Proceedings of 2004 IEEE/RSJ International Conference on
Intelligent Robots and Systems
September 28 - October 2, 2004, Sendai, Japan

A Sliding Controller for Bipedal Balancing Using Integrated Movement of Contact and Non-Contact Limbs

Andreas Hofmann
The Media Laboratory, MIT
Computer Science and Artificial Intelligence Laboratory (CSAIL), MIT
Cambridge, MA, USA
hofma@csail.mit.edu

Marko Popovic
The Media Laboratory, MIT
CSAIL, MIT
marko@media.mit.edu

Steven Massaquoi
CSAIL, MIT
sgm@csail.mit.edu

Hugh Herr
The Media Laboratory, MIT
Harvard Division of Health Sciences & Technology
hherr@media.mit.edu

Abstract—**We present an algorithm that provides enhanced flexibility and robustness in the control of single-leg humanoid standing through the coordination of stance leg ankle torques and stabilizing movements of non-contact limbs. Current control approaches generally assume the presence of explicitly specified joint reference trajectories or desired virtual force calculations that ignore system dynamics. Here we describe a practical controller that 1) simplifies control of abstract variables such as the center of mass location using a two-stage model-based plant linearization; 2) determines motion of non-contact limbs useful for achieving control targets while satisfying dynamic balance constraints; and 3) provides robustness to modeling error using a sliding controller. The controller is tested with a morphologically realistic, 3-dimensional, 18 degree-of-freedom humanoid model serving as the plant. It is demonstrated that the controller can use less detailed control targets, and reject stronger disturbances, than previously implemented controllers that employ desired virtual forces and static body calculations.**

I. Introduction

The control of balance for bipedal humanoid robots has been studied extensively. In recent years, a number of humanoid robots capable of walking have been developed. These include the Honda P3 and Asimo robots [5, 6], the Sony SDR [22], and Tokyo University's H6 [10]. These robots achieve balance control using a motion planner that generates reference joint trajectories, and by using simple PD controllers to track those desired trajectories. This approach results in stable walking as long as disturbances are not too large.

Hi-fidelity dynamic simulations have been used for the study of bipedal balancing and locomotion. Hodgins [7] developed a control algorithm for a simulated human runner that achieved a close match, in terms of motion trajectories, with real human motion capture data. As with the previously discussed robots, this approach relied on closely tracking a set of reference trajectories, and robustness to disturbances was not addressed.

Advances in hybrid position and force control [14] have addressed problems of robustness in the presence of disturbances and unstructured environments. These include impedance control techniques applied to robotic arms [9], and a similar virtual model control algorithm applied to legged robots [15, 16]. These algorithms first compute a desired "virtual" force at some "reaction" point on a mechanism's body, and then the required joint torques are computed to achieve that desired virtual force. The former computation is typically based on a setpoint trajectory and virtual spring and damper elements that combine to implement simple feedback control laws. The latter computation is a Jacobian-based static force computation that has been used extensively in robot manipulators [2, 13]. This type of algorithm has been extended to allow center of mass (COM) to be a reaction point [18].

Although the virtual model control algorithm has been applied successfully in the control of a number of legged robots, its usefulness for fast, dynamic motions is limited. The limitation of this approach is due to the fact that all computations are static in nature; the approach assumes that the entire mechanism's system of articulated links is a rigid body. This assumption is reasonable for slow movements, or when the system is not moving at all. However, it breaks down with rapid movements where dynamic forces become more dominant.

The problem of taking into account dynamics has been addressed using a variety of approaches. A technique called "dynamic filtering" [10, 12] involves adjusting an input set of joint trajectories so that dynamic balance constraints are satisfied. The input trajectories must be specified at a relatively high level of detail, and they have

0-7803-8463-6/04/$20.00 ©2004 IEEE

to be close to a correct solution. Slotine [17] developed a sliding controller with feedback linearization for controlling robotic manipulators. For legged systems, Kondak [11] implemented a balance controller for simple bipedal mechanisms using feedback linearization. However, due to the simplicity of the model, the issue of non-contact limb movement was not addressed. In addition, control was in terms of joint state space rather than more abstract outputs of interest, thus making it difficult to prioritize multiple goals. Finally, model error was not taken into account.

In this investigation, we describe a novel control architecture for legged systems where the acceleration of non-contact limbs is employed as a key stabilization strategy. We test the controller on a morphologically realistic humanoid model for the specific movement task of balancing on one leg. The controller incorporates feedback linearization, and quadratic programming-based optimal control, within a sliding control framework. The feedback linearization component decouples and linearizes the dynamics of the plant in terms of the reaction points to be controlled. The optimal controller observes constraints such as joint ranges, maximum joint torques, and the restriction that the foot rotation index (FRI) [4] be within the support polygon. The sliding control framework ensures robustness, allowing for modeling inaccuracies.

We test the humanoid controller using a variety of disturbed initial states, including both forward and lateral COM displacements, and we compare the performance of the controller to a previously developed control approach that does not take dynamics into account [16]. Finally, we study the resulting model behaviors by analyzing the dynamics of the system.

