# Controlling Humanoid Robots with Human Motion Data: Experimental Validation

Katsu Yamane, Stuart O. Anderson, and Jessica K. Hodgins

Abstract-Motion capture is a good source of data for programming humanoid robots because it contains the natural styles and synergies of human behaviors. However, it is difficult to directly use captured motion data because the kinematics and dynamics of humanoid robots differ significantly from those of humans. In our previous work, we developed a controller that allows a robot to maintain balance while tracking a given reference motion that does not include contact state changes. The controller consists of a balance controller based on a simplified robot model and a tracking controller that performs local joint feedback and an optimization process to obtain the joint torques to simultaneously realize balancing and tracking. In this paper, we improve the controller to address the issues related to root position/orientation estimation, model uncertainties, and the difference between expected and actual contact forces. We implement the controller on a full-body, force-controlled humanoid robot. Experimental results demonstrate that the controller can successfully make the robot track captured human motion sequences.

#### I. INTRODUCTION

Motion capture is a good source of data for programming humanoid robots because it contains the natural styles and synergies of human behaviors. However, it is difficult to directly use captured motion data because the kinematics and dynamics of humanoid robots differ significantly from those of humans.

In our previous work [1], we developed a controller that allows a robot to maintain balance while tracking a given reference motion, assuming that the contact state does not change during the motion. We have verified in simulation that the controller can successfully make a humanoid robot model track human motion capture data while maintaining balance.

Compared to existing controllers with similar objectives developed in robotics [2], [3] and graphics [4]–[7], our controller has several advantages:

- It does not explicitly use the lower body for balancing, and therefore is able to utilize reference trajectories for all joints.
- It does not require offline preprocessing such as segmentation or extensive optimization.
- The parameters are not trajectory-specific, meaning that the same controller can potentially be applied to different reference motions.

S.O. Anderson and J.K. Hodgins are with the Robotics Institute, Carnegie Mellon University. soa@ri.cmu.edu, jkh@cs.cmu.edu A supplemental movie is available at

http://www.cs.cmu.edu/~kyamane/humanoids10/movie.mp4.



Fig. 1. A humanoid robot controlled using human motion data.

This paper reports the results of hardware implementation of the controller presented in [1]. We implement the controller on a full-body, force-controlled humanoid robot to verify the performance, as shown in Fig. 1. We describe our solutions to the problems in hardware implementation such as root position/orientation estimation, model uncertainties, and difference between expected and actual contact forces. Experimental results demonstrate that the controller can track human motion capture data while maintaining balance.

After introducing related work in the next section, we briefly summarize the controller originally presented in [1] in Section III. Section IV then describes the issues that arise in hardware implementation of the controller. We present the experimental setup and results in Section V, and conclude the paper with discussions on our contribution and future work in Section VI.

## II. RELATED WORK

In contrast to humanoid control literature that focuses on balance recovery after external forces are applied [8], [9], our goal is to maintain balance while the robot is performing unstructured and potentially noisy movements. Using human motion capture data for controlling humanoid models have been actively investigated in the robotics and graphics fields.

In robotics, Nakaoka et al. [2] developed a method to convert captured dancing motions to physically feasible humanoid robot motions by dividing a motion into a sequence of motion primitives and adjusting their parameters considering the physical constraints of the robot. Their method requires an offline process to identify the required primitives and then define a set of parameters for each primitive. Ott et al. [3] developed a system that allows realtime imitation of captured human motion by biped humanoid robots. However,

K. Yamane is with Disney Research, Pittsburgh, 4615 Forbes Ave. Suite 420, Pittsburgh, PA 15213. kyamane@disneyresearch.com



Fig. 2. Overview of the controller.

imitation is only possible for the upper body because the legs are used exclusively for balancing.

