Optimal Feedback Control for Character Animation Using an Abstract Model

Yuting Ye C. Karen Liu

Georgia Institute of Technology*

Abstract

Real-time adaptation of a motion capture sequence to virtual environments with physical perturbations requires robust control strategies. This paper describes an optimal feedback controller for motion tracking that allows for on-the-fly re-planning of long-term goals and adjustments in the final completion time. We first solve an offline optimal trajectory problem for an abstract dynamic model that captures the essential relation between contact forces and momenta. A feedback control policy is then derived and used to simulate the abstract model online. Simulation results become dynamic constraints for online reconstruction of full-body motion from a reference. We applied our controller to a wide range of motions including walking, long stepping, and a squat exercise. Results show that our controllers are robust to large perturbations and changes in the environment.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation;

Keywords: Character animation, Physics-based Animation, Motion capture, Optimal Control

1 Introduction

While motion capture data is ubiquitous in the entertainment and video game industry as a source of realistic human motion, the process of acquiring and integrating motion capture data into realistic character motions in a virtual environment is still a time consuming and difficult process. In particular, adapting motion capture data to situations or physical interactions that differ from the conditions at acquisition time remains quite challenging. Often, in order to get usable motion data, the film or video game creator needs to recreate the virtual environment on the capture stage itself, each time she requires a motion in a different environmental setting. This reality clearly falls short of the ideal. Instead, we would like to be able to create a wide variety of realistic interactions from a single motion captured sequence in response to a physical simulation of the virtual world.

Within the computer animation community the primary technique for real-time adaptation of a motion sequence to physical interactions is motion tracking via feedback control algorithms. Strict motion tracking, however, is often an undesirable behavior when the performing character experiences a large perturbation, or is acting in a new environment. People anticipate potential future interactions, and may also re-plan their movements as the environmental situation dictates. In addition, human movement is governed by biomechanical and physical principles that strongly influence the shape and trajectory of the actions taken. In contrast, the goal of current feedback control algorithms for motion tracking is strictly adherence to the reference motion itself. Consequently, they often produce unnatural looking results when they need to recover from strong deviations from the original motion.

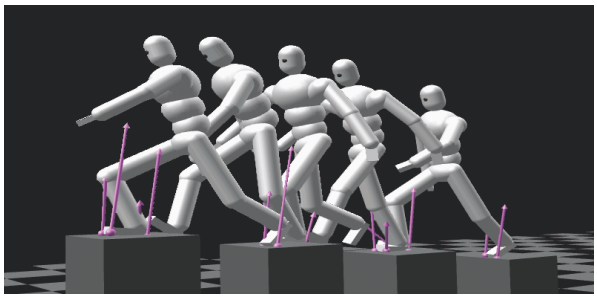


Figure 1: *A new technique to control character motion under physical perturbations and changes in the environment.*

In this paper, we describe a new approach to design feedback controllers robust to perturbations in the virtual world. We seek to design a controller that allows for online re-planning of long-term goals and incorporates an accurate nonlinear dynamic system with high-level balance strategy. Our formulation of motion tracking as an optimal control problem provides two key advantages over previous tracking controllers. First, our controller respects the final goal state and is flexible to adjust the completion time. Its ability to modify the final goal state and completion time produces strategies with anticipatory and replanning behavior. Essentially, the controlled character can "see" the change of the environment ahead of time and adjust the control forces properly in advance (Figure 1). Second, incorporation of a nonlinear dynamic system provides more accurate estimates of the control outcome, and a high-level balance strategy would ensure robust behaviors. As a result, our controller can perform well with large feedback errors. The combination of these improvements enables our control algorithm to generate realistic and robust adaptations from a reference motion to widely varying conditions.

While optimal control theory offers useful tools for solving feedback controllers for a variety of problems including those with final constraints and flexible completion time, our particular problem poses unique challenges. Our feedback controller requires the solution of a two-point boundary optimal trajectory problem, which is known to be very difficult for large nonlinear dynamic systems. The nonlinearity and complexity of human motion makes a full-body formulation impractical. A practical alternative is to formulate a linear quadratic regulator (LQR): a linearized dynamic system with quadratic objective functions. However, this simplification cannot handle higher-order objectives such as angular momentum regularization, which is an important biomechanical principle shown to be essential for balance. Moreover, the linear approximation of dynamics fails rapidly for large state errors. To deal with these issues, we designed an abstract dynamic system that expresses fundamental aspects of human motion, especially the relation between contact forces and angular momentum, and is still manageable by existing trajectory optimization techniques such as differential dynamic programming (DDP).

The abstract dynamic model takes global motion, including center of mass position and linear and angular momentum, as state and

*email:{yuting,karenliu}@cc.gatech.edu

contact forces as control. A control policy of this model addresses one of the most fundamental problem in human motion: the relation between the under-actuated degrees of freedom (DOFs) and the contact forces. With no assumptions of the underlying kinematics structure, our abstract model is generic enough to represent any motions that utilize contacts.

Our algorithm consists of an offline optimization that solves an optimal feedback control policy for the abstract model, and an online optimization that synthesizes full-body motions with consistent dynamics to the abstract model while tracking a reference motion. The offline optimization generates contact forces that reproduce the reference motion with minimum angular momentum and force usage. When applied online, the feedback controller computes contact forces to recover the motion into balanced states under perturbations while meeting the final goal. The feedback controller also anticipates changes in the environment and adjusts the motion timing when necessary. Online simulation of the feedback forces on the abstract model produces momenta that serve as constraints to reconstruct full-body motion from the reference.

Our method greatly enhances the capability of one single motion capture sequence under different dynamically challenging conditions. When we test our algorithm on a normal walk, a long stepping, and a squat exercise, results show that our controller performs robustly for different types of motion and produces natural responses to dynamical and environmental perturbations.

2 Related Work

Physically simulated character animation has been an important research area in advancing the believability of human figures in many online applications. Earlier work by Hodgins et al.[Hodgins et al. 1995; Wooten 1998] demonstrates that complex human movement and maneuvers can be physically simulated in a virtual environment. While the results are compelling, the immense manual efforts and expertise in designing robust controllers prevent the technique from being widely adapted by real-world applications. In the next decade, a number of researchers have proposed different techniques to improve the control strategies [Laszlo et al. 1996; Yin et al. 2007; Shiratori et al. 2009], automate the design process [Faloutsos et al. 2001], generalize to different characters [Hodgins and Pollard 1997], or optimize control parameters [Wang et al. 2009].

Much research effort has focused on simulating humanlike motion by incorporating motion capture data. Zordan and Hodgins showed that tracking upper body motion can be achieved by using simple proportional-derivative (PD) controllers [1999]. Their method has since been enhanced by including a feedforward control term [Yin et al. 2003], applying antagonistic controllers [Neff and Fiume 2002], controlling the timing [Allen et al. 2007], or combining with multiple tasks [Abe and Popović 2006]. Although effective, these methods largely simplified the problem by avoiding the issue of balance. When full-body balance is taken into account, dynamic controllers must overcome difficult problems due to underactuation and limited contact forces in addition to tracking the input motion.

