Biped Controller for Character Animation

KangKang Yin, kkyin@sfu.ca,
Simon Fraser University
Stelian Coros, scoros@cmu.edu,
Carnegie Mellon University
Michiel van de Panne, van@cs.ubc.ca,
University of British Columbia

Abstract. In this chapter, we first overview the common methods for building biped controllers in physics-based character animation. Then we explain in detail two closely related biped controllers: SIMBICON and GENBICON. THe SIMple BIped locomotion CONtrol (SIMBICON) strategy adopts a simple linear feedback strategy for foot placement to maintain balance during locomotion. The GENeralized BIped walking CONtrol (GENBICON) strategy improves the balance control using an inverted pendulum model for foot placement, in conjunction with Jacobian-transpose control for velocity fine-tuning and gravity compensation for all limb movement. Both SIMBICON and GENBICON use proportional-derivative joint servos to track a desired motion style, which can be interactively edited by users. The major advantages of such biped controllers include simplicity, robustness, and directable styles. Lastly, we discuss our ongoing efforts towards building more versatile and robust controllers with minimal prior knowledge.

Keywords: biped control, physics-based character animation, motion control, balance feedback, motion capture, inverted pendulum, Jacobian transpose control, foot placement

1 Introduction

Character animation has mainly relied on skilled artists and motion capture in the past. The problems with these traditional methods include the high effort required to author such animations, and the inability to reuse such animations on different models and in dynamic environments. Physics-simulated bipeds together with appropriate control strategies are a principled approach that can provide better generalization and interactivity with guaranteed physical realism. These have thus become a promising avenue for character animation Geijtenbeek and Pronost (2012).

Controlling bipedal motion is, however, extremely challenging and it lies at the intersection of diverse disciplines, including neuroscience, biomechanics, and robotics. More specifically, the control difficulties for bipeds arise from non-linear dynamics, under-actuated systems, and the obscure nature of human control strategies. Thus, motion capture remains the prevalent method for achieving high quality character animation in the film and game industry today. Ragdoll simulations, or passive character simulations, have also been widely adopted, as these

avoid the difficult control problem. Nevertheless, the problem of physics-based biped control has seen tremendous progress in recent years. Under-constrained motions such as balancing and walking, as well as highly-dynamic skills such as parkour style terrain crossing and gymnastics, can now be simulated and controlled in real time with a motion quality that is nearly indistinguishable from motion capture data Liu et al. (2016).

In this chapter, we descrube two closely-related biped locomotion controller that are easy to implement and stylize, and that are robust to perturbations and parameter changes Yin et al. (2007); Coros et al. (2010). To begin, we first overview the major state-of-the-art methods within the context of physics-based character animation.

2 State of the Art

Building biped controllers started around the late 1980s in the computer graphics community. However, controlling humanoids has been of long term interest in robotics. Recently, the control of physics-based bipeds is also a topic of interest in the machine learning community. Due to limited space, we only focus on the literature from the perspective of physics-based character animation.

2.1 Trajectory Optimization

Trajectory optimization techniques compute kinematic motion trajectories that are optimal with respect to some chosen criteria, such as energy consumption Witkin and Kass (1988); Al Borno et al. (2013). The physics, including equations of motion and contacts, are incorporated as constraints for the optimization problem. Such methods are usually hard to scale to complex models and long motions, due to the nature of large-scale nonlinear optimization. Moreover, the user only has limited control over the style of the optimized motion, through the manipulation of the terms and their weights that define the objective function. Balance is also handled implicitly via the optimization and physics constraints, and the resulting motion is therefore not robust with respect to perturbations, as required by interactive applications. The solutions of these optimizations cannot directly run in a forward dynamics simulator, and need to be coupled with forward dynamics with another layer of short-horizon optimization to achieve interactive control Macchietto et al. (2009); de Lasa et al. (2010); Hämäläinen et al. (2015).

2.2 Controllers with Explicit Balance Control

An alternative approach to controller development is to structure control policies around desired motion styles with robust balance strategies Hodgins et al. (1995); Yin et al. (2007); Coros et al. (2010); Lee et al. (2010). The desired motion styles can be specified through keyframes or imported from motion capture data. The balance strategy can be as simple as hand-designed linear feedback

laws, or can use nonlinear models such as those defined by inverted pendulum models. The major advantage of such biped controllers are their robustness. The same controller can usually run on different simulation platforms without any retuning Giovanni and Yin (2011); DART.