II. Controller Architecture

Simulation experiments were performed using a morphologically realistic humanoid model described in the next section. Subsequent sections provide details of the humanoid control system.

A. 3D Biped Model

A model that captures the essential morphological features of the human lower body relevant for standing, balancing, and walking was developed [8]. The model is three-dimensional with 12 internal (controlled) and 6 external (un-controlled) degrees of freedom. The 12 internal degrees of freedom correspond to joints that can exert torques. The 6 external degrees of freedom correspond to the position and orientation of the trunk of the body. Each leg is modeled with a ball-and socket hip joint, a pin knee joint, and a saddle-type ankle joint. Here the saddle joint architecture allows for ankle plantar/dorsiflexion motions and ankle inversion/eversion. The upper body (head, arms and torso), upper leg and lower leg are modeled with cylindrical shapes, and the feet are modeled with rectangular blocks. The dimensions of each model segment were obtained by considering morphological data that describe average human proportions [19, 21, along with motion capture data [1, 20] used to derive segment lengths, and finally direct measurements on the test subject.

B. Plant Linearization

The feedback linearization architecture of the controller is shown in Fig. 1. The purpose of the feedback linearization is to make the plant appear linear to the controller, $\mathbf{f}_{cont}$.

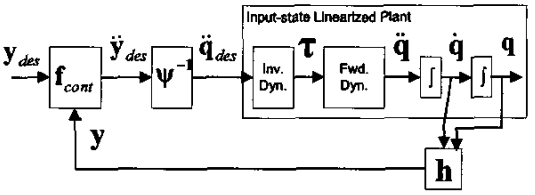


Fig. 1 – Feedback linearization controller architecture

The linearization is accomplished in two stages. First, the forward dynamics of the plant are linearized using an inverse dynamics model, resulting in an input-state linearized plant [17]. Thus, the plant is linear for a controller that selects desired joint accelerations. A second, geometric transform is used to convert from desired accelerations of the outputs (the reaction points) to desired joint accelerations. This transformation, indicated by Ψ^{-1} in fig. 1, is a specialized inverse kinematics transform.

Thus, using the above architecture, if we ignore joint and FRI constraints, the controller $\mathbf{f}_{cont}$ sees the rest of the system as being completely linearized and decoupled. Thus, simple control techniques for SISO 2nd-order linear systems (pole placement, for example) can be used within this controller.

We now discuss some details of these linearizations. The input-state linearization of the plant using inverse dynamics is straightforward. The inverse dynamics for the plant are expressed in the following standard form:

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau} \qquad (1)$$

where $\mathbf{q}$ is a vector of joint angles, $\boldsymbol{\tau}$ is a vector of joint torques (the control input to the plant), $\mathbf{H}(\mathbf{q})$ is a matrix of inertial terms, $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$ is a matrix of velocity-related terms, and $\mathbf{g}(\mathbf{q})$ is a vector of gravitational terms. Choosing eq. 1, (with $\ddot{\mathbf{q}}$ replaced by $\ddot{\mathbf{q}}_{des}$) as the control law and substituting into the forward dynamics yields

$$\ddot{\mathbf{q}} = \ddot{\mathbf{q}}_{des} \qquad (2)$$

Thus, the system is exactly linearized, and completely decoupled into a set of SISO systems. This technique is sometimes called "computed torque", "inverse dynamics" or "feedforward" control in the robotics literature [13].

This linearization is relatively straightforward due to the structure of the plant dynamics. However, the problem is

not solved, because the goal is not specifically to control plant state, but rather, to control outputs derived from plant state. These outputs (COM, for example) are nonlinear functions of plant state, so a further transformation (indicated by Ψ^{-1} in fig. 1) is needed.

The plant outputs (desired positions and orientations of reaction points) are given by

$$\mathbf{y} = \mathbf{h}(\mathbf{q}) \tag{3}$$

where $\mathbf{h}$ is a forward kinematic transformation. Taking partial derivatives yields, for each output y_i

$$\dot{y}_i = \frac{\partial h_i}{\partial(\mathbf{q})}\dot{\mathbf{q}} = \sum_{j=1}^{12}\frac{\partial h_i}{\partial q_j}\dot{q}_j \tag{4}$$

where j indicates the joint. Differentiating this again yields

$$\ddot{y}_i = \sum_{j=1}^{12}\left(\frac{\partial h_i}{\partial q_j}\ddot{q}_j + \frac{\partial^2 h_i}{\partial q_j^{\,2}}\dot{q}_j^{\,2} + \sum_{k=j+1}^{12} 2\frac{\partial^2 h_i}{\partial q_j \partial q_k}\dot{q}_j\dot{q}_k\right) \tag{5}$$

