Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2010)
M. Otaduy and Z. Popovic (Editors)

Control Systems for Human Running using an Inverted Pendulum Model and a Reference Motion Capture Sequence

Taesoo Kwon[†] and Jessica Hodgins[‡]

The Robotics Institute, Carnegie Mellon University

Abstract

Physical simulation is often proposed as a way to generate motion for interactive characters. A simulated character has the potential to adapt to changing terrain and disturbances in a realistic and robust manner. In this paper, we present a balancing control algorithm based on a simplified dynamic model, an inverted pendulum on a cart. The simplified model lacks the degrees of freedom found in a full human model, so we analyze a captured reference motion in a preprocessing step and use that information about human running patterns to supplement the balance algorithms provided by the inverted pendulum controller. At run-time, the controller plans a desired motion at every frame based on the current estimate of the pendulum state and a predicted pendulum trajectory. By tracking this time-varying trajectory, our controller creates a running character that dynamically balances, changes speed and makes turns. The initial controller can be optimized to further improve the motion quality with an objective function that minimizes the difference between a planned desired motion and a simulated motion. We demonstrate the power of this approach by generating running motions at a variety of speeds ($3\,m/s$ to $5\,m/s$), following a curved path, and in the presence of disturbance forces and a skipping motion.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

1. Introduction

Simulated characters offer the promise of true interactivity by adapting not only to the commands of the user but also to changes in the environment and the behavior of other characters. Simulated control systems that produce robust and natural looking motion have proved hard to develop however, particularly for characters with the complexity of a human figure. Controllers have been designed using algorithms that span the space between tracking controllers which blindly follow a trajectory adapted from motion capture data [SKL07] to ones that derive balance algorithms from first principles [YLvdP07].

In this paper, we present an approach that provides a compromise between these two extremes. We develop a balance controller based on a simplified dynamic model of an inverted pendulum on a cart and combine that with a modified trajectory derived from a reference sequence of motion capture data. The reference motion is modified in an on-line fashion to match the dynamics of the character and the environmental constraints using the inverted pendulum model. The modified reference motion is then tracked to produce a running motion for a human figure. The resulting motion can be made more natural by using optimization to modify several control parameters (displacement maps for the foot positions and knee torques).

We demonstrate the capability of our approach by developing a controller for human running that is robust to changes in the environment and user commands. The controller is able to run at speeds ranging from 1.5 to $5\,m/s$ and turn at speeds up to $1.1\,rad/s$, run up and down slopes of $\pm 5^\circ$ and withstand pushes up to $230\,N$ having a duration of $0.4\,s$ using a single straight running motion capture sequence as the reference motion. The same algorithm can be used for generating a skipping motion which has a quite different style. We evaluate the naturalness of the resulting mo-

[†] e-mail: taesoobear@gmail.com
[‡] e-mail: jkh@cs.cmu.edu

tion by comparing to motion capture sequences of similar behaviors.

2. Related Work

Simulation of human motion has often been addressed in graphics and robotics. In this section, we discuss research that focuses on control of dynamic simulations for biped locomotion.

Many methods have been presented to control the locomotion of humanoid robots (see, for example, [HHHT98, FOK98, JKN*02, YSIT99, Sug08]) Researchers have used inverted pendulum models to generate control algorithms for locomotion [Rai86, PR92, GLP01, KNY*02, KNK*04, UK09]. Technological limitations have constrained running motions of biped humanoid robots to a slow speed ($<$ 2ms) and very short flight phase [THS09, HHHT98, KNK*04] with the exception of light-weight simplified robots [RnC*89].

Our work is motivated by the idea of the preview control from an inverted pendulum proposed by Kajita, Sugihara and their colleagues [KNK*04, Sug08]. Our approach differs from theirs in that we incorporate a reference motion capture sequence rather than a hand-designed pattern generator, and we use the inverted pendulum model to plan a continuous trajectory across multiple steps. A standard inverted pendulum model remains rooted at the stance foot until the next step.

In computer graphics, research focuses on higher-level goals such as motion quality and interactive control. Hodgins and colleagues simulated a running human as well as other athletic behaviors such as diving and vaulting [HWBO95]. They demonstrated control algorithms for a human runner at speeds between $2.5\,m/s$ and $5\,m/s$ and along a gently curving path. Faloutsos and colleagues proposed a framework for composing controllers in order to enhance the capabilities of such figures [FvdPT01]. Recently, motion capture data has frequently been used to improve the quality of simulated motions. Sok and colleagues developed an optimization method for adapting motion capture data to allow real-time simulation of a planar character [SKL07]. Yin and colleagues introduced an effective balancing controller called SIMBICON for walking and running motions [YLvdP07]. The controller is derived from an approximate inverted pendulum model to determine foot position. Their controller was later generalized to more difficult tasks [YCBvdP08,CBvdP09] and optimized to improve motion quality [WFH09]. A balance controller coupled with quadratic programming was used by da Silva and colleagues to produce a control system for a character walking on a seesaw [dSAP08]. Tsai and colleagues employed an inverted pendulum to produce a very robust controller for walking motions [TLC*09]. Macchieto and colleagues demonstrated a momentum control scheme capable of robustly balancing a standing character [MZS09]. Muico and colleagues introduced a locomotion system that generates high-quality animation of agile movements such as sharp turns [MLPP09]. Some of these approaches produce higher quality motion because they track example motions very closely while others have better generalization capability because the controllers can be reused in various situations and are robust to external perturbations. We aim to achieve both good tracking of a modified reference motion and extrapolation of the control to unobserved situations in a simple unified framework.

Our approach is most closely related to approaches using a simplified dynamics model such as a three link model [TLC*09] and an inverted pendulum [dSAP08, CBvdP10, MdLH10]. Unlike these approaches, however, we build a controller for the simple model by analyzing the captured human motion so that the resulting motion is similar to the captured human motion.

Simultaneously to our work, many papers were accepted at Siggraph this year regarding locomotion control of biped characters. Two of them are closely related to our work because they improve the generalization capability of the motion capture tracking controllers. Yuting and Liu proposed an optimal feedback controller that enhances the capability of one single motion capture sequence under various dynamically challenging conditions, and tested their algorithm on a normal walk, a long stepping, and a squat exercise [YL10]. Yoonsang and colleagues combined existing data-driven animation techniques for modulating multiple captured motions with a simple dynamic tracking controller, and demonstrated the effectiveness of their approach through interactively steering walking motions of bipeds [LKL10]. We have a different goal: the ability to deviate far from a single reference motion under perturbations and user controls, while preserving the realism in the original sequence.