A variety of techniques are proposed to control full-body balance, such as correcting the input motion data for a planar character [Sok et al. 2007], or integrating with a hand-crafted balance controller for $3D$ motions [Zordan and Hodgins 2002]. Other methods incorporate balance control using optimization of contact forces. Short-horizon optimization is used to regulate body center of mass [Abe et al. 2007; Jain et al. 2009] and momenta [Macchietto et al. 2009] at every instant to achieve static balance. Dynamic balance in locomotion can be achieved by planning controls through the whole motion trajectory [da Silva et al. 2008; Muico et al. 2009]. Optimization of contacts and forces can also generate locomotion from scratch [van de Panne 1997; Wampler and Popović 2009] or retarget a motion to a different dynamic model [Pollard and Behmaram-Mosavat 2000]. We also plan the contact forces to achieve a long-term goal. Unlike previous work, which precisely follows the timing and action of the reference motion with approximated dynamics, our method allows for changes in final constraints and completion time with an accurate global dynamic model.

Simplified representations of human body are often useful in the simulation or optimization of character motion [Popović and Witkin 1999]. These abstract models, however, are typically designed for specific types of activities, such as a spring-mass model for running and hopping [Blickhan 1989], or an inverted pendulum for standing [Yamane and Hodgins 2009] or walking [Alexander 1995; Kuo et al. 2005; Stephens and Atkeson 2009]. Inspired by Shapiro and Lee [2009], we propose an abstract dynamic model that captures the essential relation between external forces and the momenta of the center of mass. In addition, our abstract model is an accurate description of the global motion with no assumptions of the underlying kinematics structure. It is simple enough to apply to a long-horizon optimization problem and generic enough to apply in a wide range of motions that involves contacts.

Although physically simulated characters present realistic responses to external perturbations, nearly all interactive applications today still use kinematically controlled characters. For real-time applications like video games, having a precisely predictable and controllable character is paramount. Researchers have suggested hybrid systems that can switch between simulation and motion capture data based on the timing of perturbations [Zordan et al. 2005], or divide kinematic control and dynamic control based on the coordinated actuation in the input motion [Ye and Liu 2008]. Our work also exploits the combined benefits of both approaches. We ensure physical realism of the global motion, and kinematically control the detailed joint configuration.

Many human motor skills require control of whole body linear and angular momentum to achieve task-level goals while maintaining balance. Several researchers in computer graphics have demonstrated that aggregate body momentum can be a compact representation for editing ballistic motion [Liu and Popović 2002; Abe et al. 2004] or locomotion [Komura et al. 2004]. Regulating linear and angular momenta have also been investigated for balance control. A rich body of research in robotics demonstrated the positive effect of minimizing angular momentum on walking and stepping [Popovic et al. 2004; Kajita et al. 2003; Goswami and Kallem 2004; Herr and Popovic 2008]. Macchietto et al. [2009] showed that simultaneously controlling the center of mass and the center of pressure via changes of momenta resulted in much more robust and fluid motion for standing balance against perturbations. Our method confirms the importance of momentum control in human motion both in terms of maintaining balance and producing natural movements. Rather than tracking the momentum trajectory from the input motion, we apply a zero-angular momentum strategy from biomechanics and robotics literature. As a result, our method produces robust control for high dynamic motions without any assumptions about the momentum patterns.

3 Overview

We describe a physics-based algorithm for producing realistic character motion against perturbations. The input to our algorithm is a reference motion sequence and the output is a real-time motion controller that tracks the input motion and also allows for both passive responses to perturbations and active re-planning of goals. Our algorithm controls and simulates a simple abstract model that captures the most essential physical interactions with the environment:

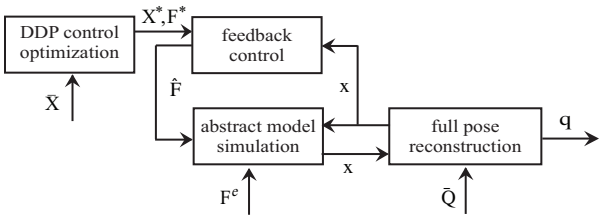


Figure 2: *Overview.*

the relation between external forces and the global motion. The global motion then serves as a constraint to reconstruct the full-body motion.

In the offline process described in section 4.1, we derive an optimal feedback control from the solution of a long-term trajectory planning problem. The control policy aims to track the reference motion while maintaining balance with minimum effort. It can also adjust the completion time and the final goal in the long-term plan when necessary. At each time step of online simulation, we simulate the abstract model under the feedback control force $\mathbf{F}$ and external perturbation $\mathbf{F}^e$. The full-body pose $\mathbf{q}$, including global translation and rotation degrees of freedom as well as joint configurations, is then constructed from the reference motion $\bar{\mathbf{Q}}$ by matching the abstract state $\mathbf{x}$. Figure 2 illustrates the simulation process.

4 Optimal control for the abstract model

In the abstract model, the state variable $\mathbf{x}$ is defined as the global motion of the character and the control variable is defined as the contact forces $\mathbf{F}$. The global translational motion can be described by the position of the center of mass (COM), $\mathbf{C}$, and linear momentum P and angular momentum L. We also want to represent the global orientation, but it cannot be directly computed from COM, so we approximate its effect with the integral of angular momentum Φ: $\Phi(t) = \Phi(t_0) + \int_{t_0}^{t} \mathbf{L}$, where $\Phi(t_0)$ is simply set to zero. The state variable of the abstract model is then defined as $\mathbf{x} = \begin{bmatrix}\mathbf{C} & \mathbf{P} & \Phi & \mathbf{L}\end{bmatrix}^T$. The dynamic equation of the abstract model can be expressed in Equation (1).

$$\dot{\mathbf{x}} = A\mathbf{x} + B(\mathbf{x},t)\mathbf{F} + G, \tag{1}$$

where

$$A_{12\times12} = \begin{bmatrix} \mathbf{0} & \mathbf{I}_{3\times3}/M & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{3\times3} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix},$$

$$B(\mathbf{x},t)_{12\times3p} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{I}_{3\times3} & \mathbf{I}_{3\times3} & \dots & \mathbf{I}_{3\times3} \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ (c_1(t)-\mathbf{C})_\times & (c_2(t)-\mathbf{C})_\times & \dots & (c_p(t)-\mathbf{C})_\times \end{bmatrix},$$

$$G_{12\times1} = \begin{bmatrix}\mathbf{0} & M\mathbf{g} & \mathbf{0} & \mathbf{0}\end{bmatrix}^T,$$

where $\mathbf{g}$ is gravity, M is the total mass of the system, and $_\times$ denotes the skew-symmetric matrix form of a vector. The number of contacts p and their positions $c(t)$ are time-varying parameters determined by the input motion. As a result, the width of $B(\mathbf{x},t)$ depends on the number of contact points p at the time t in the input motion.