In this chapter, we will focus on this class of biped controllers with explicit balance control, due to their simplicity, flexibility, and robustness. We describe two closely-related locomotion controllers: SIMBICON (SIMple BIped locomotion CONtrol) ; and GENBICON (GENeralized BIped walking CONtrol) . These both adopt a swing foot placement strategy to maintain balance during locomotion.

2.3 Tracking Controllers

With good reference to human motion data available through 3D motion capture, building biped controllers that can imitate reference motion trajectories provides a good starting point for building skilled and agile bipeds. Model-based optimal control, an extension of trajectory optimization, provides a general method for developing control about given reference trajectories (Muico et al., 2009). Relatedly, the SAMCON (Sampling-based Motion Control) algorithms method Liu et al. (2010, 2012, 2013, 2016), learns time-indexed controls around the desired reference motions. These algorithms start by sampling control actions for short duration motion fragments. The basic SAMCON Liu et al. (2010) and the improved SAMCON Liu et al. (2015) methods reconstruct open-loop controls; and the guided SAMCON Liu et al. (2016) then builds robust linear feedback strategies around these open-loop controls. When coupled with Control Graphs, a general mechanism for organizing multiple motion skills and their transitions, the guided learning framework enables real-time control of multiple characters, each capable of a diverse range of robust movement skills, including locomotion, highly dynamic kicks and gymnastics, standing, rising motions, and in-between transitions. Generally speaking, tracking controllers produce motions of high quality, but are less robust in terms of dealing with perturbations.

3 SIMBICON: SIMple BIped locomotion CONtrol

SIMBICON presents a simple and robust strategy for the control of balance during locomotion for 2D and 3D physically-simulated characters. It enables a large variety of gaits and styles, including walking in all directions (forwards, backwards, sideways, turning), running, skipping, and hopping. Controllers can be authored using a small number of parameters, or their construction can be informed by motion capture data. Their robustness is demonstrated with respect to pushes in all directions, unexpected steps and slopes, and unexpected variations in kinematic and dynamic parameters. The key to the success of SIMBICON is a feedback term that continuously modifies the swing hip target angle as a linear function of the Center of Mass (CoM) position and velocity. This provides a robust balancing behavior by changing the future point of support.

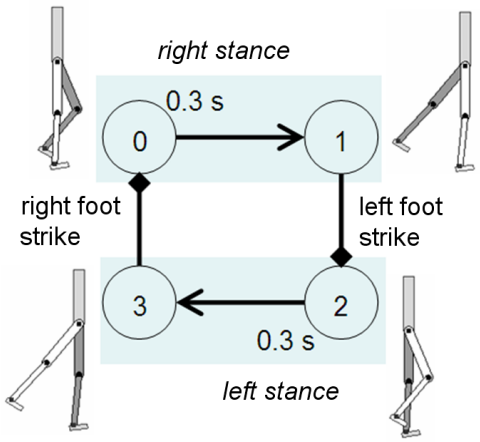

Fig. 1: Finite state machine for walking.

3.1 Base Control

In physics-based character simulation, the base controller provides default target joint angles to joint-level PD (Proportional-Derivative) servos. The style of walking is thus regulated by the base controller.

The SIMBICON base control is based on Finite State Machines (FSM) , with each state having its own target pose for internal joint angles, as shown in Figure 1. For symmetric gaits, there are pairs of left-right symmetric states, e.g., states 0 and 2, and states 1 and 3. Transitions between states occur after an elapsed time, e.g., state transition $0 \rightarrow 1$, or after foot contact, e.g., $1 \rightarrow 2$. If a foot contact has already occurred before entering a state having an outbound foot-contact transition, then the controller spends no time in that state. In any given state, in order to drive each joint to its desired local angle, the joints apply torques computed by PD control:

$$\tau = k_p(\theta_d - \theta) - k_d\dot{\theta}. \tag{1}$$

The poses represent a desired set of joint angles and are typically not actually achieved by the joints in question. For example, while in state 1 in Figure 1, the biped's pose in practice has its swing leg extended forwards. However, its target state has the swing leg extended backwards and thus has a net effect of moving the swing leg backwards and down, bringing it into down and into contact with the ground.