It is useful, at this point, to use spatial notation [3] to represent spatial accelerations of reaction points. With this notation, the spatial acceleration of link i of an articulated mechanism is formulated as

$$\hat{\mathbf{a}}_n = \sum_{i=1}^{n}\hat{\mathbf{s}}_i\ddot{q}_i + \sum_{i=2}^{n}\left[\sum_{j=1}^{i-1}\hat{\mathbf{s}}_j\dot{q}_j\right]\hat{\times}\hat{\mathbf{s}}_i\dot{q}_i \tag{6}$$

where $\hat{\mathbf{a}}_i$ is the spatial acceleration vector, and $\hat{\mathbf{s}}_i$ is the Jacobian column for joint i. All of these vectors are in global coordinates. The vector $\hat{\mathbf{s}}_i$ is the local axis vector, $\hat{\mathbf{s}}_i'$, for joint i transformed to global coordinates using the spatial transformation matrix ${}_0\hat{\mathbf{X}}_i$:

$$\hat{\mathbf{s}}_i = {}_0\hat{\mathbf{X}}_i\hat{\mathbf{s}}_i' \tag{7}$$

Note the similarity between eqs. 5 and 6. For any particular state of the mechanism, eq. 6 is a linear equation of the form

$$\hat{\mathbf{a}}_n = \Psi\ddot{\mathbf{q}} + \Psi_{const} \tag{8}$$

where Ψ is the reaction point Jacobian

$$\Psi = \begin{bmatrix}\hat{\mathbf{s}}_1 & \hat{\mathbf{s}}_2 & \dots & \hat{\mathbf{s}}_n\end{bmatrix} \tag{9}$$

and

$$\Psi_{const} = \sum_{i=2}^{n}\left[\sum_{j=1}^{i-1}\hat{\mathbf{s}}_j\dot{q}_j\right]\hat{\times}\hat{\mathbf{s}}_i\dot{q}_i \tag{10}$$

Thus, eq. 8 provides the linear relation between joint and reaction point accelerations required for the controller architecture shown in Fig. 1.

There is one additional complexity. The angular acceleration given as part of the spatial acceleration vector in eq. 8 is an angular acceleration vector. This is suitable for situations where the desired angular velocity of the reaction point is specified using such a vector. Normally, however, this is not the case; desired angular acceleration is specified in terms of second derivatives of roll, pitch, and yaw angles (a.k.a. Euler angles).

To convert to this form, consider first the angular velocity vector $\boldsymbol{\omega}$. This is related to first derivatives of roll, pitch, and yaw by

$$\boldsymbol{\omega} = \begin{bmatrix} c_\alpha & -s_\alpha c_\gamma & 0 \\ s_\alpha & c_\alpha c_\gamma & 0 \\ 0 & s_\alpha & 1 \end{bmatrix}\begin{bmatrix}\dot{\gamma}\\ \dot{\beta}\\ \dot{\alpha}\end{bmatrix} \tag{11}$$

where α is a rotation (of the reaction point) about the z (yaw) axis, β is a rotation about the y (pitch) axis, and γ is a rotation about the x (roll) axis. The rotation convention used is rotation of β about the global (fixed) y axis, followed by rotation of γ about the global x axis, followed by rotation of α about the global z axis [4]. Using eq. 11 and taking partial derivatives yields

$$\dot{\boldsymbol{\omega}} = \boldsymbol{\varphi} + \begin{bmatrix} c_\alpha & -s_\alpha c_\gamma & 0 \\ s_\alpha & c_\alpha c_\gamma & 0 \\ 0 & s_\alpha & 1 \end{bmatrix}\begin{bmatrix}\ddot{\gamma}\\ \ddot{\beta}\\ \ddot{\alpha}\end{bmatrix} \tag{12}$$

where

$$\boldsymbol{\varphi} = \begin{bmatrix}\left(-s_\alpha\dot{\gamma} - c_\alpha c_\gamma\dot{\beta}\right)\dot{\alpha} + \left(s_\alpha s_\gamma\dot{\beta}\right)\dot{\gamma} \\ \left(c_\alpha\dot{\gamma} - s_\alpha c_\gamma\dot{\beta}\right)\dot{\alpha} + \left(-c_\alpha s_\gamma\dot{\beta}\right)\dot{\gamma} \\ \left(c_\gamma\dot{\beta}\right)\dot{\gamma}\end{bmatrix} \tag{13}$$

Note that for a particular system state, eq. 12 gives a linear relation between the angular acceleration vector, and the vector of second derivatives of Euler angles.

Eq. 8 can be used to find the spatial acceleration of any reaction point. If we choose the COM of each link in the mechanism as a reaction point, then the acceleration of the system COM, in the x, y, and z directions, is given by

$$\frac{d^2\,COM}{dt^2} = \frac{1}{m_{tot}}\sum_{i=1}^{7} m_i\frac{d^2\,COM}{dt^2} \tag{14}$$

For the experiments described here, the following 12 values were chosen as the outputs to be controlled (the elements of $\mathbf{y}$ in fig. 1): x and y COM position, body z position, body roll, pitch, and yaw angle, swing foot x, y, and z position, and swing foot roll, pitch, and yaw. These outputs were controlled using simple PD control laws in $\mathbf{f}_{cont}$, with feedback gains manually tuned.