Our dynamic system is still nonlinear because of the product term of state and control. Nonetheless, without any further approximation or linearization, this abstract model significantly improves the convergence in control optimization described in Section 4.2. The problem would be otherwise impossible to solve for a full-body dynamic system. In addition, the generic representation of the abstract model enables a wide applicability of our feedback controller.

4.1 Control optimization

Our goal is to derive a controller that is robust for a wide range of states around the reference trajectory and also reflects important properties in natural human motion such as minimum effort and regulating angular momentum to maintain balance. We formulate an optimization problem with the desired objectives, so that we can approximate a feedback control policy for the neighboring states around the optimal solution.

Given a reference trajectory $\bar{\mathbf{X}}$ extracted from $\bar{\mathbf{Q}}$, we want to solve an optimal trajectory that respects its initial and final states. Incorporation of a final state constraint provides several benefits. First, it allows us to break down a long sequence into shorter segments and concatenate them seamlessly. Second, it explicitly enforces the motion to stay in balanced states. Third, it enables replanning of the final goal on the fly. To improve the robustness and naturalness of the control policy, we minimize angular momentum and control forces in addition to tracking the reference motion. From the optimal solution of this optimization problem, we obtain an online feedback controller for the neighboring states around the solution.

Equation (2) summarizes the optimization problem.

$$\begin{aligned} \min_{\mathbf{X},\mathbf{F}} \int_{t_0}^{t_f} (\|\mathbf{F}(t)\|^2_{\mathbf{W}_1} + \|\mathbf{x}(t)\|^2_{\mathbf{W}_2} + \|\mathbf{x}(t) - \bar{\mathbf{x}}(t)\|^2_{\mathbf{W}_3}) dt \\ \text{subject to} \quad \dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B(\mathbf{x}(t),t)\mathbf{F}(t) + G, \\ \mathbf{F}(t) \in K, \\ \Psi(\mathbf{x}(t_f)) = \mathbf{x}(t_f) - \bar{\mathbf{x}}(t_f) = \mathbf{0}, \\ \mathbf{x}(t_0) = \bar{\mathbf{x}}(t_0), \end{aligned} \tag{2}$$

where t_f indicates the final time of the input motion. The contact forces are unilateral and constrained by their friction limits approximated by the friction cone K ([Abe et al. 2007]). $\mathbf{W}_1, \mathbf{W}_2,$ and $\mathbf{W}_3$ are diagonal weighting matrices for force minimization, angular momentum minimization, and tracking, respectively. Determining the weight is trivial for the abstract model. We will report the weights in Section 6.

We apply Differential Dynamic Programming ([Jacobson and Mayne 1970; Gershwin and Jacobson 1968]) to solve this fixed-time continuous optimization. The final constraint is incorporated by augmentation with a Lagrangian multiplier μ in the objective function.

$$V = \mu^T \Psi(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} (\|\mathbf{F}\|^2_{\mathbf{W}_1} + \|\mathbf{x}\|^2_{\mathbf{W}_2} + \|\mathbf{x} - \bar{\mathbf{x}}\|^2_{\mathbf{W}_3}) dt. \tag{3}$$

We initialize the controls using inverse dynamics of the reference trajectory, assuming this initial guess is close to a global solution. We follow the procedure described in Chapter 2.5 in Jacobson and Mayne [1970] to solve this optimization. The midpoint method is used to numerically integrate the solution at each discrete time step.

Once we have the solution $\mathbf{X}^*$, $\mathbf{F}^*$, and μ^* to Equation (2), we can use control force $\mathbf{F}^*$ to simulate $\mathbf{X}^*$ and satisfy $\Psi = \mathbf{0}$. However, in the case of perturbations, such as pushes or changes in the environment, we need to adjust the control forces such that they still optimize V at the perturbed states. We use a first-order approximation to approximate the first derivatives of the perturbed states around $\{\mathbf{X}^*, \mathbf{F}^*, \mu^*\}$. Because first derivatives vanish at the solution, we get a linear feedback control policy. To produce a more robust and flexible controller, we in addition allow the final time to change, then our feedback control becomes a linear combination of small changes in $\mathbf{x}$, μ and t_f:

$$\delta\mathbf{F} = \mathbf{K}^{\mathbf{x}}\delta\mathbf{x} + \mathbf{K}^{\mu}\delta\mu + \mathbf{K}^{t}\delta t_f, \tag{4}$$

where the time varying gains $\mathbf{K}^{\mathbf{x}}, \mathbf{K}^{\mu}$, and $\mathbf{K}^{t}$ can be computed from $\mathbf{X}^*$ and $\mathbf{F}^*$ (details in Appendix A).

4.2 Feedback control policy

If we can evaluate the deviation $\delta\mathbf{x}$, $\delta\mu$, and δt_f, we can use the control policy (Equation 4) to compute the deviation of control force $\delta\mathbf{F}$. The relation of $\delta\mathbf{x}$, $\delta\mu$, and δt_f is expressed in two linear equations derived from the linearization of the first-order optimality condition $V_\mu = \mathbf{0}$ and $V_{t_f} = 0$:

$$\delta V_\mu(\mathbf{X}^*, \mu^*, t_f^*) = V_{\mu\mathbf{x}}(t_c)\delta\mathbf{x} + V_{\mu\mu}(t_c)\delta\mu + V_{\mu t_f}(t_c)\delta t_f = \mathbf{0} \quad (5)$$

$$\delta V_{t_f}(\mathbf{X}^*, \mu^*, t_f^*) = V_{t_f\mathbf{x}}(t_c)\delta\mathbf{x} + V_{t_f\mu}(t_c)\delta\mu + V_{t_f t_f}(t_c)\delta t_f = 0 \quad (6)$$

Change of final time In a fixed-time controller where $\delta t_f = 0$, the reference time index t_c is the same as the elapsed time index t. We can simply compute the deviation in the current state as $\delta\mathbf{x} = \mathbf{x}_t - \mathbf{x}^*_{t_c}$. However, in a free-final-time controller, t_c changes with the final time rather than incrementing along with t. At each time step, we estimate the remaining time $t_f - t_c$ based on $\delta\mathbf{x}$ and compare the new final time with t_f to get δt_f. This dependency between $\delta\mathbf{x}$ and t_c requires us to solve them simultaneously ([Speyer and Bryson 1968]).