In graphics, a number of algorithms have been proposed for synthesizing motions of virtual characters based on human motion capture data and physical simulation. Sok et al. [4] proposed a method for modifying human motions so that a proportional-derivative (PD) controller can track the entire motion without falling down. Da Silva et al. [5] developed a method for designing a controller that realizes interactive locomotion of virtual characters. They also proposed to combine a model predictive controller with simple PD servos to obtain the optimal joint trajectories while responding to disturbances [6]. Muico et al. [7] proposed a controller that accounts for differences in the desired and actual contact states. Their system relies on existing algorithms to map a motion capture sequence into a physically feasible motion for the character. Although these approaches can track known motion capture sequences, they require either an extensive optimization process or a dedicated controller for each captured motion sequence.

#### **III. BALANCE AND TRACKING CONTROLLERS**

In this section, we briefly summarize the balance and tracking controllers. These algorithms were presented in detail in [1]. Figure 2 shows the block diagram of the complete controller.

## A. Balance Controller

The balance controller takes a reference center of mass (COM) position as input and calculates the desired center of pressure (COP) position using a state-feedback controller designed for a simplified robot model by, for example, linear quadratic regulator or pole assignment.

In our implementation, we use the 3-dimensional 2-joint inverted pendulum with a mobile base shown in Fig. 3 as the simplified robot model. In this model, the entire robot body is represented by a point mass located at the COM of the robot, and the position of the mobile base  $(x \ y)^T$  represents the COP. The relative position of the COM with respect to the COP is represented by the two joint angles  $\theta_1$  and  $\theta_2$ . The state vector x of the model consists of eight variables:  $x, y, \theta_1, \theta_2$  and their velocities.

The balance controller essentially handles the joint movements as a disturbance and outputs a COP position that can maintain balance under disturbances, although it cannot guarantee global stability due to the difference between the



Fig. 3. Inverted pendulum model for the balance controller.

simplified and whole-body robot models, as well as the limited contact area of the feet.

## B. Tracking Controller

The tracking controller first calculates the desired joint and trunk accelerations,  $\hat{\vec{q}}$ , using the same feedback and feedforward scheme as in the resolved acceleration control [10]:

$$\ddot{q} = \ddot{q}_{ref} + k_d (\dot{q}_{ref} - \dot{q}) + k_p (q_{ref} - q) \tag{1}$$

where q is the current joint position,  $q_{ref}$  is the reference joint position, and  $k_p$  and  $k_d$  are constant position and velocity gains that may be different for each joint. In general, motion capture data should be filtered to obtain accurate velocity and acceleration information due to noise. The desired acceleration is also calculated for the six degrees of freedom (DOF) of the trunk using the same control scheme. We denote the vector comprising the desired accelerations of all DOF by  $\hat{q}$ .

The desired accelerations are further modified to  $\hat{\mathbf{q}}' = \hat{\mathbf{q}} + \Delta \mathbf{\ddot{q}}$  so that the accelerations of the contact links become zero. The modification  $\Delta \mathbf{\ddot{q}}$  is obtained by solving the following equation:

$$\boldsymbol{J}_{c}(\ddot{\boldsymbol{q}} + \Delta \ddot{\boldsymbol{q}}) + \boldsymbol{J}_{c} \dot{\boldsymbol{q}} = \boldsymbol{0}$$
<sup>(2)</sup>

where  $J_c$  is the Jacobian matrix of the positions and orientations of the contact links with respect to the generalized coordinates. In our implementation, we use the pseudoinverse of  $J_c$  to minimize the difference from the original  $\ddot{q}$ .

The tracking controller then solves an optimization problem to obtain the joint torques that minimizes a cost function comprising the COP and joint acceleration errors.

The unknowns of the optimization are the joint torques  $\tau_J$  and contact forces  $f_c$  including the friction, subject to the whole-body equation of motion of the robot

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{S}\boldsymbol{\tau}_{J} + \boldsymbol{J}_{c}^{T}(\boldsymbol{q})\boldsymbol{f}_{c} \qquad (3)$$

where M is the inertia matrix, c denotes the gravity, Coriolis and centrifugal forces, and S is the matrix that maps the joint torques to the generalized forces.