3. Overview

Our approach uses a single reference motion trajectory to produce a control system for human running that is robust to a range of disturbances and can run at a variety of speeds and turning angles. We accomplish this goal by performing first an off-line analysis step to create a trajectory generator for an inverted pendulum that mimics the trajectory of the center of mass extracted from the human reference motion (Figure 1a). This pendulum trajectory generator is used as part of an on-line motion planner that takes as input the desired speed and turning rate and produces a trajectory for an inverted pendulum which is then converted into footstep locations and a desired human trajectory (Figure 1b). Finally, the human running motion is synthesized by tracking the human trajectory from the motion planner with a full dynamic model of a human (Figure 1c).

This process produces a stable running controller for the speed of the human reference trajectory. We can improve the

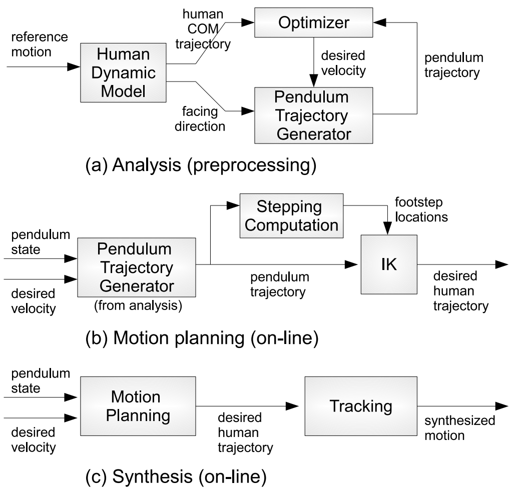


Figure 1: *Block Diagram.*

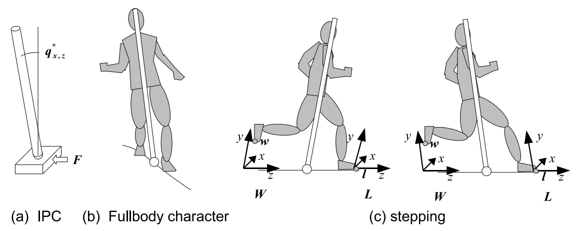


Figure 2: *Inverted pendulum and the corresponding pose of human character. (a) An inverted pendulum on a cart can be actively balanced by applying force* **F** *to the base. (b) A running motion can be characterized by a smooth continuous trajectory of the IPC. (c) Two sheared coordinate frames* **R** *and* **L** *are defined on the trajectory of IPC. Actual foot positions are represented locally.*

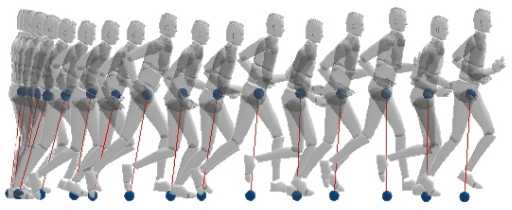


Figure 3: *Reference pendulum trajectory.*

naturalness and robustness of the computed human motion by optimizing a displacement map for some of the input parameters to the motion planner and the tracking controller. The objective function of the optimizer is the difference between the synthesized human motion and the desired human trajectory that is output from the motion planner.

By computing the trajectory for the inverted pendulum model on-line based on the current state and desired control parameters, we are able to generate motions that are significantly different from those contained in the captured reference trajectory. For example, we can generate a running motion that is almost twice as fast as the captured motion or that follows a tightly curved or sloped path although the captured motion is straight running on level ground. Our approach is also robust enough to produce a highly dynamic skipping motion.

In Section 4, we describe the off-line analysis of the reference motion. Section 5 explains the two on-line parts of our algorithm: motion planning and motion synthesis. Section 6 describes the optimization process for improving the quality and robustness of the motion. Section 7 describes our results and compares the generated motion to ground truth motion capture data for similar scenarios. Finally, Section 8 discusses the limitations of our approach and future work.

4. Motion Analysis

We analyze a captured reference motion and use the center of mass trajectory that we extract to create a trajectory generator for an inverted pendulum. This pendulum trajectory generator is used for two purposes: analysis and planning. As shown in Figure 3, the reference pendulum trajectory for the analysis step characterizes the fore-and-aft and side-to-side leaning of the human body. This reference pendulum trajectory allows us to represent the captured reference motion in a context-independent way so as to facilitate on-line modification. In planning, a new desired pendulum trajectory is generated at every time-step.

We first describe our human dynamic model, and the inverted pendulum model, and then explain how the human trajectory is reduced to the pendulum trajectory.

Our full-body character for simulation has 46 DOFs including the unactuated 6 DOFs at the pelvis and 4 sliding joints. The sliding joints are installed below the knee and hip rotational joints. Each knee and elbow is modeled using a one DOF hinge joint. Other joints have three DOFs except the lower-back and neck joint which have two DOFs and the sliding joints which have one DOF. The sliding joints together with the use of a penalty-based method for ground contacts approximate the shock absorbtion of the human body due to the compliant joints and soft tissues. The mass and inertia matrix of each body part are calculated from a surface mesh based on a uniform density assumption and the mass of the human subject. The surface mesh is manually placed such that it closely matches the motion capture marker positions.

An inverted pendulum on a cart (IPC) is a rigid body system that has two translation joints to move the cart and one rotational joint for the pendulum (Figure 2a). Because the rotational joint is unactuated, the inverted pendulum is inherently unstable and must be actively balanced by moving the cart horizontally. This property resembles the balancing actions of humans. The leaning angle of the character's body in dynamic behaviors such as turning, accelerating, or maintaining balance can be effectively modeled using the IPC.

We use a three dimensional pendulum that has a total of four degrees of freedom: two sliding joints for the cart and a ball joint with a constraint for preventing rotation about the vertical axis (yaw). With this constraint, the inverted pendulum model by itself cannot represent the facing direction of the character. Instead, we define the facing direction arbitrarily at the first frame, and update it kinematically: $\theta_{i+1}^{pend} = \theta_i^{pend} + \hat{\dot{\theta}}_i^{pend} dt$ where θ_{i+1}^{pend} is the angle between the current facing direction and a reference direction $(0,0,1)$, and $\hat{\dot{\theta}}_i^{pend}$ is the desired turning speed at frame i.