The choice of the number of states reflects the detail with which to model the various phases of a locomotion gait. We use four states to model our walking gaits, consisting of two symmetric walking steps. Each step has two states, the first of which lifts the swing foot upwards and forwards for a fixed duration of time, and the second of which drives the swing foot towards the ground until

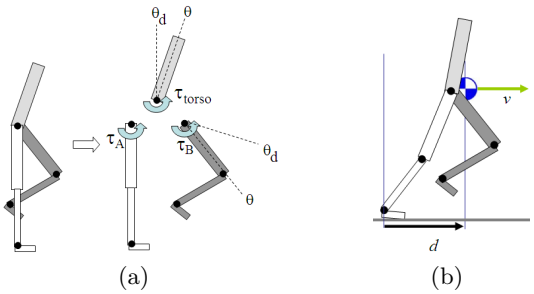

Fig. 2: Elements of the balance control strategy: (a) Relationship between torso, stance-hip, and swing-hip torques; (b) Center-of-mass position and velocity.

contact is made. This model is capable of many different walking styles, both forwards and backwards. The FSM states also serve as a coarse model of the phase of the motion when switching between controllers. Thus, if a request is made to switch from one walk style to another, this is done by transitioning from state n of one controller to state $n+1$ of another controller. For this reason, while our running gaits can be modeled using simple two-state controllers (one for each running step), we add two zero-duration dummy states in order to have the same four-state structure as for the walking gaits. This allows for transitions between walking and running gaits. Our skipping controller has 8 states, reflecting its more complex sequence of actions.

Torso and Swing-hip Control The stance hip and swing hip are handled separately, as illustrated in Figure 2(a). First, there is a need to control the orientation of the torso with respect to the world frame. This can be accomplished using a virtual PD controller that operates in the world frame to compute a net torso torque τ_{torso}, as shown in the figure. Second, there is also a need to decouple swing foot positioning from the current torso pitch angle. This is accomplished by also controlling the swing leg (via the swing hip) with respect to the world coordinate frame. The swing hip torque, τ_B, is thus also computed using a virtual PD controller that operates in the world frame. Last, there is a requirement that the virtual torques, τ_{torso} and τ_B, be realisable using only internal torques. We require that the desired value of τ_{torso} is in fact the net torque seen by the torso, $-\tau_A - \tau_B$. This is accomplished by computing the stance hip torque as

$$\tau_A = -\tau_{torso} - \tau_B. \tag{2}$$

3.2 Linear Foot Placement

The key component of the SIMBICON control strategy is to apply a balance feedback strategy to the swing foot placement. We employ a linear feedback law

of the form

$$\theta_d = \theta_{d0} + c_d d + c_v v \tag{3}$$

to the swing hip, where θ_d is the target angle used for PD control at any point in time, θ_{d0} is the default fixed target angle as described in the FSM, d is the horizontal distance from the stance ankle to the center of mass (CoM) as shown in Figure 2(b), and v is the velocity of the center of mass. The midpoint of the hips can be used as a simple and effective proxy for estimating the position and velocity of the center of mass. We use this simplification in both 2D and 3D.

The feedback gain parameter c_d is important for providing balance during low-speed gaits or in-place stepping. Consider a situation for an in-place (desired zero velocity) walking gait with current velocity $v = 0$, and two possible CoM positions $d_a = +10cm$, $d_b = -10cm$. In the first case, there is a need to step forward quickly, while in the second case there is a need to step backwards quickly in order to recover balance. The combination of (d, v) provides complete information about the current position in the gait cycle, i.e., the current phase, whereas v alone only provides information with regards to the current velocity error.

In order to extend the control scheme to 3D, the control strategy is applied in both the sagittal and coronal planes. Balance feedback in the coronal plane uses an analogous measure of d, v in order to make alterations to the lateral placement of the swing foot using the swing hip.

The balance feedback can be extended more generally to multiple joints using the form

$$\theta_{\mathbf{d}} = \theta_{\mathbf{d0}} + \mathbf{F} \begin{bmatrix} d \\ v \end{bmatrix} \tag{4}$$

where $\mathbf{F}$ is an $n \times 2$ matrix with feedback coefficients to the desired target joints. We use this more general structure to add stance ankle feedback for quiescent stance poses, for example.

3.3 Controller Design

Given the controller architecture described in the previous section, we need methods for choosing the parameters of each state. The resulting parameters should satisfy the requirements of the animator or control-system designer. Unfortunately, it is difficult to precisely pin down such requirements. Criteria for locomotion may include measures of style, robustness to perturbation, and energetic efficiency, all of which may push the solution in different directions and with design compromises that will be unknown in advance. Therefore, before resorting to more complex schemes, we first investigate manual interactive design of the required parameters.

The control parameters can be grouped into several categories: (a) state-transition parameters; (b) the balance feedback gains, c_d and c_v; (c) the target poses for each state; (d) the initial state for using the controller; and (e) the joint limits, torque limits, and PD-controller gains. In our work, we fix the parameters belonging to category (e) and document these in the results section. The remainder of our discussion focuses on the other parameters.

