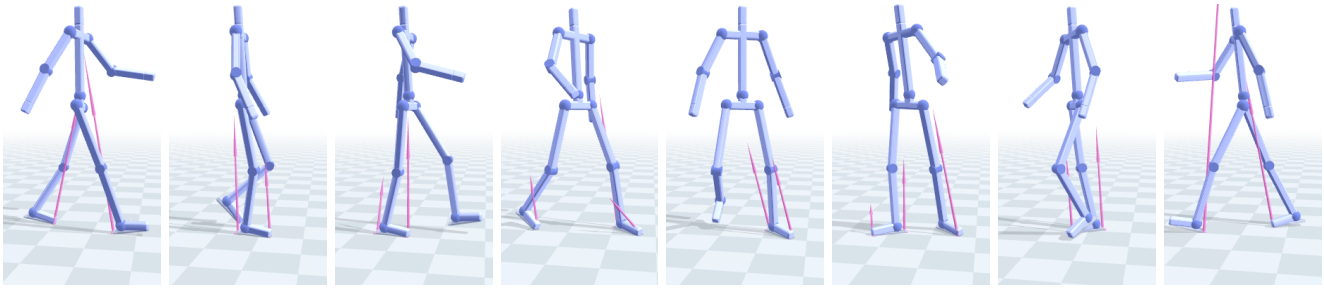
Contact-aware Nonlinear Control of Dynamic Characters

Uldarico Muico[†] Yongjoon Lee[†] Jovan Popović[†‡] Zoran Popović[†]

[†]University of Washington

[‡]Advanced Technology Labs
Adobe Systems Incorporated



Abstract

Dynamically simulated characters are difficult to control because they are underactuated—they have no direct control over their global position and orientation. In order to succeed, control policies must look ahead to determine stabilizing actions, but such planning is complicated by frequent ground contacts that produce a discontinuous search space. This paper introduces a locomotion system that generates high-quality animation of agile movements using nonlinear controllers that plan through such contact changes. We demonstrate the general applicability of this approach by emulating walking and running motions in rigid-body simulations. Then we consolidate these controllers under a higher-level planner that interactively controls the character's direction.

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

Keywords: Character simulation, character control, physics-based character animation

1 Introduction

Real-time high-fidelity character animations are primarily achieved through blending existing captured motion sequences. In order to expand the range of possible motions and to provide more natural interactions with the environment, researchers have focused on designing controllers for dynamically simulated characters. Recent years have shown some significant advances in this area [Hodgins et al. 1995; Faloutsos et al. 2001; Yin et al. 2007; da Silva et al. 2008b]. Still, the ultimate goal of controllable fully-dynamic animation that approaches the quality of motion capture systems remains unsolved for a number of key reasons.

High dimensionality. Characters have a relatively high number of degrees of freedom, making the search for the appropriate control parameters hard. Although continuous numerical optimizations can cope with large search spaces, the stringent demands of interactive applications make it clear that optimization cannot solely be performed at the time control is needed.

Underactuation. Dynamically simulated characters are difficult to control because they have no direct control over their global position and orientation. Even staying upright is a challenge for large disturbances. In order to succeed, a control law must plan ahead to determine actions that can stabilize the body.

Contacts. Characters are restricted to move within a certain region of its three-dimensional environment, and these constraints are difficult to maintain in real-time control systems. Furthermore, frequent ground contacts create a highly discontinuous search space rendering most continuous controller synthesis methods ineffective at planning over longer time horizons.

Realism. A particular character model gives rise to a large set of possible motions with different styles. Even if robust and stabilizing control laws can be found, it is challenging to construct those that reproduce the intricate and agile locomotions we observe in nature.

In this paper, we try to address these difficulties with an automatic method that infers control from motion capture. We divide our control strategy into two components: the look-ahead policy and its contact adaptation. The look-ahead component is based on a quadratic-cost formulation that assimilates an entire motion sequence to form a policy called the nonlinear quadratic regulator. The second component corrects this policy, so that its control forces are consistent with the contacts encountered at each simulation step. Together, these two components yield controllers that begin to emulate human motion in physically based simulations.

2 Related Work

Physically based simulation provides an automated framework for the synthesis of realistic motion, and today natural phenomena are automatically simulated with extreme detail and realism. Realistic character animation has been harder to attain because the control processes that generate internal muscle forces are still not known. Witkin and Kass [1988] suggested optimization as a general and automatic method for computing necessary controls. Their formulation showed that a numerical procedure could generate realistic

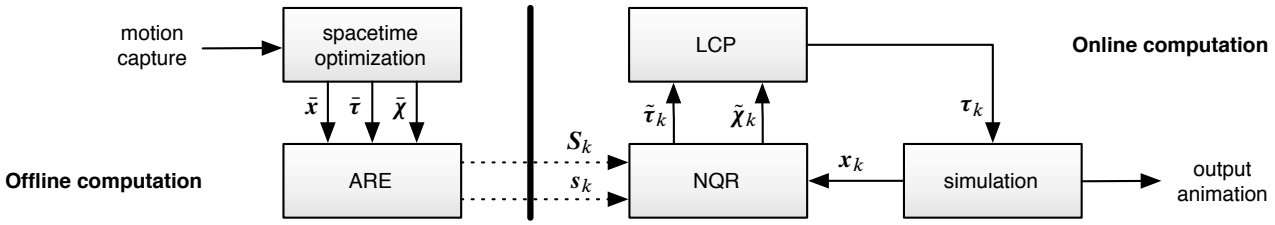


Figure 1: Control system overview. *Our system uses spacetime optimization to precompute reference motions $\bar{x}$ and controls, including both joint forces $\bar{\tau}$ and ground forces $\bar{\chi}$. It solves a system of algebraic Riccati equations (ARE) to form a nonlinear quadratic regulator (NQR) as an online look-ahead policy that anticipates and tracks the reference trajectories. NQR assimilates expected contact changes by predicting both joint controls $\tilde{\tau}$ and ground forces $\tilde{\chi}$. A solution to a linear complementarity problem (LCP) adapts these predictions to compute joint controls τ_k needed for the current contact configuration.*

motions automatically by anticipating the effects of dynamics with controls that optimize power consumption. Similar optimization techniques are now capable of generating realistic character animations [Popović and Witkin 1999; Fang and Pollard 2003; Sulejmanpasić and Popović 2005; Safonova et al. 2004; Liu et al. 2005], but these computations are too slow for interactive character animation. We use a similar optimization procedure *offline* to precompute reference controls that emulate captured motion patterns. We then construct an *online* control policy that reproduces these motions in physically based simulations despite potential disturbances.

Disturbances quickly invalidate optimized reference controls. Although spacetime optimization is too slow, optimization over a single time step can be performed online to adjust control for slow purposeful motions [Vukobratovic and Juricic 1969; Stewart and Cremer 1989; Fujimoto et al. 1998; Wieber and Chevallereau 2006; Abe et al. 2007; da Silva et al. 2008a], as often seen and implemented on walking robots [Hirai et al. 1998]. However, control of agile legged creatures requires longer anticipation to account for the behavior beyond the single step of numerical integration. Early strategies for agile control [Miura and Shimoyama 1984; Raibert 1986; Raibert and Hodgins 1991] handcrafted such controls by analyzing the behavior of simplified systems such as the spring-loaded inverted pendulum and other low-dimensional systems. Our approach also approximates the high-dimensional nonlinear dynamics, but it does so with high-dimensional linear time-varying systems. The linear dynamics makes it possible to anticipate longer-term effects automatically. In place of manual parameter tuning, our approach relies on efficient quadratic-cost optimization to compute optimized time-varying control policies. Furthermore, the same approach applies to a variety of motions without the need to adjust objective weights for different motions.