We first derive the relation between $\delta\mathbf{x}$ and δt_f from Equation (5) and Equation (6) as:

$$\begin{aligned} \delta t_f &= K^d \delta\mathbf{x}, \\ K^d &= \frac{V_{t_f\mathbf{x}} - V_{t_f\mu}V_{\mu\mu}^{-1}V_{\mu\mathbf{x}}}{V_{t_f\mu}V_{\mu\mu}^{-1}V_{\mu t_f} - V_{t_f t_f}}, \end{aligned} \quad (7)$$

where K^d is evaluated at t_c. We then approximate $\delta\mathbf{x}$ at $\mathbf{x}^*_{t_c}$ as:

$$\delta\mathbf{x} = (\mathbf{x}_t - \mathbf{x}^*_{t_c}) - \dot{\mathbf{x}}^*_{t_c}(t - t_c)h, \quad (8)$$

where h is the time step.

It is easy to see that $\delta t_f = (t - t_c)h$, the change in final time is the same as the change in the current reference index. Arranging terms in Equation (7) and Equation (8), we get Equation (9):

$$(t - t_c)h = \frac{K^d(t_c)}{1 + K^d(t_c)\dot{\mathbf{x}}^*_{t_c}}(\mathbf{x}_t - \mathbf{x}^*_{t_c}). \quad (9)$$

We can precompute K^d for all the time indices in the input motion offline and enumerate the entire sequence online to find a t_c that best satisfies Equation (9). Given t_c, We can compute δt_f and $\delta\mathbf{x}$, and compute $\delta\mu$ from either Equation (5) or Equation (6).

Change of final constraint In addition, we derive the relation $V_\mu = \Psi$ from Equation (3). If we take derivative on both sides: $\delta V_\mu = \delta\Psi$, we can change the final constraint value by substituting the desired change $\delta\Psi$ in Equation (5). In our case, because Ψ has no explicit dependence on time, $\delta\Psi$ is simply $\Delta\Psi$. Equation (7) and Equation (9) then become the following:

$$\delta t_f = K^d\delta\mathbf{x} + K^c\Delta\Psi, \quad (10)$$

$$(t - t_c)h = \frac{K^d(t_c)}{1 + K^d(t_c)\dot{\mathbf{x}}^*_{t_c}}(\mathbf{x}_t - \mathbf{x}^*_{t_c}) + \frac{K^c(t_c)}{1 + K^d(t_c)\dot{\mathbf{x}}^*_{t_c}}\Delta\Psi, \quad (11)$$

where

$$K^c = \frac{V_{t_f\mu}V_{\mu\mu}^{-1}}{V_{t_f\mu}V_{\mu\mu}^{-1}V_{\mu t_f} - V_{t_f t_f}}.$$

We again precompute K^c offline and specify $\Delta\Psi$ on the fly. A nonzero $\Delta\Psi$ changes the value of t_c, thus affects both the state and the final time. For example, when we change the desired final position of COM, the character will replan her motion as well as the completion time.

4.3 Contact force correction

Because the contact position in the dynamics system is prescribed for a fixed length, we cannot use the elapsed time to index $B(\mathbf{x}, t)$ when the final time changes during simulation. The reference index t_c is not a good candidate neither because it will cause discontinuity in contact when jumping back and forth in time. We need another time index t_d that tracks the current time of the dynamic system. Initially, t_d is the same as the elapsed time. When the final time changes, we warp the remaining time according to the current t_c, and advance t_d with a different ratio than the elapsed time. After every time step, we increment t_d by $\Delta = \frac{t_f - ht_d}{t_f - ht_c}$. If t_d and t_c are the same, $\Delta = 1$ and t_d advances at the same speed as elapsed time. When t_c jumps ahead or lags behind, Δ adjusts accordingly to catch up with t_c.

When t_c is different from t_d, the dynamic system used to compute control forces can be different from the dynamic system used for forward simulation. Direct application of the feedback control forces in simulation might cause inconsistent contact situation. We circumvent this issue by using a method similar to Muico et. al. [2009], which matches the results of control (i.e. $\dot{\mathbf{x}}$), rather than the control force itself. A simple quadratic programming (Equation (12)) solves this problem:

$$\begin{aligned} \min_{\hat{\mathbf{F}}} \; & \|\dot{\mathbf{x}}(\mathbf{x}_t, \hat{\mathbf{F}}, t_d) - \dot{\mathbf{x}}(\mathbf{x}_t, \bar{\mathbf{F}}, t_c)\|^2 \\ & \text{subject to} \quad \hat{\mathbf{F}} \in K, \end{aligned} \quad (12)$$

where $\bar{\mathbf{F}} = \mathbf{F}^* + \delta\mathbf{F}$. Finally, we can use the solution $\hat{\mathbf{F}}^*$ as control to simulate $\mathbf{x}_{t+1}$, and then we update t and t_d to the next time step.

5 Pose reconstruction

The goal of pose reconstruction is to produce a full-body pose similar to the input motion sequence and consistent with the dynamics of the abstract model. At each time step, we formulate an optimization to solve for a new joint state that matches the linear and angular momentum produced from the simulation of the abstract model. We only solve for joint velocities and use explicit Euler to update the joint configurations as $\mathbf{q}_t = \mathbf{q}_{t-1} + h\dot{\mathbf{q}}_{t-1}$. The optimization is then simplified to a quadratic programming problem. The objective function tracks the joint velocity and the foot velocity in the warped reference motion. We need to specifically track the foot motion so that it is consistent with the contact positions prescribed in the abstract model. The optimization problem is defined as follows:

$$\begin{aligned} \min_{\dot{\mathbf{q}}_t} \; & w_1\|\dot{\mathbf{q}}_t - g_1(\bar{\mathbf{Q}}, t_c, t_d)\|^2 + w_2\|\mathbf{J}_c(\mathbf{q}_t)\dot{\mathbf{q}}_t - g_2(\bar{\mathbf{Q}}, t_c, t_d)\|^2 \\ & \text{subject to} \quad \mathbf{J}(\mathbf{q}_t)\dot{\mathbf{q}}_t = \hat{\mathbf{x}}_t, \end{aligned} \quad (13)$$

where $\mathbf{J}_c$ is the Jacobian for foot contacts, $\mathbf{J}$ is the Jacobian of linear and angular momentum, and $\hat{\mathbf{x}}_t$ denotes momenta from $\mathbf{x}_t$. g_1 and g_2 compute the desired joint velocities and foot velocities respectively by warping the reference motion. w_1 and w_2 are two scalar weights that balance between these two objectives. We will discuss the selection of them in Section 6.