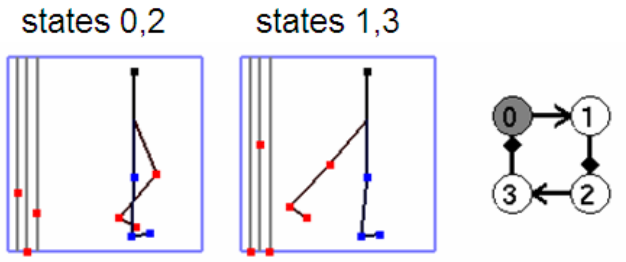

Fig. 3: Graphical interface for adjusting controller parameters. Sliders on the left control Δt, c_d, and c_v.

We begin controller design using the planar biped model, and then use the resulting parameters as a starting point for the design of corresponding 3D controllers. We use a graphical user interface (GUI) to allow a user to directly explore the parameters settings associated with each of the controllers states, as shown in Figure 3. Users can immediately observe the effect of parameter changes reflected in an ongoing simulation. Three sliders on the left of each state GUI are used to set the state duration, c_d, and c_v parameters. The target pose parameters are set by using the handle points on the stick figure. The target poses for the torso and the swing femur are interpreted with respect to the world frame. The target pose angle for the stance femur is ignored by the controller, given that the stance hip torque is treated as a free parameter whose value is determined from the torso and swing-hip torques. All the remaining joint angles define target angles with respect to their parent's coordinate frame. The interface readily exposes the key-frame like nature of many of the controller parameters.

The most important parameters for each state are the state duration Δt and the target angles for the swing hip and swing knee. Because the resulting motion style is most heavily dependent on only these three parameters per state, it becomes relatively easy for users to interactively explore their settings to yield desired motions. The ankles make a significant contribution to some styles, such as the skipping gait. The stance knee is usually almost straight. The torso is usually desired to be vertical. The balance feedback gains are set in a similar fashion across many of our controllers.

The design of a stepping-in-place gait for 2D locomotion represents a good starting point that can then be modified for the design of other motions. The target angles, as shown in Table 1, look very much like a simple pair of keyframes, one in a standing posture, and another with the swing leg in the air in a bent pose as one might expect for stepping-in-place. This leaves very few remaining parameters to set, principally the duration of the leg lift pose and the balance feedback gains for the swing hip.

Small changes ($\pm 15°$) to the desired torso pitch can be easily accomodated by treating the extra torque produced by gravity as a disturbance. During lo-

comotion, the torso may exhibit a somewhat unnatural bobbing motion. This is the result of the torso servo always reacting to the motion of the hip, rather than anticipating it. We can be addressed with feedback-error learning Yin et al. (2007).

A reasonable choice of initial state is required in order for a controller to function as designed. In practice, the balance feedback terms endow the controllers with relatively large basins of attraction, as demonstrated by their robustness to external pushes and changes in terrain, and the ability to transition directly between many of the controllers. We begin our walking controllers from a double stance state with a moderate forward velocity ($1m/s$), although we note that our basic forward walking controller can begin just as well from rest. We note that symmetric controllers can exhibit asymmetric gaits from some initial states while producing symmetric gaits from other initial states. This difficulty can be overcome by using initial states closer to the desired limit cycle.

An alternative to manual design is to use motion capture data as the basis for developing a controller. This allows a kinematic motion to be imported into a dynamic setting. Whereas kinematic motion capture data cannot be made to stumble for an unseen step or respond to a push, a style-mimicing controller allows for these effects. We refer interested readers to Yin et al. (2007) for more details on the creation of controllers that imitate locomotion styles from motion capture data.

3.4 Results

We apply the SIMBICON framework to simulated 2D and 3D bipeds having human-like proportions and mass distributions. The 2D model has 6 internal DoFs and 9 DoFs in total. The 3D model has 28 internal DoFs and 34 DoFs in total. More details of the kinematic and dynamic parameters of the models can be found in Yin et al. (2007). The combined simulation and control runs faster than real time on a standard laptop. Related source code and demonstration videos are available from the project webpage[1].

2D Biped Locomotion A set of 12 periodic gaits have been authored using the GUI described in Section 3.3. We designed these gaits to achieve a wide variety of motion styles using a small number of states. They have not been designed to be optimal gaits with respect to any given criterion. We have also authored acyclic controllers for stopping and remaining balanced on two feet, stopping and remaining balanced on one foot, and taking a single large step in the middle of a longer walking sequence. A subset of the motions are illustrated in Figure 4.