Our quadratic-cost optimization draws from the large body of literature on linear quadratic regulators that track reference trajectories [Lewis and Syrmos 1995]. Although there are alternatives, some seem difficult to apply on high-dimensional 3D characters [Ngo and Marks 1993; Tedrake 2004; Sharon and van de Panne 2005; Sok et al. 2007; Byl and Tedrake 2008], while others do not demonstrate a common search strategy for skills such as starting, stopping, walking and running [Laszlo et al. 1996; Hodgins and Pollard 1997; Westervelt et al. 2003; Yin et al. 2007; Coros et al. 2008]. Linear quadratic regulators have been explored in graphics [Brotman and Netravali 1988; da Silva et al. 2008b; Barbič and Popović 2008] and robotics [Atkeson 1994; Atkeson and Morimoto 2002; Tassa et al. 2008]. Our work adds two major improvements. First, it employs a nonlinear quadratic regulator instead of the more common linear quadratic regulator. This change allows us to control high-dimensional characters and more complex motions because it accounts for more state-dependent nonlinearities. Second, it controls both ground and joint forces making it possible to plan through contacts and over longer time periods. Our results demonstrate that these contributions make it possible to generate more realistic and more agile motions.

3 Look-ahead Policy

The look-ahead policy exploits the fact that both the dynamics equations and typical contact interactions are known well in advance. We are able to extend the farsightedness of our controllers by constructing the ground contact force predictions within a nonlinear variant of the quadratic regulator framework. By assuming typical contact interactions, this longer-term planning produces controllers that can be more easily sequenced into longer locomotion skills.

3.1 System dynamics

Our dynamics model approximates human motion with a set of rigid body limbs and constraints that roughly model the joints in the body. The pose of the simulated character is described by a set of independent joint coordinates $\boldsymbol{q} \in \mathbb{R}^n$. Human muscles generate torques $\boldsymbol{\tau} \in \mathbb{R}^{n-6}$ about each joint, leaving global position and orientation of the body as unactuated joint coordinates. Modifying those coordinates requires contact interaction with the environment.

Although our simulations incorporate the more general model with Coulomb friction, we describe the frictionless case here for the sake of simplicity. The i^{th} contact imposes the following complementarity conditions on the pose $\boldsymbol{q}$ and the contact force χ_i:

$$\chi_i \, g_i(\boldsymbol{q}) = 0, \quad \chi_i \geq 0, \quad g_i(\boldsymbol{q}) \geq 0, \tag{1}$$

where g_i is a measure of nonpenetration. This can represent either the Euclidean distance of a point to the ground or the angular distance to the articulation limits of a joint. In both cases, an active contact with $g_i(\boldsymbol{q}) = 0$ allows a constraint force $\chi_i > 0$ to push back to prevent geometric overlap. If the contact breaks with $g_i(\boldsymbol{q}) > 0$, then the force must vanish to satisfy $\chi_i \, g_i(\boldsymbol{q}) = 0$.

The dynamics of an underactuated, tree-structured articulated system are defined by differential-algebraic constraints that express the relationship between the generalized coordinates $\boldsymbol{q}$, velocities $\boldsymbol{v}$, accelerations $\dot{\boldsymbol{v}}$, inertial properties $\boldsymbol{M}$, and other forces:

$$\dot{\boldsymbol{q}} = \boldsymbol{v}, \tag{2a}$$

$$\boldsymbol{M}(\boldsymbol{q})\,\dot{\boldsymbol{v}} = \boldsymbol{h}(\boldsymbol{q}, \boldsymbol{v}) + \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{\tau} \end{bmatrix} + \mathrm{D}\boldsymbol{g}(\boldsymbol{q})^{\mathsf{T}} \boldsymbol{\chi}, \tag{2b}$$

$$\boldsymbol{\chi}^{\mathsf{T}} \boldsymbol{g}(\boldsymbol{q}) = 0, \quad \boldsymbol{\chi} \geq \boldsymbol{0}, \quad \boldsymbol{g}(\boldsymbol{q}) \geq \boldsymbol{0}. \tag{2c}$$

Note that constraints (2c) are the vector equivalent of (1) for n_{c} contacts, and the function $\boldsymbol{h} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ reflects the action of

external forces and inertial accelerations. A physically based simulator generates motion by numerically integrating these equations to generate motion. The task of a controller is to compute joint controls $\boldsymbol{\tau}$ that elicit realistic human motion.

3.2 Trajectory optimization

Our controller follows precomputed reference trajectories. By deriving such trajectories from captured sequences of human motion, the controller can emulate the same motion in physically based simulations. Captured motions themselves are not good references because of modeling disparities between the real human and the simulated dynamical system. Indeed, given the joint coordinates, velocities and accelerations from motion capture, there may be no joint and contact forces that satisfy all of the constraints (2). Instead, we compute a reference motion $\bar{\boldsymbol{q}}(t)$ and forces $\bar{\boldsymbol{\tau}}(t), \bar{\boldsymbol{\chi}}(t)$ that are consistent with the dynamical equations and as close as possible to the captured data.

Our approach follows the spacetime formulation in computer graphics literature [Witkin and Kass 1988; Cohen 1992; Liu et al. 2005]. Briefly, we minimize the deviation from captured data and also the magnitude and smoothness of the actuation torques. This optimization is subject to foot-ground contact constraints and the discretization of physics constraints as determined by a finite-difference scheme.

While this optimization accomplishes the main goal of adapting human motion to simulated dynamics, we can also modify constraints and objective functions to derive a set of related motions from a single captured sequence [Popović and Witkin 1999; Sulejmanpasić and Popović 2005]. We use this approach to generate a number of trajectories for mild turns, different speeds and stride lengths.

3.3 Optimal control

Given a reference motion and forces for a task like walking or turning, the purpose of optimal control is to compute a state-feedback policy that supplies the joint torques needed to track the reference trajectory. Doing this well is challenging because the joint torques cannot directly regulate global position and orientation of the character. They can only do so indirectly through ground contacts and the corresponding forces. Yet intermittent contacts create discontinuities that complicate such planning.

Our approach computes control policies by anticipating both the required joint torques $\boldsymbol{\tau}$ and the contact forces $\boldsymbol{\chi}$. That is, we depart from the conventional practice of controlling muscle forces only, and instead we define the control vector by

$$\boldsymbol{u} = \begin{bmatrix} \boldsymbol{\tau} \\ \boldsymbol{\chi} \end{bmatrix}. \tag{3}$$

Although one can also anticipate the constraint forces that limit the articulation of the joints, we choose to augment only the ground forces in order to specifically address the issue of underactuation. In this idealized system, the control is an unconstrained quantity, and therefore we ignore the algebraic constraints (2c). Also, the location of these anticipated contacts can be chosen arbitrarily, and in our experience they can be as simple as two points at the base of each foot. Keep in mind that the actual geometry used in the simulation is independent of this simplification and can be more elaborate.

Physically based simulations use numerical integration to approximate the system dynamics with a discrete-time system. So define the state vector $\boldsymbol{x} = (\boldsymbol{q}, \boldsymbol{v})$ to be composed of the generalized coordinates and velocities. Then the simplified dynamics is given by some discrete-time map

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{u}_k). \tag{4}$$

Our formulation requires that $\boldsymbol{f}_k : \mathbb{R}^{2n} \times \mathbb{R}^{n+n_c-6} \to \mathbb{R}^{2n}$ be affine in the control argument as follows:

$$\boldsymbol{f}_k(\boldsymbol{x}, \boldsymbol{u}) = \hat{\boldsymbol{h}}_k(\boldsymbol{x}) + \hat{\boldsymbol{G}}_k(\boldsymbol{x})\,\boldsymbol{u}. \tag{5}$$

This discretization can be satisfied by the explicit Euler method, or as in our experiments, by the semi-implicit Euler method.

Now, the goal of our control policy is to drive the states of the dynamics simulation towards a sequence of reference states and controls

$$\bar{\boldsymbol{x}}_1, \bar{\boldsymbol{u}}_1, \bar{\boldsymbol{x}}_2, \ldots, \bar{\boldsymbol{x}}_{N-1}, \bar{\boldsymbol{u}}_{N-1}, \bar{\boldsymbol{x}}_N$$

sampled at times $t_1, \ldots, t_N$. An optimal control formulation of this tracking problem can account for the dynamics of the system and maximize foresight by considering the entire trajectory:

$$\begin{aligned} \text{minimize} \quad & \|\boldsymbol{x}_N - \bar{\boldsymbol{x}}_N\|^2_{\boldsymbol{Q}_N} \\ & + \sum_{k=1}^{N-1} \|\boldsymbol{x}_k - \bar{\boldsymbol{x}}_k\|^2_{\boldsymbol{Q}_k} + \|\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k\|^2_{\boldsymbol{R}_k} + \|\boldsymbol{u}_k\|^2_{\boldsymbol{P}_k} \\ \text{over all} \quad & \boldsymbol{x}_1, \boldsymbol{u}_1, \ldots, \boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}, \boldsymbol{x}_N \\ \text{subject to} \quad & \boldsymbol{x}_{k+1} = \boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{u}_k), \quad \boldsymbol{x}_1 \text{ is fixed.} \end{aligned} \tag{6}$$

The weighting between different objective terms is usually determined by diagonal (semi-) positive definite matrices $\boldsymbol{Q}_N$, $\boldsymbol{Q}_k$, $\boldsymbol{R}_k$ and $\boldsymbol{P}_k$. For example, we observe that estimated ground forces $\bar{\boldsymbol{\chi}}$ are less reliable due to frequent contact changes, so we set the corresponding weights in $\boldsymbol{R}_k$ to zero while also penalizing large ground forces with positive weights in $\boldsymbol{P}_k$.