The cost function to be minimized is

$$Z = Z_b + Z_q \tag{4}$$

where the two terms of the right-hand side represent the errors in COP  $(Z_b)$  and the desired accelerations  $(Z_q)$ .

The term  $Z_b$  addresses the error between the desired and actual COP positions by evaluating the moment around the desired COP, which is zero if the COP exactly matches the desired. Using the desired COP position  $\boldsymbol{r}_b = (r_{bx} r_{by} 0)^T$  given by the balance controller,  $Z_b$  is computed by

$$Z_b = \frac{1}{2} \boldsymbol{f}_c^T \boldsymbol{P}^T \boldsymbol{W}_b \boldsymbol{P} \boldsymbol{f}_c$$
(5)

where P is the matrix that maps  $f_c$  to the moment around the desired COP, and  $W_b$  is a user-defined positive-definite weight matrix. We compute P by first obtaining a matrix T that converts the individual contact forces to total contact force and moment around the world origin by

$$\boldsymbol{T} = \begin{pmatrix} \boldsymbol{T}_1 & \boldsymbol{T}_2 & \dots & \boldsymbol{T}_{N_C} \end{pmatrix}$$
(6)

and

$$\boldsymbol{T}_{i} = \begin{pmatrix} \mathbf{1}_{3\times3} & \mathbf{0}_{3\times3} \\ [\boldsymbol{r}_{i}\times] & \mathbf{1}_{3\times3} \end{pmatrix}$$
(7)

where  $r_i$  is the position of the *i*-th contact link and  $[a \times]$  is the cross product matrix of a 3-dimensional vector a. The total force and moment are then converted to the moment around COP by multiplying the following matrix:

$$\boldsymbol{C} = \left(\begin{array}{cccc} 0 & 0 & r_{by} & 1 & 0 & 0\\ 0 & 0 & -r_{bx} & 0 & 1 & 0 \end{array}\right)$$
(8)

which leads to P = CT.

The term  $Z_q$  denotes the error from the desired joint accelerations, i.e.,

$$Z_q = \frac{1}{2} (\hat{\ddot{\boldsymbol{q}}}' - \ddot{\boldsymbol{q}})^T \boldsymbol{W}_q (\hat{\ddot{\boldsymbol{q}}}' - \ddot{\boldsymbol{q}})$$
(9)

where  $W_q$  is a user-defined, positive-definite weight matrix.

Using Eq.(3), the cost function is converted to a quadratic form with an analytical solution. Our previous paper [1] also discusses how to choose the weights for the cost function and deal with joint angle and torque limits.

## **IV. HARDWARE IMPLEMENTATION**

In order to implement the controller on hardware systems, we have to address several issues that do not arise in simulation. More specifically, we describe practical solutions to three issues that have great impact on the controller performance. First, we have to estimate the position and orientation of the root from available sensor data because there is no fixed reference frame on the robot. The second issue is the significant uncertainties in the model parameters and external disturbances. Although the controller is designed to tolerate a limited amount of uncertainty, the actual error may be larger than the stability margin. Lastly, the contact force distribution obtained from the optimization may not match the actual one. The discrepancy in the expected and actual contact forces can lead to significant tracking error that cannot be compensated for by the feedback controller, because the joint torques depend on the contact forces.

#### A. Root Position/Orientation Estimation

In our implementation, we estimate the root position and orientation from internal joint angle sensor data. We compute the root position and orientation that minimize the total squared distance between reference and actual contact points using an algorithm equivalent to Arun et al. [11].

Let us denote the position of the *i*-th contact point computed from the reference motion by  $\hat{p}_i$ , which is obtained by performing the standard forward kinematics computation. Because we know the joint angles, we can compute the contact point positions relative to the root link coordinate,  $p_i^r$ . Their positions in the global frame,  $p_i$ , are represented by the root position r and orientation R as  $p_i = Rp_i^r + r$ . In our implementation, we use four corners of each foot as the contact points.