Controllers are bound to keystrokes and it is possible to interactively request transitions between the controllers. This is accomplished during state transitions, jumping from state n of controller A to state $n+1$ of controller B. We note that in

[1] http://www.cs.ubc.ca/~van/papers/Simbicon.htm

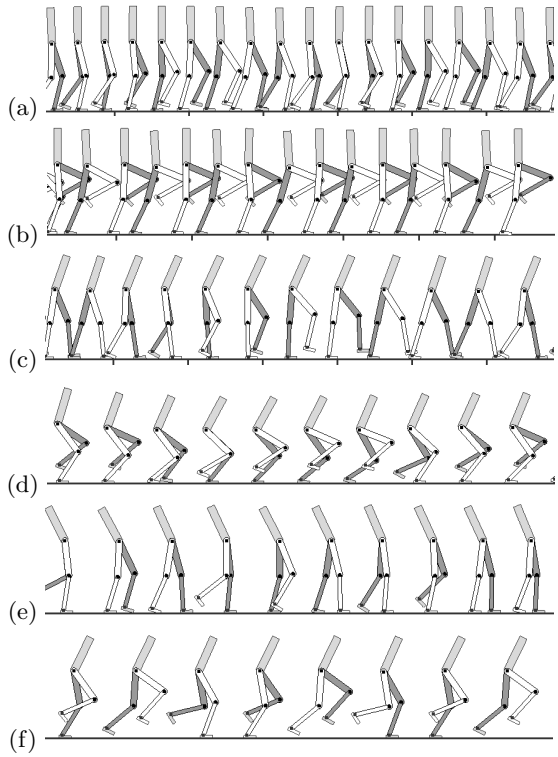

Fig. 4: A subset of the manually-designed controllers for the planar biped. (a) walk; (b) high-step walk; (c) bent walk; (d) crouch walk; (e) backwards walk, right-to-left; (f) fast run.

the absence of specially-designed transition motions, not all controller transitions are feasible.

The robustness of the walk controller to variations in terrain is shown in Figure 5. The terrain includes unanticipated downward steps of $20cm$ and slopes of ± 6 degrees. The robustness of the gaits with respect to unanticipated pushes was tested by applying 10 pushes at $5s$ intervals, which serves to sample various phases of the gait while also allowing the biped time to fully recover between pushes. Any single stumble from which the biped cannot recover is deemed a failure. The walking controller can withstand $0.1s$ duration pushes of up to $600N$ forwards and $500N$ backwards at all 10 sampled points in the locomotion cycle. Other gaits are more sensitive to disturbances. For example, the skipping gait can withstand $0.1s$ duration pushes of up to $40N$ forwards and $50N$ backwards. Larger pushes, as measured by their induced change in momentum, $F\Delta t$, can be sustained by increasing Δt and decreasing F. Specific portions of the walk cycle can also withstand larger pushes.

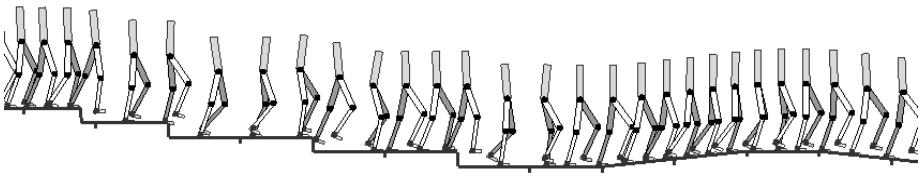

Fig. 5: A planar biped responds to unanticipated changes in terrain.

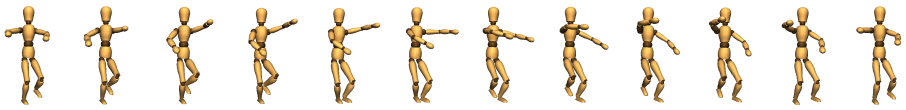

Fig. 6: Knee-bent sideways walking with hands circling in opposite phase for the "cloud hands" Tai Chi movement. Every third frame.

3D Biped Locomotion Controllers for 3D walking require twice the number of parameters as for 2D control because of the need for lateral (coronal-plane) control. Manually-designed controllers have been developed for two-foot hopping, three styles of forward walking, walking up a slope of 20 degrees, two styles of forward running, a sideways-walk Tai Chi movement, and skipping. Figure 7 illustrates walking up a slope of 20 degrees, and Figure 6 illustrates the Tai Chi "cloud hands". Direct transitions are possible between most of the walking and running gaits. Many of our 2D control strategies work directly when applied to the 3D model and applying a common set of feedback gains for the lateral hip movement that is responsible for lateral balance. There is some variation in the style of the 2D and 3D motions, likely because our 2D and 3D models have different masses and proportions. For the few cases where the difference in models is problematic, minor adjustments to target angles and gains are sufficient to achieve a functional 3D motion.