According to optimal control theory [Lewis and Syrmos 1995], the first-order optimality conditions require the existence of costate vectors $\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_N \in \mathbb{R}^{2n}$ satisfying

$$\boldsymbol{\lambda}_k = \mathrm{D}_{\boldsymbol{x}} \boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{u}_k)^{\mathsf{T}} \boldsymbol{\lambda}_{k+1} + \boldsymbol{Q}_k\,(\boldsymbol{x}_k - \bar{\boldsymbol{x}}_k), \tag{7a}$$

$$\boldsymbol{0} = \mathrm{D}_{\boldsymbol{u}} \boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{u}_k)^{\mathsf{T}} \boldsymbol{\lambda}_{k+1} + \boldsymbol{R}_k\,(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k) + \boldsymbol{P}_k \boldsymbol{u}_k, \tag{7b}$$

$$\boldsymbol{\lambda}_N = \boldsymbol{Q}_N\,(\boldsymbol{x}_N - \bar{\boldsymbol{x}}_N). \tag{7c}$$

Unfortunately, a real-time computation of the ensuing two-point boundary value problem is rarely possible, especially for high-dimensional characters with nonlinear dynamics. Instead, we first approximate the dynamics and then solve for a feedback policy.

3.4 Nonlinear quadratic regulator

The linearization of the system dynamics allows us to compute an optimal policy for problems with quadratic cost functions. The result is a linear policy known as the linear quadratic regulator (LQR). To obtain this control law, we define for all time indices k, the quantities

$$\begin{aligned} \boldsymbol{\xi}_k &= \boldsymbol{x}_k - \bar{\boldsymbol{x}}_k, \quad & \boldsymbol{A}_k &= \mathrm{D}_{\boldsymbol{x}} \boldsymbol{f}_k(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k), \\ \boldsymbol{\eta}_k &= \boldsymbol{u}_k - \bar{\boldsymbol{u}}_k, \quad & \boldsymbol{B}_k &= \mathrm{D}_{\boldsymbol{u}} \boldsymbol{f}_k(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k), \\ & & \boldsymbol{d}_k &= \boldsymbol{f}_k(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k) - \bar{\boldsymbol{x}}_{k+1}. \end{aligned}$$

Then consider the linearized problem corresponding to the optimal control problem (6) as follows:

$$\begin{aligned} \text{minimize} \quad & \|\boldsymbol{\xi}_N\|^2_{\boldsymbol{Q}_N} + \sum_{k=1}^{N-1} \|\boldsymbol{\xi}_k\|^2_{\boldsymbol{Q}_k} + \|\boldsymbol{\eta}_k\|^2_{\boldsymbol{R}_k} + \|\bar{\boldsymbol{u}}_k + \boldsymbol{\eta}_k\|^2_{\boldsymbol{P}_k} \\ \text{over all} \quad & \boldsymbol{\xi}_1, \boldsymbol{\eta}_1, \ldots, \boldsymbol{\xi}_{N-1}, \boldsymbol{\eta}_{N-1}, \boldsymbol{\xi}_N \\ \text{subject to} \quad & \boldsymbol{\xi}_{k+1} = \boldsymbol{A}_k \boldsymbol{\xi}_k + \boldsymbol{B}_k \boldsymbol{\eta}_k + \boldsymbol{d}_k, \quad \boldsymbol{\xi}_1 \text{ is fixed.} \end{aligned} \tag{8}$$

Note that we present a modified version of the time-varying LQR that accounts for the dynamics constraints violations represented by $\boldsymbol{d}_k$. Usually this term is zero, unless there is an disparity in finite-difference methods between the reference trajectory optimization of Section 3.2 and the discretization in Equation (5).

Following a standard derivation in the optimal control literature [Lewis and Syrmos 1995], we arrive at a costate solution given by the linear relationship

$$\boldsymbol{\lambda}_k = \boldsymbol{s}_k + \boldsymbol{S}_k \boldsymbol{\xi}_k, \tag{9}$$

where $\boldsymbol{S}_k \in \mathbb{R}^{2n \times 2n}$ and $\boldsymbol{s}_k \in \mathbb{R}^{2n}$ are defined recursively via the algebraic Riccati equations (ARE)

$$\begin{aligned}
\boldsymbol{S}_k &= \boldsymbol{Q}_k + \boldsymbol{A}_k^\mathsf{T} \left[\boldsymbol{S}_{k+1}^{-1} + \boldsymbol{B}_k \left(\boldsymbol{R}_k + \boldsymbol{P}_k \right)^{-1} \boldsymbol{B}_k^\mathsf{T} \right]^{-1} \boldsymbol{A}_k, \\
\boldsymbol{s}_k &= \boldsymbol{A}_k^\mathsf{T} \boldsymbol{s}_{k+1} + \boldsymbol{A}_k^\mathsf{T} \left[\boldsymbol{S}_{k+1}^{-1} + \boldsymbol{B}_k \left(\boldsymbol{R}_k + \boldsymbol{P}_k \right)^{-1} \boldsymbol{B}_k^\mathsf{T} \right]^{-1} \times \\
&\qquad \left[\boldsymbol{d}_k - \boldsymbol{B}_k \left(\boldsymbol{R}_k + \boldsymbol{P}_k \right)^{-1} \left(\boldsymbol{P}_k \bar{\boldsymbol{u}}_k + \boldsymbol{B}_k^\mathsf{T} \boldsymbol{s}_{k+1} \right) \right]
\end{aligned}$$

with $\boldsymbol{S}_N = \boldsymbol{Q}_N$ and $\boldsymbol{s}_N = \boldsymbol{0}$. As a result, we obtain a familiar LQR control law for problem (8) given by

$$\begin{aligned}
\boldsymbol{\eta}_k = &- \left(\boldsymbol{R}_k + \boldsymbol{P}_k + \boldsymbol{B}_k^\mathsf{T} \boldsymbol{S}_{k+1} \boldsymbol{B}_k \right)^{-1} \times \\
&\left[\boldsymbol{P}_k \bar{\boldsymbol{u}}_k + \boldsymbol{B}_k^\mathsf{T} \left(\boldsymbol{s}_{k+1} + \boldsymbol{S}_{k+1} \left(\boldsymbol{d}_k + \boldsymbol{A}_k \boldsymbol{\xi}_k \right) \right) \right].
\end{aligned} \tag{10}$$

Although this linear strategy works well on low-dimensional characters [Brotman and Netravali 1988; Atkeson and Morimoto 2002; da Silva et al. 2008b], we have found it to be unreliable for agile, high-dimensional characters.