C. *Multivariable Optimal Controller*

Using the linearization techniques described in the previous section, the system appears to be completely linearized and decoupled to the controller $\mathbf{f}_{cont}$ in fig. 1, but only if there are no constraints. Unfortunately, there are bounds on plant inputs due to saturation limits, and this complicates the problem. If the controller does not take these bounds into consideration, it could generate values to satisfy $\ddot{\mathbf{y}}_{des}$ that cause the bounds to be violated. For example, if the controller does not take into account the fact that the foot support polygon is of finite size, it might generate ankle torques that are too large, and that cause the foot to roll. The controller may be unable to satisfy the desired input while maintaining constraints. To avoid this type of infeasibility, slack variables are introduced for each element of $\ddot{\mathbf{y}}_{des}$, so that the new controller output is

$$\ddot{\mathbf{y}}_{des} = \ddot{\mathbf{y}}_{cont\_out} + \ddot{\mathbf{y}}_{slack} \tag{15}$$

This provides flexibility in that $\ddot{\mathbf{y}}_{des}$ conforms to the controller's linear PD control law (without regard to the actuation bounds), while $\ddot{\mathbf{y}}_{cont\_out}$, the true output of the controller, does obey actuation bounds. The goal of the overall control system is then to minimize $\ddot{\mathbf{y}}_{slack}$, taking into account the relative importance of each output. Thus, it is important to decide which outputs are the "important" ones, and therefore need to be controlled most closely. The goal for the overall control system is then to make the slack variables be 0 for the important outputs, and relatively small for the others.

The question now is how to formulate an optimization problem that minimizes the slack variables and obeys the actuation constraints. There are three types of constraints: 1) constraints on joint angle positions, 2) constraints on joint torques, and 3) constraints that keep the FRI within the support polygon. The FRI is the point on the foot/ground contact surface where the net ground reaction force would have to act to keep the foot stationary [4]. When in a single-support stance, if the FRI is outside the bounds of the actual support polygon, the support foot will begin to roll. Thus, keeping the FRI within these bounds amounts to limiting ankle torques of the stance leg so that the stance foot does not roll. The FRI is given by

$$FRI_x = \frac{\sum_{i=2}^{7} m_i RPx_i\left(RP\ddot{z}_i + g\right) - \sum_{i=2}^{7} m_i RPz_i RP\ddot{x}_i - \sum_{i=2}^{7} H\dot{y}_i}{\sum_{i=2}^{7} m_i\left(RP\ddot{z}_i + g\right)} \tag{16}$$

$$FRI_y = \frac{\sum_{i=2}^{7} m_i RPy_i\left(RP\ddot{z}_i + g\right) - \sum_{i=2}^{7} m_i RPz_i RP\ddot{y}_i + \sum_{i=2}^{7} H\dot{x}_i}{\sum_{i=2}^{7} m_i\left(RP\ddot{z}_i + g\right)}$$

$$\dot{H}_{Gi} = I_{Gi}\,\dot{\omega}_i$$

where $\dot{\omega}_i$ is the angular acceleration vector of link i, in global coordinates.

Fortunately, all bounds can be expressed using linear inequality constraints. Since the equality constraints used for the linearization described in the previous section are all linear, it is possible to formulate the optimization problem as a quadratic program. The variables (columns) of this formulation are as follows: $\ddot{\mathbf{y}}_{slack}$ (the slack variables), $\ddot{\mathbf{y}}_{cont\_out}$ (the specified acceleration output by the controller to the linearized plant), $\ddot{\mathbf{q}}_{des}$ (the joint accelerations), $\mathbf{RP\ddot{x}}$, $\mathbf{RP\ddot{y}}$, $\mathbf{RP\ddot{z}}$ (the COM reaction point x, y, and z accelerations for each link), $\dot{\boldsymbol{\omega}}_x$ and $\dot{\boldsymbol{\omega}}_y$ (the x and y components of angular acceleration of each link), and $\boldsymbol{\tau}$ (the joint torques). Linear equality constraints of the quadratic programming formulation are as follows. Eq. 15 relates $\ddot{\mathbf{y}}_{slack}$ and $\ddot{\mathbf{y}}_{cont\_out}$ through $\ddot{\mathbf{y}}_{des}$, which is determined, outside the quadratic programming optimization, based on simple PD control laws that take $\mathbf{y}$ and $\mathbf{y}_{des}$ as input. Eqs. 8 and 12 relate $\ddot{\mathbf{q}}_{des}$ to $\ddot{\mathbf{y}}_{cont\_out}$, and to $\mathbf{RP\ddot{x}}$, $\mathbf{RP\ddot{y}}$, $\mathbf{RP\ddot{z}}$, $\dot{\boldsymbol{\omega}}_x$, and $\dot{\boldsymbol{\omega}}_y$. Eq. 1 relates $\ddot{\mathbf{q}}_{des}$ to $\boldsymbol{\tau}$.

Linear in-equality constraints of the formulation represent the bounds on joint angle positions, joint torques, and FRI. Bounds on FRI are represented using eq. 16. Bounds on joint torques are represented simply as bounds on the $\boldsymbol{\tau}$ variables in the quadratic program. Bounds on joint angle position are translated to bounds on joint angular accelerations which are set as bounds on the $\ddot{\mathbf{q}}_{des}$ variables.