A robust balance controller also means that the locomotion can deal with unexpected environmental disturbances automatically. For the 3D walk controller given in Table 1, the largest recoverable pushes as measured in eight evenly-sampled directions are $(0, 340)$, $(230, 230)$, $(330, 0)$, $(220, -220)$, $(0, -270)$, $(-190, -190)$, $(-240, 0)$, $(-190, 190)$, where each pair defines the (lateral,sagittal) push magnitudes in Newtons. The pushes are applied at chest height at a phase angle of $\phi = 0.1$ and have a duration of 0.4s. In Figure 8 we show a walking controller constructed from a motion capture example recovers from a large push.

4 GENBICON: GENeralized BIped Walking CONtrol

The GENBICON control framework improves SIMBICON in terms of its generalization ability across gait parameters, motion styles, character proportions, and a variety of walking-related skills, including picking up objects placed at

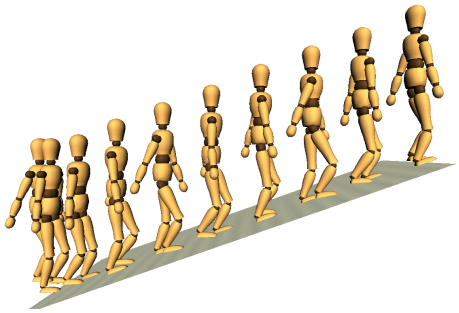

Fig. 7: Climbing a slope of 20 degrees. 1s lapse.

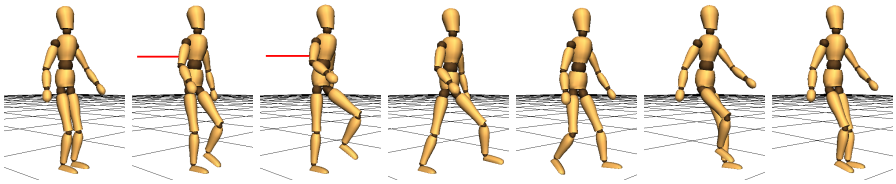

Fig. 8: A walk controller reconstructed from motion capture data responds to a 350N, 0.2s diagonal push to the torso.

any height, lifting and walking with heavy crates, pushing and pulling crates, stepping over obstacles, ducking under obstacles, and climbing steps. The control requires no character-specific or motion-specific tuning, is robust to disturbances, and is simple to compute. Thus naive users can interactively author the character proportions, gait parameters, and motion styles. The key components of GENBICON are an inverted pendulum model to provide the foot placement strategy, and adjustments for gravity and velocity errors using Jacobian-transpose control. Similar to SIMBICON, the base control of GENBICON also uses joint PD servos for tracking desired trajectories.

4.1 Base Control

The GENBICON base control is a *motion generator* that produces the various desired trajectories that help create desired motion styles. The trajectories directly model desired joint angles, either relative to their parent joints (elbows, shoulders, stance knee, and toes) or relative to the character coordinate frame (waist, back, neck, and ankles). The character frame is defined by axes aligned with the vertical axis of the world and the facing direction of the character. The desired joint angles are provided as input to PD-controllers that produce tracking torques. The trajectories are modeled as a function of the phase of a step, $\phi \in [0, 1)$ using Catmull-Rom splines. A walk cycle then consists of two

state	Δt	c_d	c_v	*tor*	*swh*	*swk*	*swa*	*stk*	*sta*
0,2	0.3	0.5	0.2	0	0.5	-1.1	0.6	-0.05	0
lat		0.5	0.2	0	0	0	0	0	0
1,3	fc	0.5	0.2	0	-0.1	-0.05	0.15	-0.1	0
lat		0.5	0.2	0	0	0	0	0	0

Table 1: Locomotion parameters for the periodic, left-right symmetric 3D gaits. The columns from left to right represent the state numbers, state dwell duration, position and velocity balance feedback coefficients, and the torso, swing-hip, swing-knee, swing-ankle, stance-knee, and stance-ankle target angles. All angles are expressed in radians.

alternating steps, with the second step being left-right symmetric with respect to the first. The number of spline segments is arbitrary, although we typically use three segments. The final simulated motions do not necessarily tightly track the target trajectories and contain significant details that are not present in the desired trajectories.