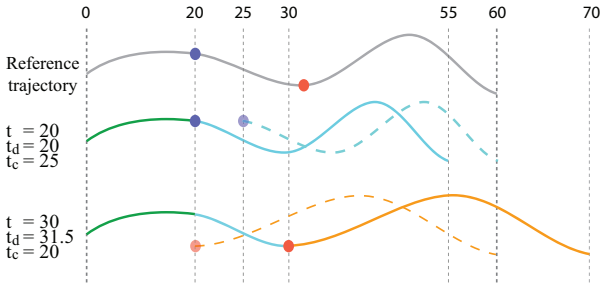


Figure 3: *Our algorithm keeps track of the remaining reference trajectory and linearly warps it in time according to t_c. Initially, t and t_d both starts from zero and advance at the same rate. Second row: at frame* 20*, the character is pushed forward and t_c jumps forward to* 25*. The remaining trajectory in shortened from* 40 *frames to* 35*. Third row:* 10 *frames later, t reaches* 30 *but t_d is at about* 31.5 *due to warping. The character now receives a backward push that delays her for* 10 *frames from the reference. t_c jumps back to* 20 *and the remaining trajectory again is warped to* 40 *frames. The reference velocity is computed for the warped trajectory at t_d.*

Trajectory warping Due to the change in final time, we need to warp the remaining trajectory in time based on the estimated remaining time (Figure 3). Function g_1 takes the reference motion $\bar{\mathbf{Q}}$, warps it according to t_c, then compute the warped velocity $\dot{\bar{\mathbf{q}}}'$ at t_d. It also tries to correct pose errors in the next time: $g_1 = \dot{\bar{\mathbf{q}}}'_{t_d} + \frac{1}{h}(\mathbf{q}_t - \bar{\mathbf{q}}_{t_d})$. Here we exclude the global translation and rotation degrees of freedom in g_1 because the global motion is determined by the abstract model. Likewise, g_2 computes the desired velocities for both the support foot and the swing foot.

Perturbation When the character receives additional external forces $\mathbf{F}^e$ such as a push, the control policy does not respond immediately until the abstract state changes at the next time step. However, the perturbed state may not respect the contacts. To help maintain contacts and balance during perturbations, we allow adjustments in the contact forces to incorporate $\mathbf{F}^e$. We solve for both $\dot{\mathbf{q}}_t$ and $\mathbf{F}$ using Equation (14) when $\mathbf{F}^e$ is present and switch back to Equation (13) when $\mathbf{F}^e$ is removed.

$$\min_{\dot{\mathbf{q}}_t, \mathbf{F}} \; w_1 \|\dot{\mathbf{q}}_t - g_3(\bar{\mathbf{Q}}, t_c, t_d, \mathbf{F}^e)\|^2 + w_2 \|\mathbf{J}_c(\mathbf{q}_t)\dot{\mathbf{q}}_t - g_2(\bar{\mathbf{Q}}, t_c, t_d)\|^2 + w_3 \|\mathbf{F} - \hat{\mathbf{F}}\|^2$$
$$\text{subject to} \quad \mathbf{J}(\mathbf{q}_t)\dot{\mathbf{q}}_t = S(\mathbf{x}_{t-1}, \mathbf{F}^e, \mathbf{F}). \quad (14)$$

This optimization modifies Equation (13) on three counts. First, because the control force $\mathbf{F}$ is also a free variable, we express the desired momenta $\hat{\mathbf{x}}_t$ in terms of the simulation function S which integrates $\mathbf{F}$ and the push $\mathbf{F}^e$ from $\mathbf{x}_{t-1}$. Second, we add one additional term to match $\mathbf{F}$ to the control forces $\hat{\mathbf{F}}$ computed from the feedback controller. w_3 weights how much to change the control compared to the tracking objectives. Third, we synthesize the impact of the push on local body parts using function g_3. It imposes the generalized impulse induced by $\mathbf{F}^e$ in each joint coordinate: $g_3 = \dot{\bar{\mathbf{q}}}'_{t_d} - h\mathbf{M}^{-1}(\mathbf{q}_t)\mathbf{J}_e^T(\mathbf{q}_t)\mathbf{F}^e$, where $\mathbf{M}$ is the inertia matrix and $\mathbf{J}_e$ is the Jacobian of the contact point. This optimization has nonlinear constraints due to the simulation function. We solve it by formulating a sequential quadratic programming using SNOPT [Gill et al. 1996].

6 Implementation details

In this section, we describe a few design choices and implementation details.

Control concatenation Although we can solve a motion of any length, in practice, we break down a long sequence into shorter segments, derive feedback controller for each segment, and concatenate them in simulation. A shorter sequence can be solved more easily and efficiently in the offline optimization. In addition, to fully take advantage of our ability to change final-state on the fly, the final state of each short sequence coincides with a key event in the input motion. For example, to generate down-stair walks, we segment a normal walk at double support phase and optimize a controller for each step. During online simulation, the final COM position is lowered for each controlled segment to guide the character walk down stairs.

To create seamless transitions from controller A to controller B, we translate the reference motion and contact positions of controller B to the desired final goal of controller A. We also linearly warp the swing foot trajectory to meet the new contact points. The same procedure can also generate walks with longer or shorter steps.

Breaking down a long sequence may introduce artificial intermediate constraints and require larger forces to meet them in a short duration. Fortunately, we can remedy these problems by overlapping controllers in time and allow the control index t_c to jump across boundaries. For instance, when we overlap two consecutive controllers A and B by 20 frames, we can start to use controller B anytime during these 20 frames in online simulation. An early transition produces smoother motion by discarding the final constraint of controller A and carrying the state errors to controller B, while a later transition respects the final constraint of controller A better. Likewise, when t_c jumps beyond the range of the current controller (i.e. Equation (9) cannot be satisfied), we continue to search the optimal t_c in the neighboring controllers. For example, suppose controller B is currently in use and the perturbation causes t_c to jump to an earlier frame beyond the first frame of controller B. In this case, we use the best gain from controller A and let it take control until t_c jumps back to the range of controller B again. The overlapping period and transition timing could be adjusted for different motions.

Objective weights Our algorithm requires tuning of only a handful of objective weights. For the offline optimization (Equation (2)), we set the weight matrix $\mathbf{W}_1$ to identity matrix, and set $\mathbf{W}_2$ and $\mathbf{W}_3$ as follows:

$$\mathbf{W}_2 = w^a \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{3\times3} \end{bmatrix}, \mathbf{W}_3 = w^t \begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{3\times3} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_{3\times3} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}.$$

In our examples, w^a and w^t are both set to 500 for normal walk and long stepping, and they are 200 and 20 for squatting. In general, larger value of w^a produces a more robust control policy, at the expense of possible larger tracking errors. Although our experiments show that a wide range of weights produce similar results, we plan to investigate inverse optimization techniques in the future to automatically design objective functions that give rise to a given reference trajectory.

The two online optimizations (Equation (13) and Equation (14)) have only three weights in total. We use $w_1 = 1, w_2 = 5,$ and $w_3 = 0.01$ in all the examples. With w_1 and w_2 fixed, w_3 controls how much to alter the optimal control force in order to satisfy the contacts and tracking. Larger value of w_3 makes the motion more compliant to the push, but also more difficult to recover.

