to the boundary of the friction cone and opposite to the sliding direction.

REFERENCES

[1] A. Billard, S. Calinon, R. Dillman, and S. Schaal, "Robot programming by demonstration," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Springer, 2008, pp. 1371–1394.

[2] S. Schaal, A. Ijspeert, and A. Billard, "Computational approaches to motor learning by imitation," *Phylosophical Transactions of the Royal Society of London B: Biological Sciences*, vol. 358, pp. 537–547, 2003.

[3] K. Nagasaka, I. Masayuki, and H. Inoue, "Dynamic walking pattern generation for a humanoid robot based on optimal gradient method," in *Proc. IEEE/RSJ Int. Conf. Syst. Man Cyber.*, Toyko, Japan, 1999, pp. 908–913.

[4] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," in *Proc. IEEE Int. Conf. Robot. Automat.*, Taipei, Taiwan, 2003, pp. 1620–1626.

[5] T. Sugihara, "Simulated regulator to synthesize ZMP manipulation and foot location for autonomous control of biped robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, Pasadena, CA, 2008, pp. 1264–1269.

[6] K. Miura, M. Morisawa, F. Kanehiro, S. Kajita, K. Kaneko, and K. Yokoi, "Human-like walking with toe supporting for humanoids," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, San Francisco, CA, 2011, pp. 4428–4435.

[7] H. Hirukawa, S. Hattori, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, and M. Morisawa, "A universal stability criterion of the foot contact of legged robots - adios zmp," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Orlando, Florida, 2006, pp. 1976–1983.

[8] C. Ott, M. A. Roa, and G. Hirzinger, "Posture and balance control for biped robots based on contact force optimization," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, Bled, Slovenia, 2011, pp. 26–33.

[9] A. Ude, C. Man, M. Riley, and C. Atkeson, "Automatic generation of kinematic models for the conversion of human motion capture data into humanoid robot motion," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, Cambridge, MA, 2000.

[10] A. Safonova, N. Pollard, and J. Hodgins, "Optimizing human motion for the control of a humanoid robot," in *Int. Symp. Adaptive Motion of Animals and Machines*, 2003.

[11] Y. Ikemata, K. Yasuhara, A. Sano, and H. Fujimoto, "Making feasible walking motion of humanoid robots from human motion captured data," in *Proc. IEEE Int. Conf. Robot. Automat.*, Detroit, MI, 1999, pp. 1044–1049.

[12] K. Yamane and Y. Nakamura, "Dynamics filterconcept and implementation of on-line motion generator for human figures," *IEEE Trans. Robot. Automat.*, vol. 19, no. 3, pp. 421–432, 2003.

[13] K. Miura, M. Morisawa, S. Nakaoka, F. Kanehiro, K. Harada, K. Kaneko, and S. Kajita, "Robot motion remix based on motion capture data – towards human-like locomotion of humanoid robots," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, Paris, France, 2009, pp. 596–603.

[14] L. Boutin, A. Eon, S. Zeghloul, and P. Lacouture, "An auto-adaptable algorithm to generate human-like locomotion for different humanoid robots based on motion capture data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Taipei, Taiwan, 2010, pp. 634–639.

[15] S. Nakaoka, A. Nakazawa, K. Yokoi, H. Hirukawa, and K. Ikeuchi, "Generating whole body motions for a biped robot from captured human dances," in *Proc. IEEE Int. Conf. Robot. Automat.*, Taipei, Taiwan, 2003, pp. 3905–3910.

[16] V. Zordan and J. Hodgins, "Motion capture-driven simulations that hit and react," in *Proc. ACM SIGGRAPH Symp. Computer Animation*, San Antonio, TX, 2002, pp. 89–96.

[17] C. Ott, D. Lee, and Y. Nakamura, "Motion capture based human motion recognition and imitation by direct marker control," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, Daejeon, Korea, 2008, pp. 399–405.

[18] K. Yamane and J. Hodgins, "Simultaneous tracking and balancing of humanoid robots for imitating human motion capture data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, St. Louis, 2009, pp. 2510–2517.

[19] ——, "Control-aware mappping of human motion data with stepping for humanoid robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Taipei, Taiwan, 2010, pp. 726–733.

[20] S. Tak, O. Song, and H. Ko, "Motion blanace filtering," *Eurographics 2000, Computer Graphics Forum*, vol. 19, no. 3, pp. 437–446, 2000.

[21] A. Safonova, J. Hodgins, and N. Pollard, "Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces," *ACM Trans. Graphics*, vol. 23, no. 3, pp. 514–521, 2004.

[22] K. Sok, M. Kim, and J. Lee, "Simulating biped behaviors from human motion data," *ACM Trans. Graphics*, vol. 26, no. 3, 2007.

[23] M. da Silva, Y. Abe, and J. Popović, "Interactive simulation of stylized human locomotion," *ACM Trans. Graphics*, vol. 27, no. 3, 2008.

[24] U. Muico, Y. Lee, J. Popović, and Z. Popović, "Contact-aware nonlinear control of dynamic characters," *ACM Trans. Graphics*, vol. 28, no. 3, 2009.

[25] Y. Zheng and C.-M. Chew, "Distance between a point and a convex cone in *n*-dimensional space: computation and applications," *IEEE Transactions on Robotics*, vol. 25, no. 6, pp. 1397–1412, 2009.

[26] K. Yamane and Y. Nakamura, "A numerical robust LCP solver for simulating articulated rigid bodies in contact," in *Robotics: Science and Systems*, 2008.

[27] ——, "Dynamics simulation of humanoid robots: forward dynamics, contact, and experiments," in *The 17th CISM-IFToMM Symposium on Robot Design, Dynamics, and Control*, 2008.

[28] "CMU graphics lab motion capture database," http://mocap.cs.cmu.edu/.

[29] K. Nishiwaki, S. Kagami, Y. Kuniyoshi, M. Inaba, and H. Inoue, "Online generation of humanoid walking motion based on a fast generation method of motion pattern that follows desired zmp," in *Proceedings of the 2002 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, 2002, pp. 2684–2689.