A quadratic cost function is used with costs assigned to the slack variables and also the joint torques. Slack costs for "important" outputs are higher than for the other outputs. In these experiments, the important outputs are COMx and COMy. These outputs also have correspondingly higher PD gains.

D. *Sliding Control Framework*

Feedback linearization is a powerful technique, but it can be insufficient for real plants because it assumes a perfect plant model. The sliding control algorithm [17] addresses this problem using a two-part structure. The first part is the nominal part; it assumes the model is perfect, and issues control commands using a feedback linearization based on this model. The second part contains additional corrective control terms that compensate for model inaccuracy.

For this problem, the nominal or feed-forward control input to the plant is $\boldsymbol{\tau}$, the joint torque vector output by the inverse dynamics block in Fig. 1. The corrective control terms are feedback torques, $\boldsymbol{\tau}_{fb}$, which are combined with

the feed-forward torques to get the new, combined plant input torque $\boldsymbol{\tau}_{plant}$.

$$\boldsymbol{\tau}_{plant} = \boldsymbol{\tau} + \boldsymbol{\tau}_{fb} \tag{17}$$

Note that the corrective control terms must be applied directly to the torques, the actual inputs to the plant. Thus, these terms bypass the kinematic and inverse dynamics models , and any associated inaccuracies in these models (see Fig. 1). For this study, the inverse dynamics block in Fig. 1 used a slightly simplified model compared with the one used in the forward dynamics plant simulation, so some model inaccuracies were introduced, just as would be the case with an actual plant. The corrective terms are of the form

$$\boldsymbol{\tau}_{fb} = -\boldsymbol{\lambda}\dot{\tilde{\mathbf{q}}} - \mathbf{k}\,\mathrm{sgn}(\mathbf{s}) \tag{18}$$

where $\tilde{\mathbf{q}}$ is the tracking error, defined as the difference between the actual and nominal joint angles

$$\tilde{\mathbf{q}} = \mathbf{q} - \mathbf{q}_{nom} \tag{19}$$

and $\mathbf{q}_{nom}$ is computed by integrating $\ddot{\mathbf{q}}_{des}$ in Fig. 1. The constants in the diagonal matrix $\boldsymbol{\lambda}$ control convergence while on the sliding surface. The vector $\mathbf{s}$ is the distance from the sliding surface, defined as

$$\mathbf{s} = \dot{\tilde{\mathbf{q}}} + \boldsymbol{\lambda}\tilde{\mathbf{q}} \tag{20}$$

The constants in the diagonal matrix $\mathbf{k}$ are made large enough to account for model uncertainty [17].

E. Static Jacobian Controller for Comparison

In order to compare performance of the controller with that of previous approaches, a second controller, based on static Jacobian linearizations [15, 16] was implemented. This controller had the same outputs as the new one, and setpoints and PD gains for each output were specified in the same way. A wide range of gain combinations was explored by manual tuning to optimize performance. For example, proportional (spring) gain for lateral components of COM ranged from 100 to 1000 N/m. Damping gains ranged from 0.1 to 1 times the proportional gains.

III. Results

A series of tests was performed with initial conditions such that the ground projection of the COM was outside the support polygon, and all velocities were set to zero. For such initial conditions, the COM cannot be stabilized by stance ankle torques alone without the foot rolling and the model going unstable. Simple reference trajectories consisting of single, time invariant setpoints were selected for the controller. These setpoints specified the desired equilibrium positions and velocities of the model's COM and swing leg foot. Because the desired final equilibrium posture was to stand on one leg assuming a static pose, all setpoint velocities were set to zero.

Fig. 2 shows the system's recovery from an initial displacement in the lateral (positive y) direction.. From the model's perspective, the left most edge of the foot support polygon is at 0.05m. As is shown in Fig. 2B, the FRI remains within the foot support polygon, while the laterally displaced COM position begins outside the stance foot, but is brought quickly to zero by the controller. Fig. 2C shows the desired, actual, and slack values for the lateral COM acceleration. Note how the slack goes to zero quickly, due to its high penalty. Fig. 2D shows the roll angle of the body. Because roll angle is less tightly controlled, the angle converges, but more slowly than the lateral COM position.

Fig. 3 shows the system's recovery from a forward initial displacement. The front most edge of the foot support polygon is at 0.22 m. As is shown in Fig. 3B, the FRI remains within the foot support polygon, while the forward COM position begins outside the foot, but is brought quickly to zero by the controller. Fig. 3C shows the desired, actual, and slack values for forward COM acceleration. Note how the slack goes to zero quickly, due to its high penalty. Fig. 3D shows the pitch angle of the body. Pitch converges, but more slowly than forward COM position because it is less tightly controlled.