Numerical errors Our simulation of the abstract model is physically correct up to the second-order integration error. However, matching both the COM position and momenta in the full-body

pose creates an infeasible optimization problem because we solve for only the velocity and use explicit Euler to compute configuration. In other words, when a full-body state has the same linear momentum as the abstract state, it could still produce a different COM position at the next time step. We prevent the accumulation of this numerical error by feeding back the full-body COM position to the abstract model so that the feedback control will try to correct it at every time step.

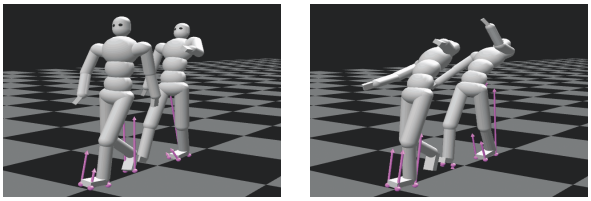
Performance We test our algorithm on a 2.8GHz Intel Core 2 Duo processor. We use motions captured at 120 Hz as input and use the same frequency for simulation. The offline optimization usually converges within 10 iterations. The actual computation time depends on the length of the motion. It takes about a minute for 60 frames of animation. For online simulation, we use a character model with 42 degrees of freedom, and the simulation runs at 20 frames per second on average.

7 Results

We demonstrate the robustness of our algorithm by building controllers for a variety of input motions. We test the feedback control policies by applying arbitrary external forces to the character, and by altering the physical properties of the environment, such as the terrain geometry and surface friction.

Change of final time A change in completion time happens almost every time a perturbation is encountered. An obvious case is when a character receives large pushes that disrupt her motion. For example, in a normal walk, a large backward push slows down a step by 10 frames while a small forward push accelerates the step by 2-3 frames. When the character receives multiple pushes, she is able to adjust her pace repeatedly on the go. In another example when the character walks upstairs of $0.2m$ height, the final time is lengthened by 4 frames, and it is shortened by 4 frames for walking down. Similarly, it takes 4 frames longer for a $0.05m$ larger step, and 17 frames faster for a shorter step. We observe similar results of timing adjustments on other motions.

A flexible plan for completion time generates more natural and robust motion. We compare our controller to a fixed-time controller on a walking motion. In the case of small pushes, our controller always produces more stable motion with smaller contact forces. For larger pushes, the character in our motion adjusts her walking speeds to recover and is finally able to complete the step, while the fixed-final-time motion failed to recover the walk (Figure 4).

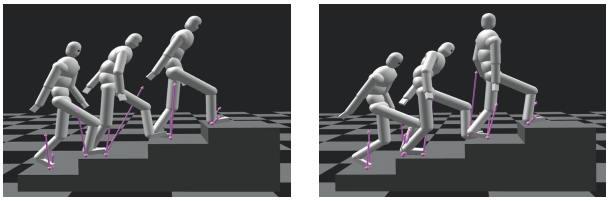


(a) flexible final time (b) fixed final time

Figure 4: *Our controller produces stable motion after a large backward push.*

Change of final constraint The ability to re-plan final constraint on the fly makes it easy for our controller to adapt to new environment and generate a larger variety of motions from a single reference. In the first experiment, we derive an optimal feedback controller for a normal walk on flat terrain and successfully apply it to walking on stairs with different step height ranging from $+0.3m$ to $-0.2m$. For walking upstairs, we change the final goal at the beginning of double support, and the character can raise her COM by as much as $0.3m$ during the double support phase. Walking down stairs is a more challenging task for our controller. The character has to twist her torso to reach the new contact points and to compensate for the angular momentum of the lower body. By simply changing the final state at the start of each step, the same controller can produce walking downstairs up to $0.2m$ per step.

We compare our results with a control policy that does not change the final constraint ahead of time. We first add an linear offset to the reference trajectory such that the final state of the trajectory meets the desired height. The control forces in this case are driven by the deviation between the current state and the modified reference trajectory, rather than the anticipation of the change in the final state. The character is able to walk down stairs with maximum step height of $0.1m$ and up stairs with maximum step height of $0.2m$, but the motions are visually unnatural in that the COM is always lagging behind the reference (Figure 5). Further, larger contact forces are used compared to our results.



(a) flexible final constraint (b) fixed final constraint

Figure 5: *Our controller produces more natural motion for walking upstairs of* $0.2m$.

Generic motion Our algorithm is generic to different types of input motion. Besides a straight walking sequence, we also apply the algorithm to a long stepping and a squat exercise. For each case, we apply random pushes to the character and observe dynamic responses and adjustments of final time. For example, when pushed backward, the long stepping takes 4 frames longer to complete and 4 frames less for a forward push. We also repeatedly push the character while she is performing a squat exercise. The character is able to balance by continuously adjust her whole body movements and the final time.

Robustness We examine the robustness of our controller by supplying pushes of different magnitudes, directions and durations. Our controller performs more robustly to pushes that do not require large change of steps. In all the examples, the character can recover from impulsive pushes (lasting less than 0.3 second) up to $200N$ in all directions. The controller is also robust against sustained pushes lasting for one second with magnitude up to $40N$.

We also compare our controller to one that tracks the reference motion without minimization of angular momentum. For reference motions with small angular momentum, both controllers perform similarly. For more dynamically challenging motions such as the long stepping, our controller exhibits better stability to small perturbations, and it can recover from extreme cases when the tracking-only controller fails. In all the tests, our controller always uses less control forces than the tracking-only controller.

8 Discussion

In conclusion, we have introduced a new technique to control and synthesize real-time character motion under physical perturbations and changes in the environment. We designed an optimal feedback

controller that allows for online re-planning of final goals and completion time. The abstract dynamic model enables incorporation of accurate global dynamics and high-level strategies such as angular momentum regulation. Our results show that allowing completion time and final constraint to vary is critical for producing robust and realistic motion. We also successfully applied our controller to various motions including walking, long stepping, and a squatting exercise.

Despite the advantages of the abstract dynamic model, the current implementation suffers from a few limitations. The major drawback is its dependency on prescribed contacts. Because the contact points are not part of the dynamic states, we cannot model the change of contacts using controls. Consequently, we can only model static frictional contact but not sliding or rolling contacts. Moreover, our controller is not able to produce different step taking behaviors from the reference, nor can it handle motions with sporadic contacts such as sparring. In the future, we want to incorporate long-term contact planning ([Whitman and Atkeson 2009]) as a separate routine in our algorithm. With the ability to plan contacts for future events, we will be able to, for example, produce ballistic motions that prepares for a safe landing when perturbed. A second drawback is the lack of knowledge of the kinematic relation between contacts and COM. As a result, the feedback control may generate kinematically impossible COM motions. Incorporating inequality constraints on the abstract states could help solve this problem.