This internal representation of the forward direction is not used for the simulation of the inverted pendulum, but defines the reference coordinate to represent the leaning angle, the desired velocity of the pendulum, the foot positions and the root orientation of the character. Specifically, let $\mathbf{X}_i^{pend} = (\mathbf{x}_i, \mathbf{q}_i)$ be a pendulum configuration at frame i, where, $\mathbf{x}_i$ denotes the position of the cart at frame i. Quaternion $\mathbf{q}_i$ denotes the orientation of the pendulum:

$$\mathbf{q} = \mathbf{q}_y \mathbf{q}_{x,z}, \tag{1}$$

where $\mathbf{q}_y$ denotes the quaternion representation of facing direction θ^{pend}, and $\mathbf{q}_{x,z}$ denotes the leaning angle along the lateral and forward direction relative to $\mathbf{q}_y$. Because such a $\mathbf{q}$ decomposition is unique, for a given pendulum orientation $\mathbf{q}$, it is possible to obtain the pendulum leaning angle in global coordinate as

$$\mathbf{q}_{x,z}^* = \mathbf{q}_y \mathbf{q}_{x,z} \mathbf{q}_y^{-1}. \tag{2}$$

$\mathbf{q}_{x,z}^*$ has no vertical component and thus can be used directly for pendulum simulation.

We adopt a linear quadratic regulator for controlling the IPC model because of its well known stability properties and computational efficiency [DCA94]. We use two independent two-dimensional LQR controllers to regulate the motion of the pendulum along the forward direction and lateral direction of the character, respectively. The linear quadratic regulator controller is derived by writing linearized dynamics of the two dimensional IPC about the upright pose in a state-space form: $\dot{\mathbf{s}} = \mathbf{A}\mathbf{s} + \mathbf{B}\mathbf{u}$, where $\mathbf{s}$ is a 4-dimensional state-vector containing the position and speed of the cart, and the joint angle and angular velocity of the pendulum. $\mathbf{u}$ is the actuation force to the cart. We use the LQR matrix $\mathbf{Q} = \mathrm{diag}(0, 30000, 30000, 0)$ and $\mathbf{R} = 1$ to minimize deviations from the desired speed of the cart while minimizing the leaning angle of the pendulum and using minimal control force.

Using these two dynamic models and the LQR controller, we define a mapping from a human full-body pose to a pendulum configuration. The traditional approach in robotics is to use the center of pressure (COP) and the center of mass (COM) of the character [KKK*03]. However, such an approach is difficult to apply for running motions because the COP is not defined during flight, and moves discontinuously when the feet change contact locations.

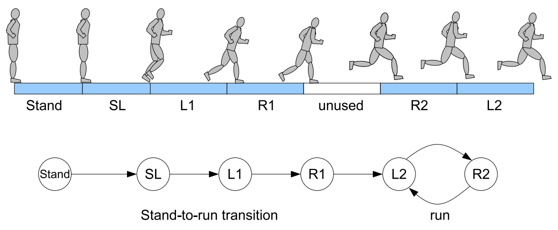


Figure 4: *Motion segments and state machine.*

We use a geometric mapping between the pendulum and the character that is intuitive and smooth. Specifically, we formulate an optimization problem where the objective is to minimize the horizontal positional differences between the COM of the character and COM of the pendulum. The unknown variables of the optimization are the key-frames of the time varying desired velocities of the IPC model. We first manually segment a given captured motion at every local maxima of the COM height so that each segment becomes a half stride (Figure 4). The forward-facing direction $(\mathbf{q}_y)_i$ at every frame i is also calculated from the vertical component of the pelvis orientation. For each motion segment s, we assign a key-frame of the desired velocity $\hat{\dot{\mathbf{x}}}_s$ represented relative to the forward direction. We used a piece-wise linear curve to produce a continuously varying desired velocity, and thus a smooth pendulum trajectory. The key-frames for the desired velocities are obtained by minimizing the following objective function:

$$\{\hat{\dot{\mathbf{x}}}_s\} = \arg\min \sum_{i \le N} \left\| \mathrm{project}(\mathbf{x}_i^{cCOM} - \mathbf{x}_i^{pCOM}) \right\|^2,$$

where $\{\hat{\dot{\mathbf{x}}}_s\}$ denotes the set of key-frames of the desired velocities, $\mathbf{x}_i^{cCOM}$ and $\mathbf{x}_i^{pCOM}$ respectively denotes the COM position of the character and pendulum at frame i. N denotes the number of frames in a motion clip, and project$(\cdots)$ discards the vertical component of the 3D vector. This optimization can be performed using a conjugate gradient algorithm within a few minutes. The gradient of the objective function is calculated numerically as a finite difference.

The resulting desired velocities define a controller for the pendulum that reproduces the center of mass trajectory of the captured reference motion. The reference pendulum trajectory for the cart near the footstep positions is obtained from the optimized simulator. By combining the simulated pendulum trajectory $\{(\mathbf{x}_i, (\mathbf{q}_{x,z}^*)_i)\}_{i<N}$ and the forward-facing direction $(\mathbf{q}_y)_i$ using Equations 1 and 2, the reference pendulum trajectory $\{(\mathbf{x}_i, \mathbf{q}_i)\}_{i<N}$ is calculated. The optimized desired velocity $\{\hat{\dot{\mathbf{x}}}_s\}$ and the turning speed $\hat{\dot{\theta}}_i^{pend}$ calculated

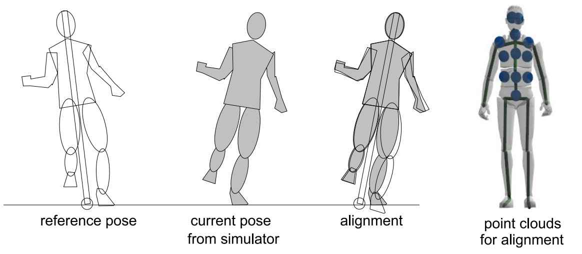


Figure 5: *A minimum-error point cloud matching algorithm is used for pendulum-state estimation.*

from the forward-facing directions are stored for use at run-time. The horizontal COM position of the pendulum and the human character are not a perfect match even with this optimization because of discrepancies between the models. We save a smoothed version of the difference $\Delta\mathbf{x}^{COM}$ represented locally to the forward direction, and use it at run-time to convert the center of mass position of a human character to that for the pendulum:

$$\Delta\mathbf{x}^{COM} = \left(\mathbf{q}_y^{-1}\cdot \text{project}(\mathbf{x}^{cCOM} - \mathbf{x}^{pCOM})\cdot \mathbf{q}_y\right), \quad (3)$$

Graph Construction: We construct a state machine consisting of six nodes (Figure 4). We separate the first two footsteps in the captured motion into the stand-to-run group because they are closer to a walking motion than a running motion, and thus the poses are significantly different from later motions. The transition between two nodes appear when the playback of a half stride is finished. Each node contains two consecutive segments that correspond to a single stride so that the half stride overlap can be used for blending at transitions.