To salvage this look-ahead mechanism, we instead derive a *nonlinear* feedback policy, referred to in this paper as the nonlinear quadratic regulator (NQR). We obtain it by using the costate solution of Equation (9) as an approximation to decouple equations (7a)–(7b). Substituting (4) and (9) into (7b) and using our assumption that $\boldsymbol{f}_k$ is affine in the control, we find

$$\begin{aligned}
\boldsymbol{0} &= \mathrm{D}_{\boldsymbol{u}} \boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{u}_k)^\mathsf{T} \boldsymbol{\lambda}_{k+1} + \boldsymbol{R}_k \left(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k \right) + \boldsymbol{P}_k \boldsymbol{u}_k \\
&= \hat{\boldsymbol{G}}_k(\boldsymbol{x}_k)^\mathsf{T} \Big[\boldsymbol{s}_{k+1} + \boldsymbol{S}_{k+1} \left(\hat{\boldsymbol{h}}_k(\boldsymbol{x}_k) + \hat{\boldsymbol{G}}_k(\boldsymbol{x}_k) \boldsymbol{u}_k - \bar{\boldsymbol{x}}_{k+1} \right) \Big] \\
&\qquad + \boldsymbol{R}_k \left(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k \right) + \boldsymbol{P}_k \boldsymbol{u}_k.
\end{aligned}$$

Solving for $\boldsymbol{u}_k$ in this equation gives us a control law that we use to predict both joint and contact forces:

$$\begin{aligned}
\tilde{\boldsymbol{u}}_k = &\left[\boldsymbol{R}_k + \boldsymbol{P}_k + \hat{\boldsymbol{G}}_k(\boldsymbol{x}_k)^\mathsf{T} \boldsymbol{S}_{k+1} \hat{\boldsymbol{G}}_k(\boldsymbol{x}_k) \right]^{-1} \times \\
&\left[\boldsymbol{R}_k \bar{\boldsymbol{u}}_k - \hat{\boldsymbol{G}}_k(\boldsymbol{x}_k)^\mathsf{T} \Big(\boldsymbol{s}_{k+1} + \boldsymbol{S}_{k+1} \left(\hat{\boldsymbol{h}}_k(\boldsymbol{x}_k) - \bar{\boldsymbol{x}}_{k+1} \right) \Big) \right].
\end{aligned} \tag{11}$$

This result is similar to those obtained using state-dependent Riccati equations [Pearson 1962; Wernli and Cook 1975]. Our formulation differs in that we assume particular affine relationships in the dynamics and costate approximations. Our problem is also non-homogeneous, as we minimize the magnitude of applied joint torques and account for any disparity between integration schemes for the reference and simulated trajectories.

4 Contact Adaptation

The nonlinear quadratic regulator is part of an idealized system that predicts joint torques and ground forces. It anticipates the activation of contacts by assuming that ground forces are directly controllable. Moreover, this system neglects to consider any nonpenetration and frictional constraints. To remove these shortcomings, contact adaptation reimposes these constraints and appropriately adjusts the predicted controls. It does so by solving a linear complementarity problem in such a way that the actual system can evolve as had been anticipated by the idealized system.

4.1 Control correction

During simulation, contact configurations will differ from those assumed by the look-ahead policy. If we ignore an untimely event such as an early heel strike, the policy can disrupt the motion by twisting the ankle or by throwing the torso forward. If we also neglect the angle limitations of critical joints like the knee or the ankles, then we cannot expect the character to behave as planned by the NQR.

To compensate for the inadequacies of the idealized system, let the vector function $\boldsymbol{\varphi}(\boldsymbol{q})$ describe the most important features to emulate. For example, it can describe the entire set of joint coordinates $\boldsymbol{\varphi}(\boldsymbol{q}) = \boldsymbol{q}$, or it can include terms containing translational coordinates of the torso and other larger body parts. Contact adaptation minimizes the difference in accelerations $\boldsymbol{\alpha} = \ddot{\boldsymbol{\varphi}}$ between simulated features $\boldsymbol{\alpha}$ and the anticipated features $\tilde{\boldsymbol{\alpha}}$:

$$\min_{\boldsymbol{\tau}} \left\| \boldsymbol{\alpha}(\boldsymbol{\tau}, \boldsymbol{\chi}) - \tilde{\boldsymbol{\alpha}}(\tilde{\boldsymbol{\tau}}, \tilde{\boldsymbol{\chi}}) \right\|_{\boldsymbol{W}}^2 + \left\| \boldsymbol{\tau} - \tilde{\boldsymbol{\tau}} \right\|_{\boldsymbol{W}_\mathrm{m}}^2, \tag{12}$$

where $\boldsymbol{W}$ and $\boldsymbol{W}_\mathrm{m}$ are symmetric positive-definite matrices. In other words, given the anticipated controls $(\tilde{\boldsymbol{\tau}}, \tilde{\boldsymbol{\chi}})$ and the simulated constraint forces $\boldsymbol{\chi}$, this optimization computes the muscle forces $\boldsymbol{\tau}$ that roughly reproduce the anticipated behavior with preferably small corrections to the muscle forces $\tilde{\boldsymbol{\tau}}$. This is a least-squares problem with a closed-form solution because the accelerations and forces are linearly related:

$$\boldsymbol{\alpha}(\boldsymbol{\tau}, \boldsymbol{\chi}) = \boldsymbol{a} + \boldsymbol{B}_\mathrm{m} \boldsymbol{\tau} + \boldsymbol{B}_\mathrm{c} \boldsymbol{\chi}, \tag{13a}$$

$$\tilde{\boldsymbol{\alpha}}(\tilde{\boldsymbol{\tau}}, \tilde{\boldsymbol{\chi}}) = \boldsymbol{a} + \boldsymbol{B}_\mathrm{m} \tilde{\boldsymbol{\tau}} + \tilde{\boldsymbol{B}}_\mathrm{c} \tilde{\boldsymbol{\chi}}. \tag{13b}$$

The vector $\boldsymbol{a}$ reflects the external and inertial forces in the system, while the matrices $\boldsymbol{B}_\mathrm{m}$, $\boldsymbol{B}_\mathrm{c}$ and $\tilde{\boldsymbol{B}}_\mathrm{c}$ reflect the action of muscle and contact forces. Note that the differences between $\boldsymbol{B}_\mathrm{c}$ and $\tilde{\boldsymbol{B}}_\mathrm{c}$ result from differences in contact geometries betweeen the idealized and simulated systems. The optimization (12) accounts for this disparity by adjusting the anticipated muscle torques according to the closed-form solution

$$\boldsymbol{\tau} = \tilde{\boldsymbol{\tau}} + \boldsymbol{C} \left(\boldsymbol{B}_\mathrm{c} \boldsymbol{\chi} - \tilde{\boldsymbol{B}}_\mathrm{c} \tilde{\boldsymbol{\chi}} \right), \tag{14}$$

where

$$\boldsymbol{C} = - \left(\boldsymbol{W}_\mathrm{m} + \boldsymbol{B}_\mathrm{m}^\mathsf{T} \boldsymbol{W} \boldsymbol{B}_\mathrm{m} \right)^{-1} \boldsymbol{B}_\mathrm{m}^\mathsf{T} \boldsymbol{W}. \tag{15}$$

This formula establishes a linear relationship between the contact forces $\boldsymbol{\chi}$ and the joint torques $\boldsymbol{\tau}$. A contact solver can subsequently use this information to compute the joint torques that are compatible with current contacts.

4.2 Linear complementarity formulation

Physically based simulations often compute contact forces by solving complementarity problems. Our adaptation policy will do the same to adjust the joint torques according to the current contact configuration.

We use a set of constraints that are a discrete-time approximation to the constraints (2). Because of the complexity of finding a solution to these nonlinear conditions, we are limited to discretizations that can be employed within the linear complementarity framework.

Following a standard formulation [Stewart and Trinkle 1996; Anitescu and Potra 1997], we determine the immediate contact force $\boldsymbol{\chi}_k$ that solves the linear complementarity problem (LCP)

$$\frac{\boldsymbol{q}_{k+1} - \boldsymbol{q}_k}{t_{k+1} - t_k} = \boldsymbol{v}_k, \tag{16a}$$

$$\boldsymbol{M}(\boldsymbol{q}_{k+1}) \frac{\boldsymbol{v}_{k+1} - \boldsymbol{v}_k}{t_{k+1} - t_k} \tag{16b}$$

$$= \boldsymbol{h}(\boldsymbol{q}_{k+1}, \boldsymbol{v}_k) + \begin{bmatrix} \boldsymbol{0} \\ \widetilde{\boldsymbol{\tau}}_k - \boldsymbol{C}(\boldsymbol{q}_{k+1}) \widetilde{\boldsymbol{B}}_{\mathrm{c}}(\boldsymbol{q}_{k+1}) \widetilde{\boldsymbol{\chi}}_k \end{bmatrix}$$

$$+ \left(\mathrm{D}\boldsymbol{g}(\boldsymbol{q}_{k+1})^{\mathsf{T}} + \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{C}(\boldsymbol{q}_{k+1}) \boldsymbol{B}_{\mathrm{c}}(\boldsymbol{q}_{k+1}) \end{bmatrix} \right) \boldsymbol{\chi}_k,$$

$$0 = \boldsymbol{\chi}_k^{\mathsf{T}} \mathrm{D}\boldsymbol{g}(\boldsymbol{q}_{k+1})\, \boldsymbol{v}_{k+1}, \tag{16c}$$

$$\boldsymbol{0} \leq \boldsymbol{\chi}_k, \tag{16d}$$

$$\boldsymbol{0} \leq \mathrm{D}\boldsymbol{g}(\boldsymbol{q}_{k+1})\, \boldsymbol{v}_{k+1}. \tag{16e}$$

This construction differs from the usual one in that we have incorporated the control law of Equation (14) directly into the dynamical constraint (16b). Handling Coulomb friction requires additional forces and imposes more complicated constraints, and we refer the reader to Appendix A and the sources for their description.