The motion generator trajectories can be authored using keyframes, as illustrated in Figure 12. In general, a controller will work robustly with all angle trajectories being set to zero, which implies having the arms pointing straight down, a straight stance leg, feet and toes remaining parallel to the ground, and a fully upright body. This nominal 'zero gait' provides a logical starting point for end-user authoring of gaits. When not zero, the target trajectory for the stance knee is assigned a constant value in order to achieve a bent-leg crouched gait, for example. The stance ankle trajectory can be driven to achieve tip-toe walks or other desired toe-off patterns. As will be described shortly, the stance hip and the swing leg are controlled separately and thus these will still actively produce joint motion in the case of all zero joint angle trajectories. Only the $y(\phi)$ curve, which specifies the swing foot height, need be non-zero (Section 4.2).

To allow for simple and coordinated control over bending of the body, the target angles for the lower back, upper back, and head are modeled as fixed ratios of a single user controllable bend angle in the sagittal plane. The constraint can be dropped in order to allow for more artist control and it is not enforced in our style editing interface (Section 4.4).

Given a desired time period for a single step, T, the current phase of an ongoing motion is computed as $\phi = t/T$. The motion generator keeps track of the current stance leg using a two-valued state variable, $s \in \{left, right\}$. A new step is assumed to begin at foot-strike or when $t \geqslant T$, whichever occurs first. Following the related idea in SIMBICON, the motion generator does not provide target trajectories for the stance hip. Instead, its torque is computed to achieve the desired net torque on the pelvis, which is the root link of the character.

4.2 Inverted Pendulum Foot Placement

GENBICON adopts the same swing foot replacement balance strategy as SIMBICON. However, rather than using a hand-tuned linear feedback law, GENBICON first computes the desired stepping point, (x_d, z_d), using an Inverted Pendulum Model (IPM) in order to achieve a general solution that works across a wide range of body types. In particular, it builds on an inverted pendulum model that assumes constant leg length Pratt and Tedrake (2006) as shown in Figure 9. The analysis equates the sum of the potential and kinetic energy of the IPM at the current state, described by its current velocity v, its height, h, and distance, d, from the future point of support, with that at the balanced rest state, i.e.,

$$\frac{1}{2}mv^2 + mgh = \frac{1}{2}mv'^2 + mgh', \tag{5}$$

where $v' = 0$ and $h' = L = \sqrt{h^2 + d^2}$. Solving this relation for d gives

$$d = v\sqrt{h/g + v^2/(4g^2)}. \tag{6}$$

The above model computes the desired value of d in order to reach zero velocity at the next step. Taking a shorter step will achieve a positive velocity while taking a larger step will achieve a negative velocity, i.e., walking backwards. Accordingly, we compute

$$d' = d - \alpha V_d, \tag{7}$$

where V_d is the magnitude of the desired velocity and α is a constant. We use $\alpha = 0.05$ for all the results demonstrated in this paper. Because the velocity is also controlled using the velocity tuning component, the control is not particularly sensitive to the particular value of α. The IPM is applied in the sagittal plane to compute $x_d = d$ and is repeated in the coronal plane to obtain z_d in an analogous fashion. We use the true center of mass position and velocity when applying the pendulum model to the biped. The target values x_d and z_d are updated at every time step, although if a particular foot placement in the world is desired, the character can ignore the IPM prediction for one or two consecutive steps. Depending on the speed of the character, the IPM prediction may be out of reach, in which case the characters uses a maximum step length of $d = 0.6L$. This situation leads to the character possibly having to take multiple steps to recover.

Inverse kinematics is used to compute target joint angles for the swing leg from the desired foot placement point. More specifically, we first synthesize a desired trajectory for the swing ankle relative to the ground and in the character coordinate frame. The height with respect to the ground, y, is modeled as a function of phase using a three-segment spline curve in the motion generator, i.e., $y(\phi)$. This curve defines the difference between a ground-hugging step and a high leg-lift step. The x and z trajectories follow linear trajectories between their locations at the start of the step (x_0, z_0) and their desired location at the end of the step, (x_d, z_d), i.e., $x(\phi) = (1-\phi)x_0 + \phi x_d$ and $z(\phi) = (1-\phi)z_0 + \phi z_d$. We then use inverse kinematics to compute target joint angles for the swing

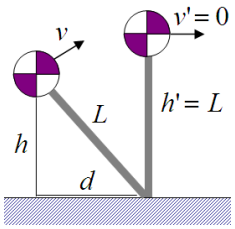

Fig. 9: Inverted pendulum model.