5. Motion Synthesis

In this section, we describe how to simulate running motions of a human character using the pendulum trajectory generator obtained from the reference human motion. As shown in Figure 1b, the synthesis step consists of three components: state estimation, motion planning and tracking.

5.1. State Estimation

At every simulation step, we first estimate the current state of the pendulum and the forward facing direction by aligning a captured reference pose to the current pose of the simulated character. For the alignment, we sample 12 points on the upper-body, and use those points to match two point clouds using a minimum-error rigid transformation $\mathbf{M} = (\mathbf{t}, \mathbf{r})$ as shown in Figure 5. This alignment can be performed analytically [Hor87]. We sampled only the upper body because the leg motions are highly state-dependent. The rotational component of the transformation $\mathbf{r}$ defines the current estimate $\hat{\mathbf{q}}$ of the leaning angle and facing direction of the pendulum: $\hat{\mathbf{q}} = \mathbf{r}\bar{\mathbf{q}}$,where $\bar{\mathbf{q}}$ is the corresponding pendulum configuration from the reference pendulum trajectory.

The current estimate of the cart position is determined from the orientation estimate $\hat{\mathbf{q}}$ such that the difference between the center of mass of the pendulum and the human character becomes the same as the difference in the reference trajectories $\Delta\mathbf{x}^{COM}$ (Equation 3). The velocity of the cart and the angular velocity of the pendulum are estimated using the current frame and the previous frame (via a finite-difference calculation).

5.2. Motion Planning

From the current state estimate of the pendulum, the pendulum trajectory generator predicts a desired pendulum trajectory. Specifically, the vertical component of the pendulum orientations are first obtained by kinematic integration, and then the leaning angles are calculated using a forward dynamics simulation of the inverted pendulum on a cart.

Given the predicted pendulum trajectory, a desired human motion is generated on the fly by adding human specific characteristics such as stepping, pelvis oscillation and joint angles to the pendulum trajectory. The intuition is that a pose of a running motion depends on both past states and the future states of the pendulum. For instance, the swing foot moves in anticipation of landing, while the support foot stays at its previous position.

Let a full-body pose of the desired motion at frame i be defined by root transformation matrix $\mathbf{X}_i^G$, support foot position $\mathbf{l}_i^G$, and the swing foot position $\mathbf{w}_i^G$ in the global frame, and local joint angles $\{\theta_i^j\}$, where the actual desired pose is constructed using an analytic inverse kinematics solver [KSG02]. To resolve the redundancy in the inverse kinematics, we preserve the vertical direction and local forward direction of the feet in the captured reference motion. We will explain how we generate each element of the pose on top of the predicted pendulum trajectory in sequence.

As shown in Figure 2(c), the root transformation matrix and foot positions are obtained using coordinate frames $\mathbf{X}_i^{pend}, \mathbf{L}_i, \mathbf{W}_i$ located on the predicted pendulum trajectory:

$$\mathbf{X}_i^G = \mathbf{X}_i^{pend}\bar{\mathbf{X}}_j, \quad (4)$$

$$\mathbf{l}_i^G = \mathbf{L}_i\bar{\mathbf{l}}_j, \quad \mathbf{w}_i^G = \mathbf{W}_i\bar{\mathbf{w}}_j. \quad (5)$$

Here, $\mathbf{X}_i^{pend}$ denotes the configuration of the predicted pendulum at frame i. The frame number of the corresponding pose in the reference motion is denoted by j. $\bar{\mathbf{X}}_j$ is the local configuration for the root joint obtained from the reference human motion by assuming that it is rigidly attached to the pendulum: $\bar{\mathbf{X}}_j = \left(\bar{\mathbf{X}}_j^{pend}\right)^{-1}\bar{\mathbf{X}}_j^G$,where $\bar{\mathbf{X}}_j^{pend}$ denotes the reference pendulum configuration at frame j , and $\bar{\mathbf{X}}_j^G$ denotes the root transformation matrix of the reference human motion at frame j. For the remainders of this section, we use

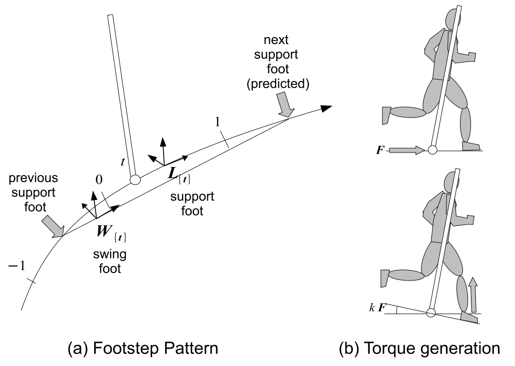


Figure 6: *(a) We generate reference coordinates $\mathbf{L}_{\{t\}}$ and $\mathbf{W}_{\{t\}}$ at phase t for foot placements by sampling the predicted pendulum trajectory. (b) An error feedback scheme to convert force $\mathbf{F}$ to torque for the fullbody character.*

overlined letters to denote properties obtained from the reference trajectories to distinguish them from those computed from the predicted/desired trajectories.

Matrix $\mathbf{L}_i$ and $\mathbf{W}_i$ are provided by a footstep pattern generator. As shown in Figure 2(c), a foot position $\bar{\mathbf{l}}_j$ is represented in a sheared coordinate frame so that the height of the foot is invariant to the pendulum leaning angle. We used the ball of the foot to indicate the foot position. Local foot positions $\bar{\mathbf{l}}_j$ and $\overline{\mathbf{w}}_j$ are obtained from the reference human motion. We first describe the footstep pattern generator and then explain how local foot positions are obtained.

Footstep pattern generation: We generate a stepping behavior by sampling the reference position and orientation of foot placement on the trajectory of the cart (Figure 6a). Let a half stride of running be defined such that it is delimited the moment when the center of mass of the runner is at its local maxima (matching our definition of a motion segment.) During a half stride, one foot is swung in the air while the other foot lands on the ground and takes off. We call the feet swing foot and support foot, respectively. The footstep pattern generator is designed such that the support foot stays at its desired position while the swing foot moves from the previous support foot position to the next foot position along a shortest path.

For notational simplicity, let us define an operator denoted by curly brackets $\{\cdot\}$ that converts locomotion phase $t, 0 \leq t \leq 1$ to frame number: $\{t\} = f + t(l - f)$ where f and l are the first and last frame of the current half stride. Then, the reference coordinate for the support foot at phase t is defined as $\mathbf{L}_{\{t\}} = \text{shear}\left(\mathbf{x}^{pend}_{\{0.5\}}, \mathbf{q}^{pend}_{\{0.5\}}\right)$, where shear$(\cdot)$ denotes a transformation matrix that shears the vertical(y) axis to the pendulum axis defined by the second argument $\mathbf{q}^{pend}_{\{0.5\}}$. The matrix is defined by a sequential multiplication of horizontal translation matrix and x, z-shearing matrix and a vertical rotation matrix. The center of the coordinate frame is fixed at $\mathbf{x}^{pend}_{\{0.5\}}$ during the half stride using the pendulum configuration at the middle frame. The amount of shearing and the vertical orientation of the foot is also defined by the pendulum at the middle frame $\mathbf{q}^{pend}_{\{0.5\}}$.