4.3 Putting it all together

Given a reference trajectory $(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k)$, we compute and store the the matrices $(\boldsymbol{S}_k, \boldsymbol{s}_k)$ that represent the look-ahead policy given by Equation (11). At every simulation step, we evaluate this policy to obtain the candidate controls $(\widetilde{\boldsymbol{\tau}}_k, \widetilde{\boldsymbol{\chi}}_k)$. Then we form the local policy (14) by computing the matrices $\boldsymbol{C}$, $\boldsymbol{B}_{\mathrm{m}}$, $\boldsymbol{B}_{\mathrm{c}}$ and $\widetilde{\boldsymbol{B}}_{\mathrm{c}}$ associated with the pose $\boldsymbol{q}_{k+1}$ of Equation (16a). At that point, we solve the LCP (16) for $\boldsymbol{\chi}_k$. This process simultaneously establishes the solution $\boldsymbol{\tau}_k$, which the simulator can use to advance to the next time frame.

An additional advantage of our approach is that simulations relying on LCPs to compute constraint forces can implement a shortcut that eliminates the need for solving two LCPs. Instead of solving one LCP for control and another for simulation, a constraint-based simulator can directly incorporate the linear expression in Equation (14) to solve a single LCP that computes contact forces.

This integrated control strategy is a consequence of the fact that the simulator itself is controlling a world that includes both the character and its environment. As a subordinate component, the look-ahead controller is largely ignorant of the complex interactions that occur within the latency of a simulation step. Just as it may be beneficial to apply constraint stabilization methods that combat drift, a simulator can be further extended to accommodate linear policies that reassert the intentions of its subordinate controllers.

5 Results

Our design methodology is applicable to a variety of mechanical models. Two-dimensional characters are easier to control but do not capture the full complexity of three-dimensional motion. We therefore focus our efforts on the control of a full-body, three-dimensional character.

5.1 Experimental setup

Our bipedal model is an articulated system with $n = 29$ degrees of freedom as shown in Figure 2. For the purpose of comparison, the inertial parameters, shown in Table 1, are similar to those found in earlier work [da Silva et al. 2008b].

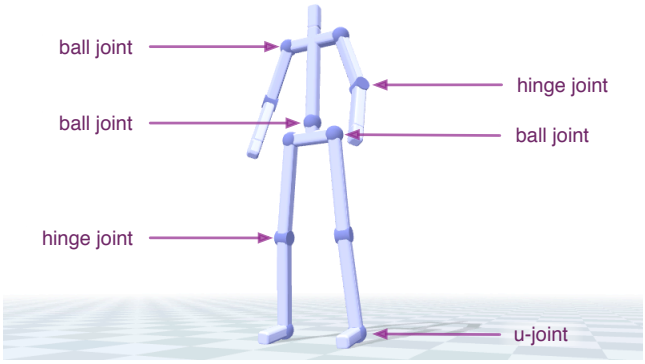


Figure 2: Character model. *Our character has 29 degrees of freedom and consists of rigid body limbs that approximate the human body.*

body	mass (kg)	I_{x} (kg · m^2)	I_{y} (kg · m^2)	I_{z} (kg · m^2)
pelvis	12.8	0.120	0.148	0.174
thigh	9.01	0.208	0.0356	0.204
shank	3.91	0.0600	8.70×10^{-3}	0.0585
foot	0.757	1.99×10^{-3}	1.99×10^{-3}	1.16×10^{-4}
torso	17.0	0.281	0.167	0.335
clavicle	2.51	6.91×10^{-3}	0.0111	0.0111
arm	1.42	0.0133	1.40×10^{-3}	0.0132
forearm	0.570	3.00×10^{-3}	3.00×10^{-4}	3.00×10^{-3}
hand	0.451	2.18×10^{-4}	1.30×10^{-5}	2.18×10^{-4}
neck	0.230	2.00×10^{-4}	2.00×10^{-4}	2.00×10^{-4}
head	4.07	0.0248	0.0176	0.0200

Table 1: System parameters. *We use a (4-sided) polyhedral approximation of the Coulomb friction cone with a static friction coefficient of* $\mu = 1.0$.

We generated various walking motions to serve as nominal trajectories. Given a kinematic sequence captured at 120 Hz, we solve for the corresponding control signals and also refine the trajectory to account for noise and modeling errors. We used the optimal control software SOCS [Betts and Huffman 1997] to perform these optimizations. This is a time-consuming process that takes a few hours to complete for each single-cycle motion clip.

The shaping of the objective used to synthesize the NQR controllers is a trial-and-error task. We simplify this process by using the same parameters for all time and all controllers. Furthermore, we only use diagonal weighting matrices. Fortunately, solving the ARE is efficient, allowing us to find a single set of parameters that produce the controllers performing to our satisfaction. Table 2 lists the weights we used in our implementation.

We use the same process for the contact adaptation component in determining the objective weights of Table 3. To further simplify matters and aid analysis, we define an adaptation parameter $\gamma > 0$, and we set the regularization weight as

$$\boldsymbol{W}_{\mathrm{m}} = \gamma^{-1} \boldsymbol{1}. \tag{17}$$

Notice that as γ vanishes, the contact adaptation mechanism becomes irrelevant, and we recover the naive scheme with $\boldsymbol{\tau} = \widetilde{\boldsymbol{\tau}}$.

5.2 Locomotion controller synthesis

Walking. We built a number of walking controllers from motion capture trajectories, and we can synthesize perpetual walkers with relative ease. Although the trajectory optimizer can compute a cyclical walking trajectory, the controller will definitely not ensure

feature	DOF	$\boldsymbol{Q}_{\mathrm{q}}$	$\boldsymbol{Q}_{\mathrm{v}}$ ($\times 10^{-2}$)	$\boldsymbol{R}$ ($\times 10^{-7}$)	$\boldsymbol{P}$ ($\times 10^{-5}$)
height	1	10	12	–	–
location	2	4.0	12	–	–
orientation	3	12	8.0	–	–
trunk joint	3	5.0	4.0	1.0	0.40
ankle joint	2	8.0	4.0	0.20	0.30
knee joint	1	10	6.0	0.40	0.30
hip joint	3	11	8.0	0.60	0.30
shoulder joint	3	2.0	2.0	1.0	0.60
elbow joint	1	2.0	2.0	1.0	0.50
heel position	3	–	–	0	0.060
toe position	3	–	–	0	0.060

Table 2: Look-ahead weights. *We use standard units* m, s *and* rad *for length, time and angles. Each foot anticipates contacts on two points: one at the heel and one at the toes.*

this periodicity during simulation. It is therefore important that the NQR algorithm be performed over many cycles. This is why the look-ahead controller is crucial. We have found that a single-cycle sweep of the ARE is inadequate, and that a longer horizon extending for a duration of two cycles is sufficient to stabilize the motion.

One of our tasks is to produce a wide variety of walking motions. An animator can, of course, capture all the possible styles of walking to obtain a complete set of controllers. On the other hand, our trajectory optimizer is capable of warping our motions within moderation to yield the controllers we need. For example, we successfully generated controllers for slower and faster walks from a single walking sequence.