We first obtain the center of the contact points,  $\bar{p}^r = \sum p_i^r/n_c$  and  $\bar{\hat{p}} = \sum \hat{p}_i/n_c$ , and compute  $p_i'^r = p_i^r - \bar{p}^r$ and  $\hat{p}_i' = \hat{p}_i - \bar{\hat{p}}$  for all contact points. We then perform principal component analysis (PCA) on  $p_i'^r$  and  $\hat{p}_i'$  and obtain the three principal axes,  $s_k^r$  and  $\hat{s}_k$  (k = 1, 2, 3) respectively. We determine r and R so that the center of the contact points match, i.e.,

$$\bar{\hat{\boldsymbol{p}}} = \boldsymbol{R}\bar{\boldsymbol{p}}^r + \boldsymbol{r} \tag{10}$$

and the principal axes match

$$\hat{\boldsymbol{s}}_k = \boldsymbol{R} \boldsymbol{s}_k^r \ (k = 1, 2, 3).$$
 (11)

We can obtain r and R by

$$\boldsymbol{R} = \begin{pmatrix} \hat{\boldsymbol{s}}_1 & \hat{\boldsymbol{s}}_2 & \hat{\boldsymbol{s}}_3 \end{pmatrix} \begin{pmatrix} \boldsymbol{s}_1^r & \boldsymbol{s}_2^r & \boldsymbol{s}_3^r \end{pmatrix}^T \quad (12)$$

$$\boldsymbol{r} = \bar{\boldsymbol{p}} - \boldsymbol{R} \bar{\boldsymbol{p}}^r. \tag{13}$$

## B. Model Uncertainties

There are many possible sources of modeling errors, including uncertain link inertial parameters and unknown external forces from the communication wires and hydraulic hoses. While link inertial parameters are mostly constant and can be identified by an identification process (e.g., [12]), external forces are unknown and cannot be determined in advance.

In this paper, we first approximate the effects of model uncertainties by scaling the inertia matrix by a scaling factor chosen based on the total mass of the robot. Although this method is not as precise as formal parameter identification algorithms, we found that it works reasonably well in our experiments.

We also correct the COM position by adding a constant COM offset to the COM position obtained from the robot model. The offset is obtained by assuming that the robot is initially stationary and therefore the COM is above the initial COP, and that the model's COM height is correct.

We estimate the effect of unknown external forces using sensory information at the initial state. We measure the initial joint angles  $q_0$ , joint torques  $\tau_J^0$  and contact forces  $f_c^0$ . Assuming that the robot is initially stationary, i.e.,  $\ddot{q} = 0$  and  $\dot{q} = 0$ , we can estimate the contribution of external forces by

$$\boldsymbol{h} = \boldsymbol{S}\boldsymbol{\tau}_{J}^{0} + \boldsymbol{J}_{c}^{T}(\boldsymbol{q}_{0})\boldsymbol{f}_{c}^{0} - \boldsymbol{g}(\boldsymbol{q}_{0}). \tag{14}$$

At every control cycle, we add h to the left-hand side of the equation of motion (3).

# C. Contact Force Distribution

When both feet are in contact with the ground, the contact forces at the feet become indeterminate, i.e., there are an infinite number of contact forces that result in the same motion. The optimization algorithm described in the previous section chooses one distribution based on a particular choice of cost function, but it may not match the actual distribution. Because the joint torques required to realize a desired motion depend on the contact forces, a discrepancy between the expected and actual contact forces can result in large joint tracking errors. In simulation, this problem does not arise because we apply the optimized torques immediately after starting the simulation. In hardware experiments, on the other hand, we typically start from a default controller to achieve a balanced posture and then gradually switch to the target controller. Even if we try to formulate the dynamics in a form that is independent of the contact forces [13], we have to make an assumption on the contact force distribution, which may or may not be correct on real hardware.

We ensure that our controller's initial contact force distribution is the same as the actual distribution by 1) adding a term to the cost function that penalizes the moment around the initial COP at each foot, and 2) feeding back the contact force differences to bias the optimization results towards the actual contact force distribution.