The position of the swing foot is encoded using a coordinate that linearly interpolates the nearby supporting foot coordinates. That is,

$$\mathbf{W}_{\{t\}} = \text{shear}\Big(\mathbf{x}^{pend}_{\{\text{-}0.5\}}(1-t) + \mathbf{x}^{pend}_{\{1.5\}}t, \\ \text{slerp}\left(t, \mathbf{q}^{pend}_{\{\text{-}0.5\}}, \mathbf{q}^{pend}_{\{1.5\}}\right)\Big),$$

where $\mathbf{x}^{pend}_{\{\text{-}0.5\}}$ and $\mathbf{x}^{pend}_{\{1.5\}}$ denotes the position of the cart at the middle of the previous half stride and the next half stride, respectively. For a standing motion, the foot position is defined using the pendulum at the same frame:

$$\mathbf{L}_{\{t\}} = \text{shear}\left(\mathbf{x}^{pend}_{\{t\}}, \mathbf{q}^{pend}_{\{t\}}\right). \tag{6}$$

The coordinates for foot positions on the reference pendulum trajectory $\overline{\mathbf{L}}_j$ and $\overline{\mathbf{W}}_j$ can be defined in the same manner for all frames j. Then, the desired foot position $\bar{\mathbf{l}}_j$ at the local frame can be obtained as follows:

$$\bar{\mathbf{l}}_j = \left(\overline{\mathbf{L}}_j\right)^{-1} \bar{\mathbf{l}}^G_j, \;\; \overline{\mathbf{w}}_j = \left(\overline{\mathbf{W}}_j\right)^{-1} \overline{\mathbf{w}}^G_j \tag{7}$$

where $\bar{\mathbf{l}}^G_j$ and $\overline{\mathbf{w}}^G_j$ are the global foot positions in the reference human motion at frame j. Because foot positions at the local frame $\bar{\mathbf{l}}_j$ and $\overline{\mathbf{w}}_j$ are constant given j, they are computed only once in the preprocessing step. In the later optimization step, $\bar{\mathbf{l}}_j$ and $\overline{\mathbf{w}}_j$ are further modified to produce a better controller as will be explained in Section 6.

Desired Velocity: The user can specify a motion that is different from the captured reference motion by modifying the desired speed $\hat{\dot{\mathbf{x}}}_z$ and the turning speed $\hat{\dot{\theta}}$. When the desired turning speed is modified, our controller automatically modifies the lateral desired speed $\hat{\dot{\mathbf{x}}}_x$ so that centrifugal accelerations are generated:

$$\hat{\dot{\mathbf{x}}}_z \leftarrow \hat{\dot{\mathbf{x}}}_z + \alpha, \tag{8}$$

$$\hat{\dot{\theta}} \leftarrow \hat{\dot{\theta}} + \beta, \;\; \hat{\dot{\mathbf{x}}}_x \leftarrow \hat{\dot{\mathbf{x}}}_x + c\beta, \tag{9}$$

where α is the amount of modification in forward velocity, β is the amount of modification in turning speed, $\hat{\dot{\mathbf{x}}}_x$ is the lateral desired speed. c is an empirically chosen coefficient. ($c = 3$ in our experiments).

Timing and stride adjustment: If we use the same duration for each stride regardless of the speed of the character, then the stride will become too long in fast running motions. Instead, we adjust the duration to be inversely proportional to the desired velocity and leaning angle of the pendulum: $\frac{a}{b+||\hat{\dot{\mathbf{x}}}||}$.Here $\hat{\dot{\mathbf{x}}}$ denotes the desired velocity, and b is a constant to prevent too short a stride. a is a normalizing constant to ensure that the duration matches that of the original motion at that speed. We empirically chose $b = 2$.

Torque generation: Even though the desired motion is state-dependent and physically plausible, reasonable tracking of the desired motion is not trivial because of lag. We address this problem by using an additional error feedback scheme that modifies contact forces by adjusting the desired foot position and orientation proportional to the actuation force to the cart as shown in Figure 6b. Intuitively, this modification converts the force on the cart to a torque about the cart position by stretching the leg and rotating the foot. Let $\mathbf{S}^G$ be the global transformation matrix of the desired support foot, and $\mathbf{F}$ be the global control force applied to the cart. Then, we modify the foot by rotating about the current cart position $\mathbf{x}^{pend}$ by the amount proportional to $||\mathbf{F}||$ along the axis perpendicular to $\mathbf{F}$:

$$\mathbf{S}^G \leftarrow T(\mathbf{x}^{pend}) \cdot R(\text{clamp}(k\mathbf{F}^{\perp}, 10^\circ)) \cdot T(-\mathbf{x}^{pend}) \cdot \mathbf{S}^G, \tag{10}$$

where $T(\cdot)$ denotes translation, $R(\cdot)$ denotes rotation, $\mathbf{F}^{\perp}$ denotes the vertically 90-degree rotated $\mathbf{F}$. For the support foot, k has a maximum value ($= 0.001$) at the middle of a segment, and becomes 0 at the segment boundary using a piece-wise linear curve. For the swing foot, k is set to zero. The amount of modification is clamped at 10 degrees.

5.3. Tracking

Finally, the full-body animation is generated using dynamics simulation that tracks the desired motion. We used both hybrid dynamics solver and low-gain PD-servo for tracking. The hybrid dynamics solver calculates a feed-forward torques such that a low-gain PD-servo can be used for tracking. The hybrid dynamics solver computes torques/forces for all joints except the passive root joint such that the user-given desired acceleration is satisfied:

$$\tau_{HD} = HD\left(a(\Theta_d - \Theta) + b(\dot{\Theta}_d - \dot{\Theta}) + \ddot{\Theta}_d\right), \tag{11}$$

where Θ_d and $\dot{\Theta}_d$ are the desired angles and angular velocities of all joints calculated from the desired human trajectory. The feed-forward acceleration $\ddot{\Theta}_d$ is calculated from a highly smoothed reference motion to avoid jerkiness. The low gain PD-servo provides damping and compliance that is necessary for human running.

$$\tau_{PD} = k_p(\Theta_d - \Theta) - k_d(\dot{\Theta}_d - \dot{\Theta}), \tag{12}$$

$$\tau = clamp(\tau_{HD} + \tau_{PD}, 800). \tag{13}$$

The timings for sampling the desired joint orientations Θ_d and angular velocities $\dot{\Theta}_d$ are advanced by about 40ms (5 frames) to model the tracking delays. We use the gains $a = 200, b = 30, k_p = 100, k_d = 10$ for all rotational joints and $a = 1000, b = 150, k_p = 10000, k_d = 1000$ for all sliding joints in our experiments. The resulting joint torques $\{\tau\}$ are input to a forward dynamics simulator. By looping through the above three steps of state estimation, motion planning and tracking, the controller can balance a running motion.