Fig. 4 shows the system's recovery from a combined forward and lateral displacement, and Fig. 5 compares the performance of the sliding controller with that of a simpler controller that uses a static Jacobian linearization. As is shown in Fig. 5, the static Jacobian controller fails to stabilize the model from the same initial conditions outlined in Fig. 4; the forward COM output becomes unstable, and the model falls down.

IV. Discussion

In this paper, we present a controller that employs acceleration of non-contact limbs and stance leg ankle torques as key stabilization strategies. The controller incorporates feedback linearization, and quadratic programming-based optimal control, within a sliding control framework. The feedback linearization component decouples and linearizes the plant dynamics, the optimal controller ensures that important constraints are observed, and the sliding control framework ensures robustness, allowing for modeling inaccuracies.

The results show that the controller makes appropriate use of non-contact limbs and stance leg ankle torques to stabilize the system. The non-contact limbs are used in two ways: to shift the FRI, and to shift the COM. Consider, for example, the numerical simulation experiment shown in Fig. 2. From the model's perspective, the model stands on its left foot, leaning to the left (positive y direction). If the controller were to take no action, it would tip further to the left and fall down. Due to the action of the controller, the upper body leans further to the left, and the swing leg swings out to the right. Both of these actions correspond, initially, to a negative angular acceleration about the x axis.

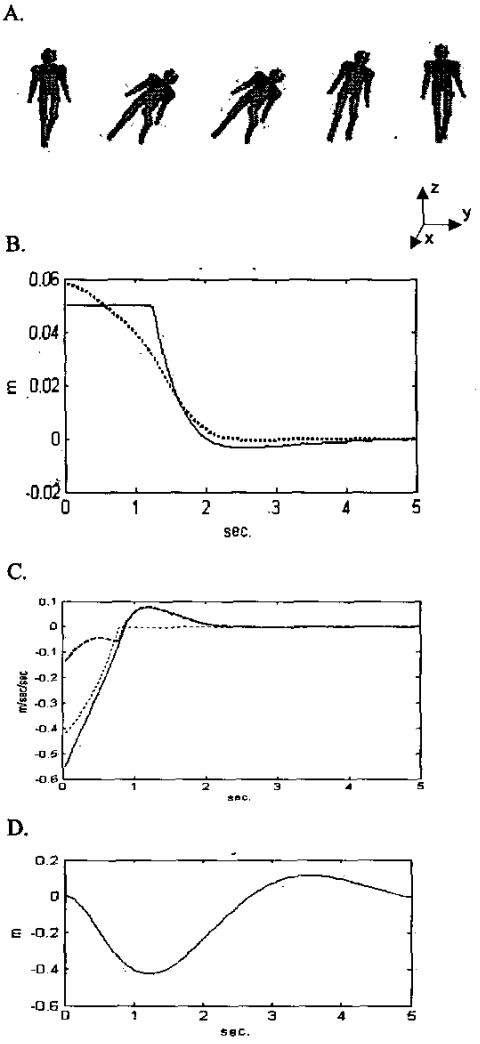


Fig. 2: Lateral disturbance recovery. In 2A, several frames of the model are shown, starting from the maximally displaced COM posture (left most image) to the final static equilibrium posture (right most image). From the perpective of the model, the right leg is the swing leg and the left the stance leg. In 2B, the lateral direction COM (dotted line) and the FRI (solid line) are plotted versus time. In 2C, the desired COM acceleration (solid line), the actual COM acceleration (heavy dashed line), and the slack value (dotted line) are plotted, showing the stabilization of the model's COM. Finally in 2D body roll is plotted, showing the corrective measures taken by the controller.

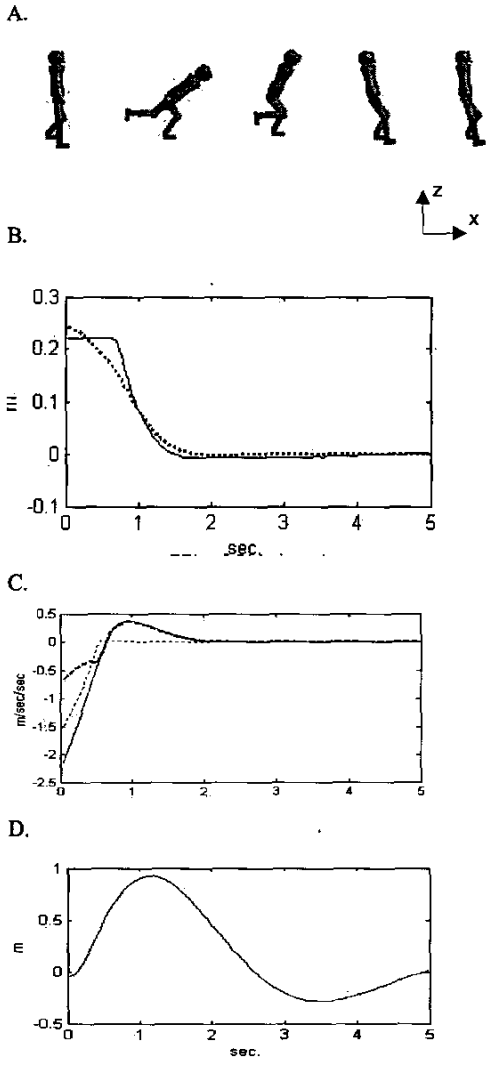
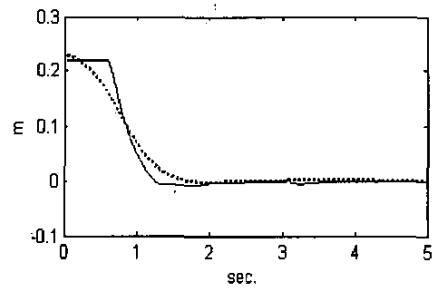