Although our controller can handle large perturbations and changes of the environment robustly, the feedback control policy is still only an approximation of the optimality condition (Equation (6) and Equation (5)). Consequently, our controller cannot initiate drastically different strategies such as a voluntary protective step. One interesting future direction is to build a motion library for various control policies and environment, then select the most appropriate one online based on the actual situation. Liu and Atkeson [2009] have successfully employed a similar technique to standing balance. We believe our method can extend their approach to dynamic balance in locomotion.

In the future, we would like to test our algorithm on an even wider range of motions such as climbing, swinging, and quadruped motions: any maneuvers utilizing contacts. We would also like to apply our algorithm to hand-animated sequences to get plausible global physics effect. Another application is to explore more sophisticated schemes for changing final goals under user command.

Acknowledgments

We would like to thank Sumit Jain and Sehoon Ha for their help with video production and image editing. This work was supported by NSF CAREER award CCF 0742303.

A Feedback gains

We provide a compact description of our implementation on computing feedback gains in Equation (4). Please refer to Jacobson and Mayne [1970] for complete derivations of implicit final time problems (Chapter 2.3.5) and final constraint problems with inequality constraints in control (Chapter 2.5).

We denote the dynamic function (Equation (1)) as f. In our problem, the Hamiltonian is defined as $H = L + V_{\mathbf{x}}^T f$, with the objective L defined inside the integral in Equation (2). The feedback gains are computed as follows:

$$\begin{aligned}\mathbf{K}^{\mathbf{x}} &= -H_{\mathbf{FF}}^{-1} Z (H_{\mathbf{F}} + f_{\mathbf{F}}^T V_{\mathbf{xx}}),\\ \mathbf{K}^{\mu} &= -H_{\mathbf{FF}}^{-1} Z f_{\mathbf{F}}^T V_{\mathbf{x}\mu},\\ \mathbf{K}^{t} &= -H_{\mathbf{FF}}^{-1} Z f_{\mathbf{F}}^T V_{\mathbf{x}t_f},\end{aligned}$$

where

$$Z = \mathbf{I} - g_{\mathbf{F}}^T (g_{\mathbf{F}} H_{\mathbf{FF}}^{-1} g_{\mathbf{F}}^T)^{-1} g_{\mathbf{F}} H_{\mathbf{FF}}^{-1}.$$

$K = \{\mathbf{F} | g(\mathbf{F}, v) \geq \mathbf{0}\}$ approximates a static friction cone using linear combination of basis vector v. In practice, we compute coefficients λ for the linear cone and feedback forces can be expressed as: $\mathbf{F}^* + \lambda \delta \mathbf{F} \in K$.

Derivatives of V are computed by integration of the following ordinary differential equations backward from t_f at $\mathbf{X}^*$ and $\mathbf{F}^*$. These expressions are simplified for our problem.

$$\begin{aligned}\dot{V}_{\mathbf{x}} =& -H_{\mathbf{x}},\\ \dot{V}_{\mathbf{x}\mu} =& -(f_{\mathbf{x}} + f_{\mathbf{F}}\mathbf{K}^{\mathbf{x}})^T V_{\mathbf{x}\mu},\\ \dot{V}_{\mathbf{x}t_f} =& -(f_{\mathbf{x}} + f_{\mathbf{F}}\mathbf{K}^{\mathbf{x}})^T V_{\mathbf{x}t_f},\\ \dot{V}_{\mu t_f} =& V_{\mathbf{x}\mu}^T f_{\mathbf{F}} Z^T H_{\mathbf{FF}}^{-1} Z f_{\mathbf{F}}^T V_{\mathbf{x}t_f},\\ \dot{V}_{\mathbf{xx}} =& -H_{\mathbf{xx}} - f_{\mathbf{x}}^T V_{\mathbf{xx}} - V_{\mathbf{xx}} f_{\mathbf{x}},\\ &+ (H_{\mathbf{Fx}} + f_{\mathbf{F}}^T V_{\mathbf{xx}})^T Z^T H_{\mathbf{FF}}^{-1} Z (H_{\mathbf{Fx}} + f_{\mathbf{F}}^T V_{\mathbf{xx}}),\\ \dot{V}_{\mu\mu} =& V_{\mathbf{x}\mu}^T f_{\mathbf{F}} Z^T H_{\mathbf{FF}}^{-1} Z f_{\mathbf{F}}^T V_{\mathbf{x}\mu},\\ \dot{V}_{t_f t_f} =& V_{\mathbf{x}t_f}^T f_{\mathbf{F}} Z^T H_{\mathbf{FF}}^{-1} Z f_{\mathbf{F}}^T V_{\mathbf{x}t_f}.\end{aligned}$$

Boundary conditions at t_f are the following:

$$\begin{aligned}&V_{\mathbf{x}} = \Psi_{\mathbf{x}}^T \mu, \quad V_{\mathbf{x}\mu} = \Psi_{\mathbf{x}}^T, \quad V_{\mathbf{x}t_f} = H_{\mathbf{x}} + V_{\mathbf{xx}} f, \qquad V_{\mu t_f} = \Psi_{\mathbf{x}} f,\\ &V_{\mathbf{xx}} = \mathbf{0}, \qquad V_{\mu\mu} = \mathbf{0}, \qquad V_{t_f t_f} = H_{\mathbf{x}}^T f + f^T V_{\mathbf{xx}} f.\end{aligned}$$

References

ABE, Y., AND POPOVIĆ, J. 2006. Interactive animation of dynamic manipulation. In *Eurographics/SIGGRAPH Symposium on Computer Animation*.

ABE, Y., LIU, C. K., AND POPOVIĆ, Z. 2004. Momentum-based parameterization of dynamic character motion. In *Eurographics/SIGGRAPH Symposium on Computer Animation*, 173–182.

ABE, Y., DA SILVA, M., AND POPOVIĆ, J. 2007. Multiobjective control with frictional contacts. In *ACM SIGGRAPH/Eurographics symposium on Computer animation*, 249–258.

ALEXANDER, R. 1995. Simple models of human movement. *Applied Mechanics Reviews 48*, 8.

ALLEN, B. F., CHU, D., SHAPIRO, A., AND FALOUTSOS, P. 2007. On the beat!: timing and tension for dynamic characters. In *Symposium on Computer Animation*, 239–247.

BLICKHAN, R. 1989. The spring-mass model for running and hopping. *Journal of Biomechanics 22*, 11-12.

DA SILVA, M., ABE, Y., AND POPOVIĆ, J. 2008. Interactive simulation of stylized human locomotion. In *ACM Trans. on Graphics (SIGGRAPH)*, vol. 27, 1–10.

FALOUTSOS, P., VAN DE PANNE, M., AND TERZOPOULOS, D. 2001. Composable controllers for physics-based character animation. In *ACM Trans. on Graphics (SIGGRAPH)*, 251–260.

GERSHWIN, S. B., AND JACOBSON, D. H. 1968. A discrete-time differential dynamic programming algorithm with application to optimal orbit transfer. Tech. rep., Harvard University.