We used an implementation of the hybrid/forward dynamics solver based on the Lie-group formulation [PBP95]. We also verified that our scheme produces visually indistinguishable results on the well-known commercial forward simulator SD/FAST. The state estimation and motion planning step are executed at 120hz while the forward dynamics simulation uses a much higher frame rates (3000hz in our experiments). At simulation steps where motion planning is not performed, the previous desired motion is used for tracking. The desired motion is treated as a continuous function using a piecewise linear curve.

6. Optimization

In this section, we describe how we improve the quality of the simulated motion using optimization as an additional preprocessing step. Although we can produce a working controller without this additional step, the discrepancy between the simple model, that is, the inverted pendulum, and the full-body character can lead to degraded motion quality. We compensate for such errors by computing corrections to the output of the motion planner and the tracking controller using optimization.

To measure the motion quality of the simulated motion, we first generate a modified reference human trajectory using the motion planner. Because we use the motion planner to generate a reference trajectory, the optimization can be done at any speed/turning speed profiles. The objective function to minimize is the difference between the modified reference motion from the motion planner, and the simulated motion. Specifically, we measured the squared sum of pose differences and the COM-trajectory difference between the reference trajectory and a simulated motion. The pose difference is measured using the distance between two sample point clouds matched by vertically rotating and horizontally translating the second cloud to best match the first [KGP02]. Here, the sample points are evenly distributed over the entire body. The trajectory difference is also measured using the same metric. The two terms are weighted such that they have similar variance.

We optimize corrections for desired foot positions $\Delta\mathbf{l}_i, \Delta\mathbf{w}_i$ and knee torques $\Delta\tau_L, \Delta\tau_W$. Knee torque corrections are added because the foot position correction is not very effective in generating joint torques near the singularity condition of the knee joints. These combinations of corrections manipulate contact forces so that the output trajectory becomes similar to the desired trajectory. Because we used a piece-wise linear curve with three key-frames for each segment, this sedtup corresponds to a total of 48 dimensional search space when optimizing only cyclic running motions. This reduces to 24 dimensions if we assume symmetry between the left leg and the right leg. We found optimization

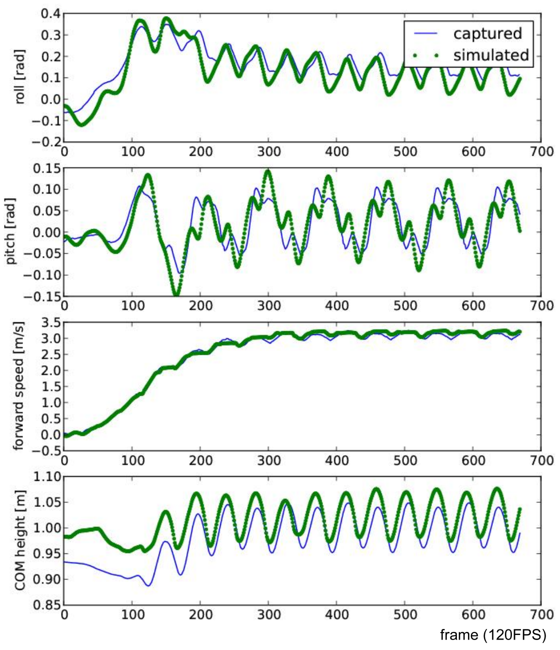


Figure 7: *Comparision between the captured motion and simulated motion.*

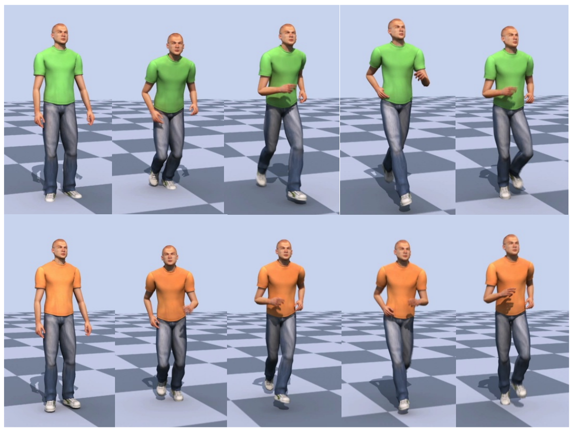


Figure 8: *The upper row (green T-shirt) contains screenshots of the input captured motion. The character with the orange T-shirt is from the simulated motion.*

in the 24-dimensional space often lead to a local minima. Therefore, we decompose it into two overlapping sets of variables, and alternate optimization of each subset. The first set optimizes for $\{\Delta\mathbf{l}_k, \Delta\mathbf{r}_k\}_{k\in\{1,2,3\}}$ (18 dimensions), and the second set optimizes for the x component (lateral direction) of the foot positions and the knee torques (12 dimensions). $k \in \{1,2,3\}$ denotes the index for key-frames. Note that $18+12 \neq 24$ because of an overlap between the two sets. The optimization was performed for ten strides of running.

Because the search space is highly non-linear and noisy, we used a randomized optimization algorithm called covariance matrix adaption evolution scheme (CMAes) [HO96]. For each subset, 200 iterations were performed, and each subset of variables was optimized two times alternately. This optimization took about 8 hours on a cluster using 8 nodes (64 cores).

7. Results

In order to validate our control algorithms, we simulate running motions with various speeds and turning rates. All of the motions are generated from a single captured reference motion of straight running at $3\,m/s$. Examples of each type of motion are shown on the accompanying video.

In the first experiment, we show how the simulated motion is similar to the captured motion. The controller is optimized to reproduce the input motion capture sequence. (The speed and turning speed were not modified by the user.) As shown in Figure 7 and 8, the two motions have similar speed profiles and appearance.

The next experiment demonstrates the variation that a controller can produce without re-optimization (Figure 9a). In the accompanying video, the character makes several turns, and then accelerates from $3\,m/s$ to $3.7\,m/s$. Figure 9c and 9d show that the same controller can be used to generate running motions on a 5-degree uphill or downhill slope.