Turning. We were able to produce turning controllers of varying degrees and style. Specifically, we generated 15°-, 45°-, 60°-, 90°- and 180°-turning controllers by using the same objective weights we used for straight walking. These turns are usually transition motions as opposed to cyclical motions like the straight walk. Using our spacetime optimization, we impose terminal constraints on the state of the character to match those of a typical straight-walking trajectory. Doing so allows the ankles to behave more smoothly under such common transitions, and enables direct switching between multiple turns and straight walks.

Running. We also built a few running controllers. These motions have pronounced heel-strike and toe-off phases followed by a less controllable flight phase. Due to these complications, the running controllers are more delicate and appear to be less robust on uneven ground than the usual walking controllers. Nevertheless, we produced controllers that can stabilize the runner indefinitely and in environments with small perturbations. Again, we were able to use the same parameters from the walking controllers to synthesize the running controllers.

5.3 Real-time performance

Our controllers are capable of performing in real time at a simulation frequency of 120 Hz. Even so, we found the calculation of the constraint forces in the LCP (16) to be a bottleneck in our pipeline.

Table 3: Adaptation weights. *Our contact adaptation policy emphasizes both body locations and joint configurations. Our adaptation parameter is tuned around* $\gamma = 0.02$*. During simulation, we consider four contact points per foot: two at the heel and two at the toes.*

feature	$\boldsymbol{W}$
torso height	0.010
torso location	6.0
shoulder height	0.0010
shoulder location	0.50
ankle joint	0.090
knee joint	0.20
hip joint	0.38
trunk joint	0.64
shoulder joint	0.67
elbow joint	0.54

motion	contact count	computation time (ms)		
	average	low	high	average
walking	3.3	1.0	4.1	1.6
running	1.6	0.7	4.2	1.3
standing	8.0	4.5	9.6	6.0

Table 4: Timing measurements. *Measurements are based on the following machine specifications: Mac OS X 10.5, Apple iMac* 2.4 GHz *Intel Core 2 Duo with* 2 GB *RAM.*

As is evident in Table 4, the faster motions tend to take less time on average because of infrequent contact events. In contrast, a standing controller with 8 active contacts is almost four times slower than the walking controller. In practice, it is unnecessary to consider so many contact points for such stiff motions, and the standing controller can suffice with only two points per foot. Notwithstanding, these results are dependent on the method used to solve the LCP. For our experiments, we implemented the Lemke algorithm [Cottle et al. 1992], a method that in hindsight is too precise for our purpose of controlling character animation.

Alternatively, one can choose lower simulation frequencies to achieve speed-ups. Unfortunately, this comes at the expense of accuracy and numerical stability. For example, at 60 Hz we observe that the NQR policy (11) can stabilize straight-walking motions, but it cannot execute more agile motions like turning. The LQR policy (10) is even more unstable at this frequency and diverges quickly for all motions we tested.

5.4 Motion quality and stability

We found that contact adaptation (14) improves all our controllers. Without this component, the character collapses more easily due to a failure of balancing. To see this, we consider the average cost of a simulated trajectory defined by the expression

$$\frac{1}{2N}\sum_{k=1}^{N} \|\boldsymbol{q}_k - \overline{\boldsymbol{q}}_k\|^2_{\boldsymbol{Q}_{\mathrm{q}}} + \|\boldsymbol{v}_k - \overline{\boldsymbol{v}}_k\|^2_{\boldsymbol{Q}_{\mathrm{v}}} + \|\boldsymbol{\tau}_k - \overline{\boldsymbol{\tau}}_k\|^2_{\boldsymbol{R}_{\mathrm{m}}} + \|\boldsymbol{\tau}_k\|^2_{\boldsymbol{P}_{\mathrm{m}}}, \quad (18)$$

where $\boldsymbol{Q}_{\mathrm{q}}$, $\boldsymbol{Q}_{\mathrm{v}}$, $\boldsymbol{R}_{\mathrm{m}}$ and $\boldsymbol{P}_{\mathrm{m}}$ are the same objective weights listed in Table 2. Since these parameters are fixed throughout our experiments, we can use this cost quantity to determine the fidelity of our various control systems. Figure 3 summarizes our results and confirms that contact adaptation is an essential component for sustained locomotion including sharp turning and other agile movements. In fact, all the stable regimes are nontrivial with respect to the adaptation parameter γ. That is, none of our controllers perform well in the absence of adaptation, and conversely, the motion quality is generally enhanced with diminishing costs as the adaptation strengthens within these stable basins.

As we reported earlier, the NQR policy stabilizes the character more effectively than the LQR policy. Indeed, Figure 3 reveals some of the weaknesses of the linear approach relative to our nonlinear approach. First, the elevated costs for LQR imply that it does not track the desired trajectories as faithfully as does NQR. Secondly, LQR suffers under more difficult maneuvers like the 180° turn of Figure 3(c) and in disturbed settings like that of the disturbed 45° turn of Figure 3(e). While NQR can accomplish 45° turning with even larger disturbances like a 4-cm descent, the LQR fails to stabilize the character under such stress for any value of γ. These failures are often characterized by erratic behavior at the ankles that leads to scuffing, tripping, and ultimately numerical instabilities. Lastly, we found that it is more difficult to determine a common γ that performs well across all LQR controllers. In short, NQR produces better motions, more consistently, and more reliably than LQR.

Despite these gains, there are cases where NQR and contact adaptation cannot stabilize the character on their own. Uneven or unlevel terrain, for example, can significantly disturb the rhythm of the motion. In this case, we resort to more conventional measures by resetting the phase of the controller depending on changes in the ground contacts. More specifically, since the idealized system anticipates the phase changes of the gait, we can choose to advance the controller phase upon detecting the presence or absence of contacts that characterize each phase. One can consider more sophisticated approaches [Kolter et al. 2008] to handle these disturbances, and like our simple approach, these strategies usually require that we depart from the unconditional tracking of the reference trajectory.

For uneven terrain, we encounter another obstacle. Here, the bounds on stability stem primarily from the character not adapting to lift the leg higher due to a higher ledge. For this reason, the walking controller is more robust for going downhill than uphill. For example, the turning controller successfully guided the character down a staircase with 6-cm ledges. When going uphill, the walking controller trips on the ledge due to the low-foot motion present in the natural ground walk. Ideally, the controller should discontinue tracking the trajectory and choose a different one that lifts its foot higher. The larger momentum of a runner makes it more sensitive to disturbances. Interestingly, due to the flight phase, it can handle higher uphill ledges of about 4 cm, while it is more sensitive to 2-cm downhill ledges that cause it to tumble forward.

5.5 Interactive locomotion controller

In our final experiment, we constructed an interactive controller from a collection of our contact-aware nonlinear controllers by using the value-function-based control framework [Treuille et al. 2007; McCann and Pollard 2007]. This aggregate controller consists of five walking turns and one straight step. All subcontrollers were created with the expectation that they would start from and follow into the straight-walking controller. This ensures maximal connectivity of controllers within the motion graph.

Given a user command at every instant, there are many ways for a planner to concatenate the dynamic controllers and reach the desired state. Some sequences of motions are obviously more delicate or riskier than others, and failures can arise from poor decision-making. Since the value function encapsulates the simulated connectivity between all the primitive controllers, the high-level policy can avoid the long-term failures of poorly sequenced motions. Using this optimal policy, we successfully controlled the character in real time by varying the desired direction.

6 Conclusion

We proposed a nonlinear control system that plans through contacts to emulate motion-capture data. It adapts the efficient quadratic regulator framework to derive a nonlinear look-ahead policy. This nonlinear quadratic regulator outperforms linear quadratic control, which has been previously suggested by the literature on bipedal control. We demonstrate that it can be applied to three-dimensional characters without any simplifications, and that it succeeds on sharp turns and other test cases that have stumped earlier approaches.

Look-ahead planning allows us to construct controllers that emulate long motion sequences. The method can be applied without modification to sequences with many locomotion skills including stopping, starting, turning and running. It can also be used in combination with higher-level planners for interactive control of physically animated characters. Motion capture sequences can be further adapted to produce related controllers. For example, a walking step can be shortened or extended to produce controllers for walking at different speeds.