Fig. 3: Forward disturbance recovery. In 3A, several frames of the model are shown, starting from the maximally displaced COM posture (left most image) to the final static equilibrium posture (right most image). From the perspective of the model, the right leg is the swing leg, left is stance leg. In 3B, the forward direction COM (dotted line) and the FRI (solid line) are plotted. In 3C, the desired COM acceleration (solid line), the actual COM acceleration (heavy dashed line), and the slack value (dotted line) are plotted, showing the stabilization of the model's COM. Finally, in 2D, body pitch is plotted.

A.

B.



C.

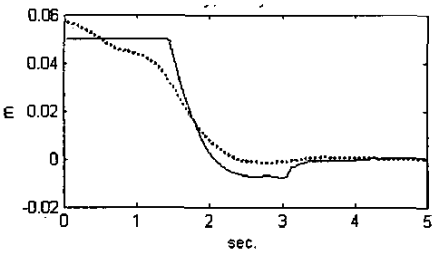


Fig. 4: Forward and lateral disturbance recovery. In 4A, several frames of the model are shown, starting from the maximally displaced COM posture (left most image) to the equilibrium posture. In 4B, the forward direction COM (dotted line) and the FRI (solid line) are plotted. 4C shows lateral COM and FRI.

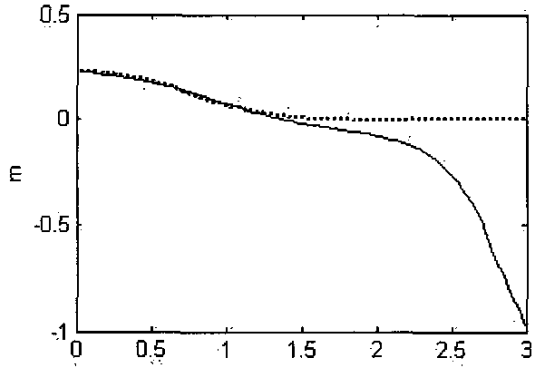


Fig. 5: Forward and lateral disturbance recovery. Trajectory for forward COM using controller based on static Jacobian linearization (solid line), and using controller described here (dotted line). Initial conditions are the same as those in Fig. 4.

From equation (16), it is easy to see that this negative angular acceleration about the x axis allows a linear acceleration of the COM to the right (in the negative y direction) while not requiring the FRI to shift further to the left (positive y direction). This is important since, as shown in Fig. 2, the FRI begins up against the left-most edge of the foot support polygon. As the COM approaches the desired position, the FRI moves away from the edge and towards the center of the foot support polygon. At this point, the swing leg and body are able to return to their nominal neutral positions.

The lateral acceleration of the swing leg to the right (negative y direction) is also beneficial in that it moves part of the model's mass to the right, and so, helps move the COM in the right direction. The net effect of the swing leg and body movements is an overall angular acceleration at the ankle joint that, together with the action of the stance ankle torque, moves the COM back to the center of the foot support polygon.

The extreme case of non-contact limb movement occurs when the support polygon becomes very small, as is the case for a tight-rope walker. A tight-rope walker's support polygon is very narrow, and therefore, little stance ankle torque can be exerted. Lateral forces by the foot against the tight-rope move the COM, but also create torques of the COM about the contact point. This must be countered by spin angular accelerations (angular accelerations about the COM), so that overall angular momentum is conserved. The spin angular accelerations are generated by movement of the non-contact limbs. Thus, a tight-rope walker extends his arms, and moves his arms, body, and non-contact leg to generate appropriate spin angular accelerations.

An important feature of the controller is that the coordinated behavior of the stance leg and non-contact limbs is not controlled explicitly, but rather, emerges indirectly from a high-level specification of desired behavior. This specification is given in terms of setpoints and PD gains for the COM, body orientation, and swing leg control outputs, in terms of constraints such as the one on the FRI (equation 16), and in terms of penalties for slacks and torques in the optimization cost function.

Another important feature of the controller is that, due to its extended range of operation, it can reject significant disturbances more easily than simpler controllers (as shown in Fig. 5). This feature also means that reference trajectories for the new controller need not be as detailed as those for simpler controllers. The reference "trajectories" for the tests of Figs. 2 through 5 were single, time invariant setpoints for COM, body orientation, and swing leg outputs. Simpler controllers require more detailed reference trajectories, with more waypoints as a function of time. For example, the static Jacobian controller that failed in the test of Fig. 5 could be made to work for this case if more detailed reference trajectories for stance leg, swing leg, and body were provided. However, this level of detail puts significant computational burden on the motion planning component of an integrated motion planning and control system. The motion planner has to be executed more frequently, when there are disturbances, and it must produce more detailed reference trajectories.