The next example shows that our controller can recover from external disturbances. As shown in Figure 9b, three $200\,N$ forces of duration $0.2\,s$ are applied at chest height. The robustness varies depending on the pushing direction and the phase of the running cycle when the disturbance occurs. Our controller can recover from $230\,N$, $0.4\,s$ pushes (and $440\,N$, $0.2\,s$) when pushed from side. We do not know of any studies of humans experiencing external disturbances in running but based on studies of disturbances during standing balance [KP09], we believe that the magnitude of these impulses is close to what can be tolerated by humans.

The next experiment demonstrates that our controller can generate a fast running motion of $5\,m/s$. For reference, the current men's $10\,Km$ record is approximately $6.4\,m/s$. For this experiment, we re-optimized and the optimizer selected different values for $\{\Delta\mathbf{l}_k, \Delta\mathbf{r}_k, \Delta\tau_L, \Delta\tau_R\}_{k\in\{1,2,3\}}$. This set of parameter values suffices to generate a running motion that first accelerates from $3\,m/s$ to $5\,m/s$ and then maintains the speed. The optimized parameters were similar to those for $3\,m/s$ except that the z-components (forward) were increased.

The final experiment shows that our scheme works for a skipping motion without any modification. We segmented

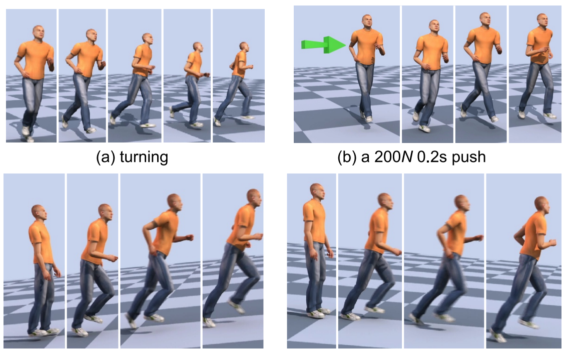


Figure 9: *Our controller for human running is robust to changes in the environment and user commands.*

	computation time	note
Analysis	129s	200 iterations
Optimization	462m	800 iterations
Synthesis	25s	10s of motion

Table 1: *Statistics on the computation times for each step of our approach.*

the captured motion as if the skipping foot is the support foot. The skipping behavior is automatically generated from the local foot positions $\mathbf{l}_i$ and $\mathbf{w}_i$ extracted from motion analysis even though we did not specifically design a skipping pattern generator.

8. Discussion

In this paper, we present an approach to constructing a control system for human running. Our approach analyzes a reference motion data sequence to extract information on human running behavior which is then used to adjust an LQR controller for an inverted pendulum. When the motion is synthesized, a motion planner is used to compute where the feet should be placed and the details of the trajectory which the dynamic simulation should track. An optimization loop adjusts a set of parameters to make the motion more robust and more natural looking.

We manually tune the weighting factors for the LQR controller and the gains for the HD-solver, PD-servo and foot rotations. Although a wide range of constants work in practice, we also have noticed that the motion quality and the robustness of the controller vary depending on the choices of constants. We observed that increasing the gains improved robustness at the expense of motion quality and execution speed (smaller simulation timestep required). A more through analysis of this trade-off would be an interesting area for future work.

We used only two sets of optimized parameters, one for fast running and a second for all other experiments. We can easily optimize to compute many different sets of parameters (e.g. grid sampling along the dimensions of forward and turning speed). It is possible that interpolating among those parameters would generate a controller that is more robust to rapid accelerations and decelerations.

Because we do not enforce collisions between the two legs, the legs may intersect when the character is pushed or is running fast around a curve. We believe that this problem could be fixed by adding a term to the objective function of the optimization and allowing the baseline (footstep spacing) of the running motion to be a parameter in the optimization.

Our control system is based on an inverted pendulum, and therefore it uses the placement of the foot for balance recovery. Humans often use arm motions or upper-body motions to assist in recovery. The use of a more complex model such as a double inverted pendulum or reaction wheel pendulum might further improve the robustness of our scheme and allow motion to be mapped onto the upper body. However, the cost and difficulty of the analysis and optimization would increase.

The current implementation of our scheme is about twice as slow as real-time. However, we believe that a speedup can be achieved by code optimization. The main bottlenecks are the prediction of the future pendulum trajectory performed at 120 hz and the use of a scripting language for easier debugging. The control might be equally robust at a lower frequency of motion planning which would reduce the required computation time significantly.

Our controller is "blind" in that it does not have any knowledge about the slope, terrain or obstacles. The adaption of the foot position to the environment, particularly the anticipation of slopes, would likely further improve the robustness of our control.

In the work reported here, we have focused on running motions. However, an inverted pendulum model is central to many dynamic balancing tasks and we believe that this approach could be readily adapted to standing, walking and hopping motions. We also plan to explore controllers of a similar design for more complicated dynamic behaviors such as gymnastics.

Acknowledgements

We thank Junggon Kim for providing his dynamics simulator, Moshe Mahler for modeling the character and video editing, and Justin Macey for motion capture sessions. We also thank Autodesk for their donation of the 3D animation and rendering package Maya. This work was partially supported by the Korea Research Foundation Grant funded by the Korean Government (KRF-2008-357-D00224).

References

[CBvdP09] Coros S., Beaudoin P., van de Panne M.: Robust task-based control policies for physics-based characters. *ACM TOG 28*, 5 (2009), 1–9. 2

[CBvdP10] Coros S., Beaudoin P., van de Panne M.: Generalized biped walking control. In *ACM TOG* (2010). 2

[DCA94] Dorato P., Cerone V., Abdallah C.: *Linear-Quadratic Control: An Introduction*. Simon & Schuster, 1994. 4

[dSAP08] da Silva M., Abe Y., Popović J.: Interactive simulation of stylized human locomotion. *ACM TOG 27*, 3 (2008), 1–10. 2

[FOK98] Fujimoto Y., Obata S., Kawamura A.: Robust biped walking with active interaction control between foot and ground. In *ICRA* (1998), pp. 2030–2035. 2

[FvdPT01] Faloutsos P., van de Panne M., Terzopoulos D.: Composable controllers for physics-based character animation. In *SIGGRAPH* (2001), pp. 251–260. 2

[GLP01] Gienger M., Löffler K., Pfeiffer F.: Towards the design of a biped jogging robot. In *ICRA* (2001), pp. 4140–4145. 2

[HHHT98] Hirai K., Hirose M., Haikawa Y., Takenaka T.: The development of honda humanoid robot. In *ICRA* (1998), pp. 1321–1326. 2