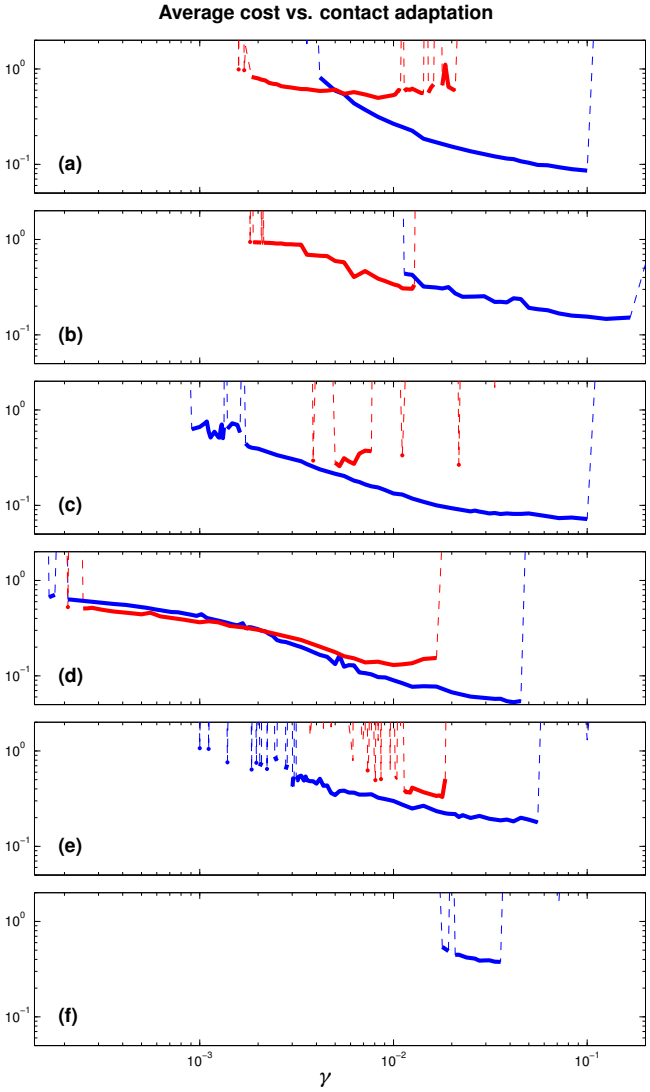


Figure 3: Endurance tests. *A few cyclical controllers were allowed to cycle for* 30 s. *We varied the adaptation parameter* γ *and computed the average cost (18) for* **(a)** *a straight-walking controller,* **(b)** *a running controller,* **(c)** *a* 180°*-turning controller, and* **(d)–(f)** *a* 45°*-turning controller. Both the LQR version (in red) and the NQR version (in blue) were evaluated. The solid segments indicate stable locomotion, while the dashed segments indicate failures. The character was undisturbed except in the cases (e) and (f), in which the character descended repeatedly by* 2 cm *and* 4 cm*, respectively. In the more extreme case of (f), the LQR policy fails to register any stable locomotion.*

We consider the high quality of final animations a strength of our control framework, but we note that it depends on two important factors. First, reference trajectories are important. At the very least, they should satisfy dynamics constraints. Ideally, they should also reflect the possibility of many different outcomes, so that they form more general building blocks for sequencing. Secondly, the parameters for the objective function still need to be tuned manually. On the positive side, tuning is fast because recursive Riccati equations are quick to solve. Furthermore, the same set of weights automatically generate time-varying control for all the motions we tried. Nevertheless, a more systematic procedure for setting these weights would be better.

A potential drawback of our control system is that it leverages LCP solves used for contact simulation. Our control system might perform worse on simulators that do not rely on LCPs for contact simulation. However, this could also be an advantage given the ubiquity of constraint-based rigid-body simulators. Our control system could connect to LCP solves within the simulator with only minor changes to the common application programming interface.

Our controller cannot recover from larger changes in the environment because that requires intentionally deviating from precomputed reference trajectories. One possibility is to compose many of our reference-tracking controllers into a single non-parametric controller that can arbitrate the tracking of different reference trajectories. Pursuing this approach or other methods for enhancing the resilience of our controllers should be tackled in the future.

Acknowledgements

This work was supported by the University of Washington Animation Research Labs (ARL), National Science Foundation (NSF) grant HCC-0811902, Intel, and Microsoft Research.

A Friction Model

Suppose we have n_f ground contacts at the k^th time frame. Then the i^th friction force $\boldsymbol{\beta}_k^i$ imposes additional linear complementarity conditions

$$0 = \boldsymbol{\beta}_k^{i\,\mathsf{T}} \left(\sigma_k^i \mathbf{e} + \boldsymbol{J}^i(\boldsymbol{q}_{k+1})\,\boldsymbol{v}_{k+1}\right), \tag{19a}$$

$$\mathbf{0} \le \boldsymbol{\beta}_k^i, \quad \mathbf{0} \le \sigma_k^i \mathbf{e} + \boldsymbol{J}^i(\boldsymbol{q}_{k+1})\,\boldsymbol{v}_{k+1}, \tag{19b}$$

$$0 = \sigma_k^i \left(\mu\chi_k^i - \|\boldsymbol{\beta}_k^i\|_1\right), \tag{19c}$$

$$0 \le \sigma_k^i, \quad 0 \le \mu\chi_k^i - \|\boldsymbol{\beta}_k^i\|_1, \tag{19d}$$

where μ is the coefficient of static friction, and $\mathbf{e}$ is a vector of ones, and $\boldsymbol{J}^i(\boldsymbol{q}_{k+1})$ provides the transformation between $\boldsymbol{\beta}_k^i$ and the corresponding body forces and torques [Stewart and Trinkle 1996; Anitescu and Potra 1997]. In this formulation, the uni-directional components of $\boldsymbol{\beta}_k^i$ come in pairs and, together with their corresponding normal components in $\boldsymbol{\chi}_k$, span a polyhedral friction cone specified by (19d). In addition to the normal forces $\boldsymbol{\chi}_k$, we must now solve for these tangential friction forces $\boldsymbol{\beta}_k^i$ and also the relative contact "speeds" σ_k^i.

Note that friction forces must also be predicted by the look-ahead policy and are therefore augmented to the control vector (3). Hence, the contact adaptation policy (14) includes additional terms to account for inconsistencies in the predicted friction forces $\widetilde{\boldsymbol{\beta}}^j$. Specifically, we have

$$\boldsymbol{\tau} = \widetilde{\boldsymbol{\tau}} + \boldsymbol{C}\left(\boldsymbol{B}_\mathrm{c}\boldsymbol{\chi} - \widetilde{\boldsymbol{B}}_\mathrm{c}\widetilde{\boldsymbol{\chi}}\right) + \boldsymbol{C}\left(\sum_{i=1}^{n_\mathrm{f}} \boldsymbol{B}_\mathrm{f}^i \boldsymbol{\beta}^i - \sum_{j=1}^{\widetilde{n}_\mathrm{f}} \widetilde{\boldsymbol{B}}_\mathrm{f}^j \widetilde{\boldsymbol{\beta}}^j\right), \tag{20}$$

where the matrices $\boldsymbol{B}_\mathrm{f}^i$ and $\widetilde{\boldsymbol{B}}_\mathrm{f}^j$ reflect the action of friction forces on the feature acceleration. Similar to the frictionless case, this policy is then substituted for $\boldsymbol{\tau}_k$ in the dynamical constraint

$$\boldsymbol{M}(\boldsymbol{q}_{k+1})\,\frac{\boldsymbol{v}_{k+1} - \boldsymbol{v}_k}{t_{k+1} - t_k} = \boldsymbol{h}(\boldsymbol{q}_{k+1}, \boldsymbol{v}_k) + \begin{bmatrix}\mathbf{0}\\ \boldsymbol{\tau}_k\end{bmatrix} + \mathrm{D}\boldsymbol{g}(\boldsymbol{q}_{k+1})^\mathsf{T}\boldsymbol{\chi}_k + \sum_{i=1}^{n_\mathrm{f}} \boldsymbol{J}^i(\boldsymbol{q}_{k+1})^\mathsf{T}\boldsymbol{\beta}_k^i.$$

Finally, an LCP solver yields the solutions $\boldsymbol{\chi}_k$ and $\boldsymbol{\beta}_k^i$ that determine the next state.

References

ABE, Y., DA SILVA, M., AND POPOVIĆ, J. 2007. Multiobjective control with frictional contacts. In *Symposium on Computer Animation (SCA)*, 249–258.