The new term of the cost function,  $Z_p$ , represents the difference between the initial and optimized COP of individual feet and computed by

$$Z_p = \frac{1}{2} \boldsymbol{f}_c^T \boldsymbol{P}_f^T \boldsymbol{W}_p \boldsymbol{P}_f \boldsymbol{f}_c$$
(15)

where  $W_p$  is a user-defined, positive-definite matrix,  $P_f$  is a block-diagonal matrix whose diagonal blocks are  $P_{fi} = C_{fi}T_i$   $(i = 1, 2, ..., N_C)$  and

$$\boldsymbol{C}_{fi} = \left(\begin{array}{cccc} 0 & 0 & r_{iy} & 1 & 0 & 0\\ 0 & 0 & -r_{ix} & 0 & 1 & 0 \end{array}\right)$$
(16)

where  $(r_{ix} r_{iy})$  is the position of the initial COP of the *i*-th contact link.

With the new term, we can ensure that the cost function takes the minimum value 0 at the initial state when the whole-body COP and individual contact link COPs match the actual initial COPs and  $\ddot{q} = \ddot{q}$ . Therefore, the optimized and actual force distributions also match at the initial state.

The second method is to feed back the difference between the optimized and actual contact forces. Because force measurement usually suffers from noise, we use the integrated force difference, i.e.,

$$\Delta \boldsymbol{f} = \boldsymbol{K}_{I} \int_{0}^{t} (\boldsymbol{f}_{c}^{act}(s) - \boldsymbol{f}_{c}^{opt}(s)) ds$$
(17)

where  $K_I$  is the gain matrix, and  $f_c^{act}(t)$  and  $f_c^{opt}(t)$  are the actual and optimized contact forces respectively. In our experiment, we only use the elements of  $\Delta f$  corresponding to the vertical force and moments around the horizontal axes.

We then add  $J_c^T \Delta f$  to the left-hand side of the equation of motion (3). If the measured vertical contact force is smaller than the optimized force, for example, the leg would tend to bend. The additional term increases the joint torques so that they can realize the desired motion under the larger external force.

#### D. Summary

We summarize the final controller after the modifications we described above. The cost function corresponding to Eq.(4) is now

$$Z = Z_b + Z_q + Z_p \tag{18}$$

where  $Z_p$  has been introduced in Eq.(15). The equation of motion (3) is also modified to

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{c} + \boldsymbol{g} + \boldsymbol{h} + \boldsymbol{J}_{c}^{T}\Delta\boldsymbol{f} = \boldsymbol{S}\boldsymbol{\tau}_{J} + \boldsymbol{J}_{c}^{T}\boldsymbol{f}_{c} \qquad (19)$$

where h and  $\Delta f$  are given in Eqs.(14) and (17) respectively.

# V. EXPERIMENTS

We implement the controller on a full-body humanoid robot developed by Sarcos and owned by Carnegie Mellon University. The robot has 34 joints (7 in each leg and arm, 3 in the torso, and 3 in the neck) driven by hydraulic actuators. Each actuator is controlled by an internal force feedback loop running at 5 KHz for force control.

Each actuator is equipped with a load cell to measure the actual force, which is converted to the joint torque by multiplying the moment arm. Each joint has a potentiometer to measure the joint angle. A six-axis force sensor is embedded in each ankle joint to measure the ground contact force.

## A. Controller Implementation

The kinematics and inertial parameters of the robot are obtained from the CAD model, and are expected to have large errors especially in the inertial parameters. In fact, the robot weighs around 90 kg while the total mass obtained from the CAD model is only 60 kg (hoses and oil are missing from the CAD model). We therefore scale the diagonal elements of the mass matrix by 1.3 assuming uniform distribution of the missing mass.

We employ an inverse kinematics algorithm [14] to obtain the joint angle trajectories from motion capture data considering the joint motion ranges. We provide reference trajectories to 26 joints out of the 34 joints, excluding the wrist, neck roll and neck pitch joints. We add constant offsets to the reference joint angles so that the initial reference pose matches the actual initial pose.