[HO96] Hansen N., Ostermeier A.: Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *International Conference on Evolutionary Computation* (1996), pp. 312–317. 8

[Hor87] Horn B. K. P.: Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A 4*, 4 (1987), 629–642. 5

[HWBO95] Hodgins J. K., Wooten W. L., Brogan D. C., O'Brien J. F.: Animating human athletics. In *SIGGRAPH* (1995), pp. 71–78. 2

[JKN*02] Jr. J. J. K., Kagami S., Nishiwaki K., Inaba M., Inoue H.: Dynamically-stable motion planning for humanoid robots. *Auton. Robots 12*, 1 (2002), 105–118. 2

[KGP02] Kovar L., Gleicher M., Pighin F.: Motion graphs. *ACM TOG 21*, 3 (2002), 473–482. 7

[KKK*03] Kajita S., Kanehiro F., Kaneko K., Fujiwara K., Harada K., Yokoi K., Hirukawa H.: Biped walking pattern generation by using preview control of zero-moment point. In *ICRA* (2003), pp. 1620–1626. 4

[KNK*04] Kajita S., Nagasaki T., Kaneko K., Yokoi K., Tanie K.: A hop towards running humanoid biped. In *ICRA* (2004), IEEE, pp. 629–635. 2

[KNY*02] Kajita S., Nagasaki T., Yokoi K., Kaneko K., Tanie K.: Running pattern generation for a humanoid robot. In *ICRA* (2002), pp. 2755–2761. 2

[KP09] Kim S., Park S.: Human postural response to linear perturbation. *The Korean Society of Mechanical Engineers A*, 33 (2009), 27–33. 8

[KSG02] Kovar L., Schreiner J., Gleicher M.: Footskate cleanup for motion capture editing. In *Proceedings of the 2002 ACM SIGGRAPH Symposium on Computer Animation* (2002), pp. 97–104. 5

[LKL10] Lee Y., Kim S., Lee J.: Data-driven biped control. In *ACM TOG* (2010). 2

[MdLH10] Mordatch I., de Lasa M., Hertzmann A.: Robust physics-based locomotion using low-dimensional planning. In *ACM TOG* (2010). 2

[MLPP09] Muico U., Lee Y., Popović J., Popović Z.: Contact-aware nonlinear control of dynamic characters. *ACM TOG 28*, 3 (2009), 1–9. 2

[MZS09] Macchietto A., Zordan V., Shelton C. R.: Momentum control for balance. *ACM TOG 28*, 3 (2009), 1–8. 2

[PBP95] Park F. C., Bobrow J. E., Ploen S. R.: A lie group formulation of robot dynamics. *I. J. Robotic Res 14*, 6 (1995), 609–618. 7

[PR92] Playter R. R., Raibert M. H.: Control of a biped somersault in 3d. In *IROS* (1992), IEEE, pp. 582–589. 2

[Rai86] Raibert M. H.: Legged robots. *Communications of the ACM, June 1986 9*, 6 (1986). 2

[RnC*89] Raibert M. H., njamin H. B. B., Chepponis M., Koechling J., Hodgins J. K., Dustman D., Brennan W. K., Barrett D. S., Thompson C. M., Hebert J. D., Lee W., Borvansky L.: Dynamically stable legged locomotion. progress report: September 1985-september 1989, 1989. 2

[SKL07] Sok K. W., Kim M., Lee J.: Simulating biped behaviors from human motion data. *ACM TOG 26*, 3 (2007), 107. 1, 2

[Sug08] Sugihara T.: Simulated regulator to synthesize ZMP manipulation and foot location for autonomous control of biped robots. In *ICRA* (2008), pp. 1264–1269. 2

[THS09] Tajima R., Honda D., Suga K.: Fast running experiments involving a humanoid robot. In *ICRA* (2009), pp. 1571–1576. 2

[TLC*09] Tsai Y.-Y., Lin W.-C., Cheng K. B., Lee J., Lee T.-Y.: Real-time physics-based 3d biped character animation using an inverted pendulum model. *IEEE TVCG 99* (2009), 325–337. 2

[UK09] Ugurlu B., Kawamura A.: Real-time running and jumping pattern generation for bipedal robots based on ZMP and euler's equations. In *IROS* (2009), IEEE, pp. 1100–1105. 2

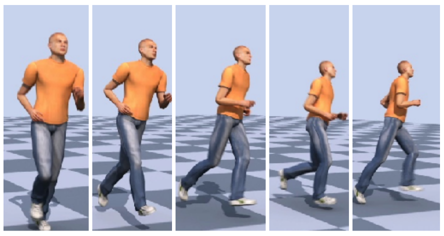
[WFH09] Wang J. M., Fleet D. J., Hertzmann A.: Optimizing walking controllers. *ACM TOG 28*, 5 (2009), 1–8. 2

[YCBvdP08] Yin K., Coros S., Beaudoin P., van de Panne M.: Continuation methods for adapting simulated skills. *ACM TOG 27*, 3 (2008), 1–7. 2

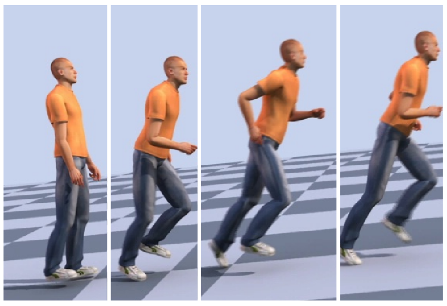
[YL10] Ye Y., Liu C. K.: Optimal feedback control for character animation using an abstract model. In *ACM TOG* (2010). 2

[YLvdP07] Yin K., Loken K., van de Panne M.: Simbicon: simple biped locomotion control. *ACM TOG 26*, 3 (2007), 105. 1, 2

[YSIT99] Yamaguchi J., Soga E., Inoue S., Takanishi A.: Development of a bipedal humanoid robot: Control method of whole body cooperative dynamic biped walking. In *ICRA* (1999), pp. 368–374. 2


(a) turning

(b) a 200*N* 0.2s push


(c) 5-degrees uphill slope

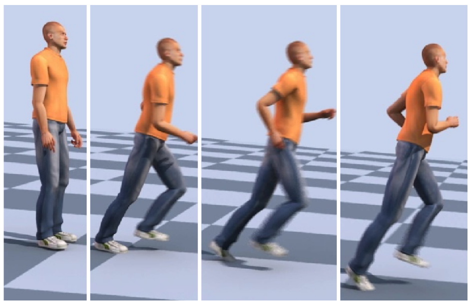

(d) 5-degrees downhill slope

Figure 10: *Our controller for human running is robust to changes in the environment and user commands.*