Detailed evaluation of the sliding control framework's ability to handle model uncertainty was beyond the scope of this investigation. We plan to perform such an evaluation by testing the algorithm on simulations where model errors are introduced, and on actual robots.

Additionally, it would be interesting to investigate the extent to which this framework allows for use of simplified dynamics models, where some of the terms in equation (1) are simplified or omitted.

In the future, we plan to conduct a series of tests using human subjects, where the COM of the subject is initially displaced in a manner similar to that for the above-described tests. We will analyze the stabilizing motions of these subjects and compare them with those of the controller. We expect that this will be useful for fine-tuning the controller's PD gains and slack variable costs. We also plan to combine this controller with a fully integrated motion planning system, and evaluate its performance for more complex maneuvers such as walking and running.

REFERENCES

[1] AMTI OR6-5 Biomechanics Platforms, http://www.amtiweb.com.

[2] Craig, J. J., (1989) "Introduction to Robotics: Mechanics and Control", Reading, Massachusetts, Addison-Wesley, pp. 152 – 180

[3] Featherstone, R., 1987, "Robot Dynamic Algorithms", Boston, Massachusetts, Kluwer Academic Publishers, pp. 155 – 172

[4] Goswami, A., 1999, "Postural stability of biped robots and the foot rotation indicator (FRI) point", International Journal of Robotics Research, July/August

[5] Hirai K., 1997, "Current and Future Perspective of Honda Humanoid Robot" *Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robot and Systems* Grenoble, France:IEEE, New York, NY, USA. pp. 500-508.

[6] Hirai K., Hirose, M., Haikawa Y. and Takenaka T., 1998, "The Development of Honda Humanoid Robot" *IEEE International Conference on Robotics and Automation* Leuven, Belgium:IEEE, New York, NY, USA. pp. 1321-1326.

[7] Hodgins, J. K., 1996, "Three-Dimensional Human Running", Proceedings of the IEEE Conference on Robotics and Automation,

[8] Hofmann, A. G., Popovic, M. B., and Herr, H. (2002) "Humanoid Standing Control: Learning from Human Demonstration", Journal of Automatic Control 12(1), pp. 16 – 22

[9] Hogan, N. (1985) "Impedance Control: An Approach to Manipulation – Part I: Theory", Journal of Dynamic Systems, Measurement, and Control, 107:1-7

[10] Kagami, S., Kanehiro, F., Tamiya, Y., Inaba, M., Inoue, H., 2001, "AutoBalancer: An Online Dynamic Balance Compensation Scheme for Humanoid Robots", in "Robotics: The Algorithmic Perspective", Donald, B. R., Lynch, K. M., and Rus, D., editors, A. K. Peters Ltd. pp. 329 – 340

[11] Kondak, K., Hommel, G., 2003, "Control and Online Computation of Stable Movement for Biped Robots", *Proc. International Conference on Intelligent Robots and Systems* (IROS)

[12] Kuffner, J., Kagami, S, Nishiwaki, K., Inaba, M., Inoue, H., 2002, "Dynamically-stable Motion Planning for Humanoid Robots", Autonomous Robots vol. 12, No. 1, pp. 105 - 118

[13] Paul, R. P., 1981, "Robot Manipulators", Cambridge, Massachusetts: The MIT Press

[14] Pratt, G., Williamson, M., 1995, "Series Elastic Actuators", IEEE International Conference on Intelligent Robots and Systems, pp 1:399-406

[15] Pratt, J., Torres, A., Dilworth, P., Pratt, G., 1996, "Virtual Actuator Control," *Proc. International Conference on Intelligent Robots and Systems* (IROS)

[16] Pratt, J., Dilworth, P., Pratt, G., 1997, " Virtual Model Control of a Bipedal Walking Robot", *Proc. International Conference on Robotics and Automation* (ICRA)

[17] Slotine, J., Li, W. , 1990, "Applied Nonlinear Control", Prentice Hall

[18] Sugihara, T., Nakamura, Y., Inoue, H., "Realtime Humanoid Motion Generation through ZMP Manipulation based on Inverted Pendulum Control", 2002, *Proc. International Conference on Robotics and Automation*

[19] Tilley, A. R., and Dreyfuss, H. (1993) "The measure of man and woman," *Whitney Library of Design*, an imprint of Watson-Guptill Publications, New York

[20] Vicon Motion Systems, http://www.vicon.com.

[21] Winters, D. A. (1990) "*Biomechanics and Motor Control of Human Movement*," John Wiley & Sons, Inc., New York

[22] Yamaguchi, J., Soga, E., Inoue, S., Takanishi, A., 1999, "Development of a Bipedal Humanoid Robot -Control Method of Whole Body Cooperative Dynamic Biped Walking", ICRA '99, IEEE, pp. 368 - 374