ANITESCU, M., AND POTRA, F. A. 1997. Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. nonlinear dynamics. *Nonlinear Dynamics 14*, 231–247.

ATKESON, C. G., AND MORIMOTO, J. 2002. Nonparametric representation of policies and value functions: A trajectory-based approach. In *Advances in Neural Information Processing Systems (NIPS)*, vol. 15. 1611–1618.

ATKESON, C. G. 1994. Using local trajectory optimizers to speed up global optimization in dynamic programming. In *Advances in Neural Information Processing Systems (NIPS)*, vol. 6, 663–670.

BARBIČ, J., AND POPOVIĆ, J. 2008. Real-time control of physically based simulations using gentle forces. *ACM Transactions on Graphics 27*, 4, 163:1–163:10.

BETTS, J. T., AND HUFFMAN, W. P. 1997. Sparse optimal control software SOCS. Tech. Rep. MEA-LR-085, Boeing Information and Support Services, The Boeing Co., Seattle, USA.

BROTMAN, L. S., AND NETRAVALI, A. N. 1988. Motion interpolation by optimal control. In *Computer Graphics (Proceedings of SIGGRAPH 88)*, 309–315.

BYL, K., AND TEDRAKE, R. 2008. Approximate optimal control of the compass gait on rough terrain. In *International Conference on Robotics and Automation (ICRA)*, 1258–1263.

COHEN, M. F. 1992. Interactive spacetime control for animation. In *Computer Graphics (Proceedings of SIGGRAPH 92)*, vol. 26, 293–302.

COROS, S., BEAUDOIN, P., YIN, K. K., AND VAN DE PANN, M. 2008. Synthesis of constrained walking skills. *ACM Transactions on Graphics 27*, 5, 113:1–113:9.

COTTLE, R., PANG, J., AND STONE, R. 1992. *The Linear Complementarity Problem*. Academic Press, San Diego.

DA SILVA, M., ABE, Y., AND POPOVIĆ, J. 2008. Simulation of human motion data using short-horizon model-predictive control. *Computer Graphics Forum 27*, 2, 371–380.

DA SILVA, M., ABE, Y., AND POPOVIĆ, J. 2008. Interactive simulation of stylized human locomotion. *ACM Transactions on Graphics 27*, 3, 82:1–82:10.

FALOUTSOS, P., VAN DE PANNE, M., AND TERZOPOULOS, D. 2001. Composable controllers for physics-based character animation. In *Proceedings of ACM SIGGRAPH 2001*, Annual Conference Series, 251–260.

FANG, A. C., AND POLLARD, N. S. 2003. Efficient synthesis of physically valid human motion. *ACM Transactions on Graphics 22*, 3, 417–426.

FUJIMOTO, Y., OBATA, S., AND KAWAMURA, A. 1998. Robust biped walking with active interaction control between foot and ground. In *International Conference on Robotics and Automation (ICRA)*, 2030–2035.

HIRAI, K., HIROSE, M., HAIKAWA, Y., AND TAKENAKA, T. 1998. The development of honda humanoid robot. In *International Conference on Robotics and Automation (ICRA)*, 1321–1326.

HODGINS, J. K., AND POLLARD, N. S. 1997. Adapting simulated behaviors for new characters. In *Proceedings of SIGGRAPH 97*, Computer Graphics Proceedings, Annual Conference Series, 153–162.

HODGINS, J. K., WOOTEN, W. L., BROGAN, D. C., AND O'BRIEN, J. F. 1995. Animating human athletics. In *Proceedings of ACM SIGGRAPH 95*, Annual Conference Series, 71–78.

KOLTER, J. Z., COATES, A., NG, A. Y., GU, Y., AND DUHADWAY, C. 2008. Space-indexed dynamic programming: learning to follow trajectories. In *ICML '08: Proceedings of the 25th International Conference on Machine Learning*, 488–495.

LASZLO, J. F., VAN DE PANNE, M., AND FIUME, E. L. 1996. Limit cycle control and its application to the animation of balancing and walking. In *Proceedings of SIGGRAPH 96*, Annual Conference Series, 155–162.

LEWIS, F. L., AND SYRMOS, V. L. 1995. *Optimal Control*. John Wiley & Sons, Inc., New York, NY, USA.

LIU, C. K., HERTZMANN, A., AND POPOVIĆ, Z. 2005. Learning physics-based motion style with nonlinear inverse optimization. *ACM Transactions on Graphics 24*, 3, 1071–1081.

MCCANN, J., AND POLLARD, N. 2007. Responsive characters from motion fragments. *ACM Transactions on Graphics 26*, 3, 6:1–6:7.

MIURA, H., AND SHIMOYAMA, I. 1984. Dynamic walk of a biped. *International Journal of Robotics Research 3*, 2, 60–74.

NGO, J. T., AND MARKS, J. 1993. Spacetime constraints revisited. In *Proceedings of ACM SIGGRAPH 2000*, Annual Conference Series, 343–350.

PEARSON, J. D. 1962. Approximation methods in optimal control. *Journal of Electronics and Control 13*, 453–465.

POPOVIĆ, Z., AND WITKIN, A. P. 1999. Physically based motion transformation. In *Computer Graphics (Proceedings of SIGGRAPH 99)*, Annual Conference Series, 11–20.

RAIBERT, M. H., AND HODGINS, J. K. 1991. Animation of dynamic legged locomotion. In *Computer Graphics (Proceedings of SIGGRAPH 91)*, ACM SIGGRAPH, Annual Conference Series, 349–358.

RAIBERT, M. H. 1986. *Legged Robots That Balance*. MIT Press, Cambridge, MA.

SAFONOVA, A., HODGINS, J., AND POLLARD, N. 2004. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. *ACM Transactions on Graphics 23*, 3, 514–521.

SHARON, D., AND VAN DE PANNE, M. 2005. Synthesis of controllers for stylized planar bipedal walking. In *International Conference on Robotics and Automation (ICRA)*, 2387–2392.

SOK, K. W., KIM, M., AND LEE, J. 2007. Simulating biped behaviors from human motion data. *ACM Transactions on Graphics 26*, 3, 107:1–107:9.

STEWART, A. J., AND CREMER, J. F. 1989. Algorithmic control of walking. In *International Conference on Robotics and Automation (ICRA)*, 1598–1603.

STEWART, D. E., AND TRINKLE, J. C. 1996. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. *International Journal for Numerical Methods in Engineering 39*, 15, 2673–2691.

SULEJMANPASIĆ, A., AND POPOVIĆ, J. 2005. Adaptation of performed ballistic motion. *ACM Transactions on Graphics 24*, 1, 165–179.

TASSA, Y., EREZ, T., AND SMART, W. 2008. Receding horizon differential dynamic programming. In *Advances in Neural Information Processing Systems (NIPS)*, vol. 20. MIT Press, Cambridge, MA, 1465–1472.

TEDRAKE, R. L. 2004. *Applied Optimal Control for Dynamically Stable Legged Locomotion*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA.

TREUILLE, A., LEE, Y., AND POPOVIĆ, Z. 2007. Near-optimal character animation with continuous control. *ACM Transactions on Graphics 26*, 3, 7:1–7:7.

VUKOBRATOVIC, M., AND JURICIC, D. 1969. Contribution to the synthesis of biped gait. *IEEE Transactions on Biomedical Engineering 16*, 1–6.

WERNLI, A., AND COOK, G. 1975. Suboptimal control for the nonlinear quadratic regulator problem. *Automatica 11*, 75–84.

WESTERVELT, E., GRIZZLE, J., AND KODITSCHEK, D. 2003. Hybrid zero dynamics of planar biped walkers. *IEEE Transactions on Automatic Control 48*, 1, 42–56.

WIEBER, P.-B., AND CHEVALLEREAU, C. 2006. Online adaptation of reference trajectories for the control of walking systems. *Robotics and Autonomous Systems 54*, 7, 559–566.

WITKIN, A., AND KASS, M. 1988. Spacetime constraints. In *Computer Graphics (Proceedings of SIGGRAPH 88)*, vol. 22, 159–168.

YIN, K., LOKEN, K., AND VAN DE PANNE, M. 2007. SIMBICON: Simple biped locomotion control. *ACM Transactions on Graphics 26*, 3, 105:1–105:10.