Our implementation of the controller runs at 500 Hz. Because the most computationally expensive part is the calculation of the inertia matrix M, we update only five rows of the inertia matrix at each control cycle to reduce the computation time.



Fig. 4. COP and COM positions in the x (front-aft) direction during the balance control experiment. The drifts in the COM position after 5 seconds are caused by the pushes. Red solid: optimized COP; green dotted: actual COP; blue dashed: COM.

The specific parameters used in the experiments are as follows:

- We use pole assignment to design the balance controller, with closed-loop poles at (-70, -69.5, -69.3, -69.8, -5, -4.8, -4.7, -4.9).
- The tracking controller parameters in Eq.(1) are  $k_p = 3$  (1/s<sup>2</sup>) and  $k_d = 0.5$  (1/s) for the root position feedback and  $k_p = 7$  (1/s<sup>2</sup>) and  $k_d = 0.2$  (1/s) for the other DOF.
- The weight matrices for the cost function are diagonal and the diagonal elements are 1, 1 and 0.001 for W<sub>b</sub>, W<sub>q</sub> and W<sub>p</sub> respectively.
- The gain matrix  $K_I$  of the contact force feedback is diagonal and the diagonal elements are 10.

Compared to the simulation study [1], the proportional gains are similar but we have to use significantly smaller derivative gains, mostly because of the noise in both motion capture data and velocity measurements, as well as the relatively slow control cycle for a force controller.

In our experiments, the robot is initially controlled by a local PD controller at each joint to establish a balanced posture. Our controller then takes over by smoothly blending the joint torque commands from the balance and tracking controllers. The transition completes in two seconds.

## B. Balance Control

We first demonstrate the performance of the balance controller by pushing the robot from several different directions and forces while using a fixed pose as the reference. As shown in Fig. 4, the balance controller effectively commands the COP to maintain balance and the resulting joint torques can realize the commanded COP. Figure 5 depicts the vertical contact force at the left foot. After the controller switching time (2 seconds), the optimized and actual contact forces match well throughout the trial.

# C. Tracking Human Motion Capture Data

We use a motion capture sequence from a publicly available motion capture database [15] performing a nursery theme "I'm a little teapot." Snapshots from a trial are



Fig. 5. Optimized (red solid) and actual (green dotted) vertical contact forces at the left foot during the balance control experiment.



Fig. 7. COP and COM positions in the x (front-aft) direction during the balance control experiment. Red solid: optimized COP; green dotted: actual COP; blue dashed: COM.

shown in Fig. 6 along with the sensor data and controller output, reference motion, and human poses during the motion capture session<sup>1</sup>. The relatively large difference in the left arm pose, which is particularly obvious when the human subject's hand is on the hip, is due to the limited motion range of the shoulder rotation joints. The feet in the sensor data visualization are not in flat contact with the ground. This is caused by the drift of the potentiometers at the ankle joints.

Figures 7 and 8 show that balance controller works similarly to the case without reference motion. Figures 9 and 10 show the representative reference and actual joint angles of the left hip flexion-extension and right shoulder flexion-extension joints. In general, while the joints in the upper body show fairly good tracking performance, the leg joint trajectories have large oscillations due to disturbances from the upper body and low joint velocity feedback gains. However, overall motion is similar to the reference.

<sup>&</sup>lt;sup>1</sup>A corresponding movie is available at

http://www.cs.cmu.edu/~kyamane/humanoids10/movie.mp4.



Fig. 6. Snapshots from the hardware experiment (top row), sensor data and controller output (second row), reference motion (third row) and human motion capture (bottom row), taken every 3 seconds. The lines in the second row are: thick red—actual total contact force; thin red—optimized total contact force; thick white—actual foot contact force; thin white—optimized foot contact force.

## VI. CONCLUSION

In this paper, we reported the results of hardware implementation of our humanoid controller for balancing while tracking motion capture data without contact state changes. We addressed three issues that arise in hardware experiments: root position/orientation estimation from internal joint angle sensors, model uncertainties, and difference between expected and actual contact forces. Experimental results demonstrate that our implementation successfully controls a force-controlled humanoid robot so that it tracks human motion capture data while maintaining balance.