GILL, P., SAUNDERS, M., AND MURRAY, W. 1996. Snopt: An sqp algorithm for large-scale constrained optimization. Tech. Rep. NA 96-2, University of California, San Diego.

GOSWAMI, A., AND KALLEM, V. 2004. Rate of change of angular momentum and balance maintenance of biped robots. In *Proc. IEEE Int'l Conf on Robotics and Automation*, IEEE, 3785–3790.

HERR, H., AND POPOVIC, M. 2008. Angular momentum in human walking. In *Journal of Experimental Biology*.

HODGINS, J. K., AND POLLARD, N. S. 1997. Adapting simulated behaviors for new characters. In *ACM Trans. on Graphics (SIGGRAPH)*, 153–162.

HODGINS, J. K., WOOTEN, W. L., BROGAN, D. C., AND O'BRIEN, J. F. 1995. Animating human athletics. In *ACM Trans. on Graphics (SIGGRAPH)*, 71–78.

JACOBSON, D. H., AND MAYNE, D. Q. 1970. *Differential dynamic programming*. American Elsevier Pub. Co., New York.

JAIN, S., YE, Y., AND LIU, C. K. 2009. Optimization-based interactive motion synthesis. *ACM Trans. on Graphics 28*, 1.

KAJITA, S., KANEHIRO, F., KANEKO, K., FUJIWARA, K., HARADA, K., YOKOI, K., AND HIRUKAWA, H. 2003. Resolved momentum control: humanoid motion planning based on the linear and angular momentum. In *Intelligent Robots and Systems*, 1644–1650.

KOMURA, T., LEUNG, H., AND KUFFNER, J. 2004. Animating reactive motions for biped locomotion. In *VRST '04: Proceedings of the ACM symposium on Virtual reality software and technology*, 32–40.

KUO, A., DONELAN, J., AND RUINA, A. 2005. Energetic consequences of walking like an inverted pendulum: step-to-step transitions. *Exerc Sport Sci Rev 33*, 2.

LASZLO, J., VAN DE PANNE, M., AND FIUME, E. 1996. Limit cycle control and its application to the animation of balancing and walking. In *ACM Trans. on Graphics (SIGGRAPH)*.

LIU, C., AND ATKESON, C. 2009. Standing balance control using a trajectory library. In *Int'l Conf on Intelligent Robots and Systems (IROS)*, 3031–3036.

LIU, C. K., AND POPOVIĆ, Z. 2002. Synthesis of complex dynamic character motion from simple animations. *ACM Trans. on Graphics (SIGGRAPH) 21*, 3 (July), 408–416.

MACCHIETTO, A., ZORDAN, V., AND SHELTON, C. R. 2009. Momentum control for balance. *ACM Trans. on Graphics (SIGGRAPH) 28*, 3, 1–8.

MUICO, U., LEE, Y., POPOVIĆ, J., AND POPOVIĆ, Z. 2009. Contact-aware nonlinear control of dynamic characters. *ACM Trans. on Graphics (SIGGRAPH) 28*, 3, 1–9.

NEFF, M., AND FIUME, E. 2002. Modeling tension and relaxation for computer animation. In *Eurographics/SIGGRAPH Symposium on Computer Animation*, 81–88.

POLLARD, N. S., AND BEHMARAM-MOSAVAT, F. 2000. Force-based motion editing for locomotion tasks. In *Proceedings of the IEEE International Conference on Robotics and Automation*.

POPOVIĆ, Z., AND WITKIN, A. 1999. Physically based motion transformation. In *ACM Trans. on Graphics (SIGGRAPH)*.

POPOVIC, M., HOFMANN, A., AND HERR, H. 2004. Zero spin angular momentum control: definition and applicability. In *IEEE/RAS International Conference on Humanoid Robots*.

SHAPIRO, A., AND LEE, S.-H. 2009. Practical character physics for animators. In *ACM SIGGRAPH 2009 Talks*.

SHIRATORI, T., COLEY, B., CHAM, R., AND HODGINS, J. K. 2009. Simulating balance recovery responses to trips based on biomechanical principles. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*.

SOK, K. W., KIM, M., AND LEE, J. 2007. Simulating biped behaviors from human motion data. *ACM Trans. on Graphics (SIGGRAPH) 26*, 3 (Aug.), 107.

SPEYER, J. L., AND BRYSON, A. E. 1968. A neighboring optimum feedback control scheme based on estimated time-to-go with application to re-entry flight paths. In *AIAA Journal*, vol. 6.

STEPHENS, B., AND ATKESON, C. 2009. Modeling and control of periodic humanoid balance using the linear biped model. In *Proc. IEEE Int'l Conf. on Humanoid Robotics*.

VAN DE PANNE, M. 1997. From footprints to animation. *Computer Graphics Forum 16*, 4 (October).

WAMPLER, K., AND POPOVIĆ, Z. 2009. Optimal gait and form for animal locomotion. *ACM Trans. on Graphics 28*, 3, 1–8.

WANG, J., FLEET, D., AND HERTZMANN, A. 2009. Optimizing walking controller. *ACM Trans. on Graphics (SIGGRAPH) 28*, 5 (Dec.).

WHITMAN, E., AND ATKESON, C. G. 2009. Control of a walking biped using a combination of simple policies. In *IEEE Int'l Conf. on Humanoid Robotics*.

WOOTEN, W. L. 1998. *Simulation of Leaping, Tumbling, Landing, and Balancing Humans*. PhD thesis, Georgia Institute of Technology.

YAMANE, K., AND HODGINS, J. 2009. Simultaneous tracking and balancing of humanoid robots for imitating human motion capture data. In *Int'l Conf on Intelligent Robots and Systems (IROS)*.

YE, Y., AND LIU, C. K. 2008. Animating responsive characters with dynamic constraints in near-unactuated coordinates. *ACM Trans. on Graphics (SIGGRAPH ASIA) 27*, 5, 1–5.

YIN, K., CLINE, M. B., AND PAI, D. K. 2003. Motion perturbation based on simple neuromotor control models. In *Pacific Graphics*.

YIN, K., LOKEN, K., AND VAN DE PANNE, M. 2007. Simbicon: simple biped locomotion control. *ACM Trans. on Graphics (SIGGRAPH) 26*, 3, 105.

ZORDAN, V. B., AND HODGINS, J. K. 1999. Tracking and modifying upper-body human motion data with dynamic simulation. In *EG Workshop on Computer Animation and Simulation*.

ZORDAN, V. B., AND HODGINS, J. K. 2002. Motion capture-driven simulations that hit and react. In *Eurographics/SIGGRAPH Symposium on Computer Animation*, 89–96.

ZORDAN, V. B., MAJKOWSKA, A., CHIU, B., AND FAST, M. 2005. Dynamic response for motion capture animation. *ACM Trans. on Graphics (SIGGRAPH) 24*, 3 (July), 697–701.