Several issues remain as future work. The current controller cannot maintain balance if the original human motion is highly physically unrealistic for the robot. In such cases, we would have to employ an offline optimization process to adjust the motion to the robot's dynamics [4]. Another limitation is that the reference motion cannot involve contact state changes. We intend to remove this limitation by implementing our recent extension of the controller to motions with stepping [16].

#### REFERENCES

- K. Yamane and J. Hodgins, "Simultaneous tracking and balancing of humanoid robots for imitating human motion capture data," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robot Systems*, 2009, pp. 2510–2517.
- [2] S. Nakaoka, A. Nakazawa, F. Kanehiro, K. Kaneko, M. Morisawa, H. Hirukawa, and K. Ikeuchi, "Learning from observation paradigm: Leg task models for enabling a biped humanoid robot to imitate human dances," *International Journal of Robotics Research*, vol. 26, no. 8, pp. 829–844, 2010.
- [3] C. Ott, D. Lee, and Y. Nakamura, "Motion capture based human motion recognition and imitation by direct marker control," in *Proceedings of IEEE-RAS International Conference on Humanoid Robots*, 2008, pp. 399–405.
- [4] K. Sok, M. Kim, and J. Lee, "Simulating biped behaviors from human motion data," ACM Transactions on Graphics, vol. 26, no. 3, p. 107, 2007.
- [5] M. Da Silva, Y. Abe, and J. Popović, "Interactive simulation of stylized human locomotion," ACM Transactions on Graphics, vol. 27, no. 3, p. 82, 2008.



Fig. 8. Optimized (red solid) and actual (green dotted) vertical contact forces at the left foot during a balance and tracking experiment.



Fig. 9. Reference (red solid) and actual (green dotted) joint angles of the left hip flexion-extension joint.

- [6] —, "Simulation of human motion data using short-horizon modelpredictive control," in *Eurographics*, 2008.
- [7] U. Muico, Y. Lee, J. Popović, and Z. Popović, "Contact-aware nonlinear control of dynamic characters," ACM Transactions on Graphics, vol. 28, no. 3, p. 81, 2009.
- [8] S. Kudoh, T. Komura, and K. Ikeuchi, "The dynamic postural adjustment with the quadratic programming method," in *Proceedings of IEEE/RSJ International Conference on intelligent Robots and Systems*, 2002, pp. 2563–2568.
- B. Stephens, "Integral control of humanoid balance," in *Proceedings of IEEE/RSJ International Conference on intelligent Robots and Systems*, 2007, pp. 4020–4027.
- [10] J. Luh, M. Walker, and R. Paul, "Resolved Acceleration Control of Mechanical Manipulators," *IEEE Transactions on Automatic Control*, vol. 25, no. 3, pp. 468–474, 1980.
- [11] K. Arun, T. Huang, and S. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, pp. 698–700, 1987.
- [12] K. Ayusawa, G. Venture, and Y. Nakamura, "Identification of humanoid robots dynamics using minimal set of sensors," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 2854–2859.
- [13] M. Mistry, J. Buchli, and S. Schaal, "Inverse dynamics control of floating base systems using orthogonal decomposition," in *Proceedings* of *IEEE International Conference on Robotics and Automation*, 2010, pp. 3406–3412.
- [14] K. Yamane and Y. Nakamura, "Natural Motion Animation through Constraining and Deconstraining at Will," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 3, pp. 352–360, July-September 2003.
- [15] "Carnegie Mellon University Graphics Lab Motion Capture Database," http://mocap.cs.cmu.edu/.



Fig. 10. Reference (red solid) and actual (green dotted) joint angles of the right shoulder flexion-extension joint.

[16] K. Yamane and J. Hodgins, "Control-aware mapping of human motion data with stepping for humanoid robots," in *IEEE/RSJ International Conference on Intelligent Systems and Robots*, 2010 (in press).