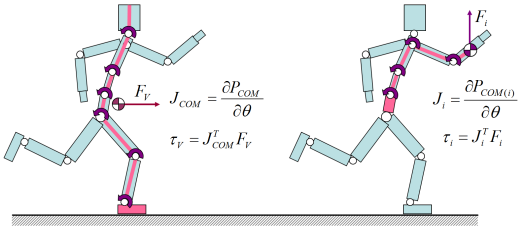

Fig. 10: Jacobians used for velocity tuning (left) and gravity compensation (right).

hip and knee, which are then tracked using PD controllers and augmented by gravity compensation torques. The inverse kinematics problem has a remaining degree of freedom which allows for knock-kneed, normal, or bow-legged walking variations. We expose this as a twist angle parameter to the animator which is held constant within any given style of motion.

4.3 Jacobian Transpose Control

The Jacobian-transpose (JT) application of virtual forces is fundamental to force-based control in robotics. The use of JT methods to generalized control variables, such as the center of mass, was proposed in Sunada et al. (1994) and further developed in Virtual Model Control Pratt et al. (2001). We use JT methods to supply joint torques for gravity compensation, as well as to apply a virtual force to the center of mass which helps regulate its velocity.

More specifically, to compute the joint torques τ needed to apply a virtual force F to a cartesian point p, we use the transpose of the kinematic Jacobian J_p as follows:

$$\tau = J_p^T F \tag{8}$$

where J_p is the Jacobian of the chain from point p to a chosen root, such as the stance foot or pelvis. JT torques are usually added to the PD tracking torques at the joint level.

Velocity Tuning While foot placement ensures robustness for the gait, it can only be enacted once per step. Using foot placement alone ignores the ability to use the stance foot (or feet) to help maintain balance by manipulating the ground reaction forces (GRFs), or equivalently, manipulating the location of the center of pressure (COP), also known as the zero-moment point (ZMP). Simply put, shifting the COP towards the toes helps in slowing the forward progression of the body, while shifting it back helps it accelerate. We use a simple virtual velocity-tuning force, akin to that proposed in Pratt et al. (2001), to provide fine-scale control over the center of mass velocity. It has the effect of altering the GRFs and the COP and so we place this control strategy in the same general

category as other techniques that manipulate the GRFs, COP, or ZMP in order to enact balance feedback.

The specific details of computing and applying the virtual force, F_V, on the center of mass are as follows. We first compute the center of mass velocity of the biped, V. In the sagittal plane, we compute a desired virtual force according to $F_V = k_V(V_d - V)$, where V_d is the desired sagittal velocity. In the coronal plane, an analogous virtual force is computed using a PD-controller that tracks a desired lateral COM position. This desired position varies linearly over time from its original position at footstrike to a step width W at the time of the next predicted footstrike. By default, using $W = 0$ results in a catwalk style of motion. A non-zero value of W results in the stance leg pushing the body to the side, which then causes the inverted pendulum model to compensate accordingly with a lateral swing-foot placement.

The combined sagittal-and-lateral virtual force is achieved using JT control as $\tau_V = J_V^T F_V$. Here, J_V is the Jacobian of the center of mass for a chain of links rooted at the stance ankle. In practice, we compute J_V using only the joints between the stance foot and the head, as shown in Figure 10 (left), which incorporates all the key joints that support most of the weight. The virtual force helps fine tune both sagittal and lateral balance. Because of the finite support of the feet, applying τ_V can in some situations cause undesired foot rotation, e.g., the toe rotating off the ground in an attempt to accelerate forwards. To mitigate this problem we zero the ankle component of the torque if we detect that the stance foot is not well planted, as measured by the toe leaving the ground or the foot exceeding a rotational velocity threshold. There is potential to have 'chatter' occur, but in practice we have not found this to be the case. During double stance each stance leg is treated independently from its foot to the torso and the results are simply summed.

Gravity Compensation The use of computed gravity compensation (GC) torques allows for the use of significantly lower gains in the PD controllers for the limbs and the body. Although highly controllable motion could also be achieved with the use of a full inverse dynamics solution, we show that this is unnecessary. The addition of gravity compensation allows simple local control methods, such as low-gain PD-control, to succeed.

Gravity compensation is applied using JT once again. We wish to apply a virtual force $F_i = -m_i g$ at the center of mass of every link i, where the negative sign implies an upwards force. The torques required for this are computed as $\tau_i = J_i^T F_i$. Here, J_i is the Jacobian of the link center of mass location with respect to the chain of joints between the root link and link i. GC torques are thus computed for all joints that lie in between each link i and the root link, i.e., the pelvis. Any given joint j thus sees the sum of the GC torques required by all links that are distal to it. The compensation is applied to all links, with the exception of those in the stance leg.

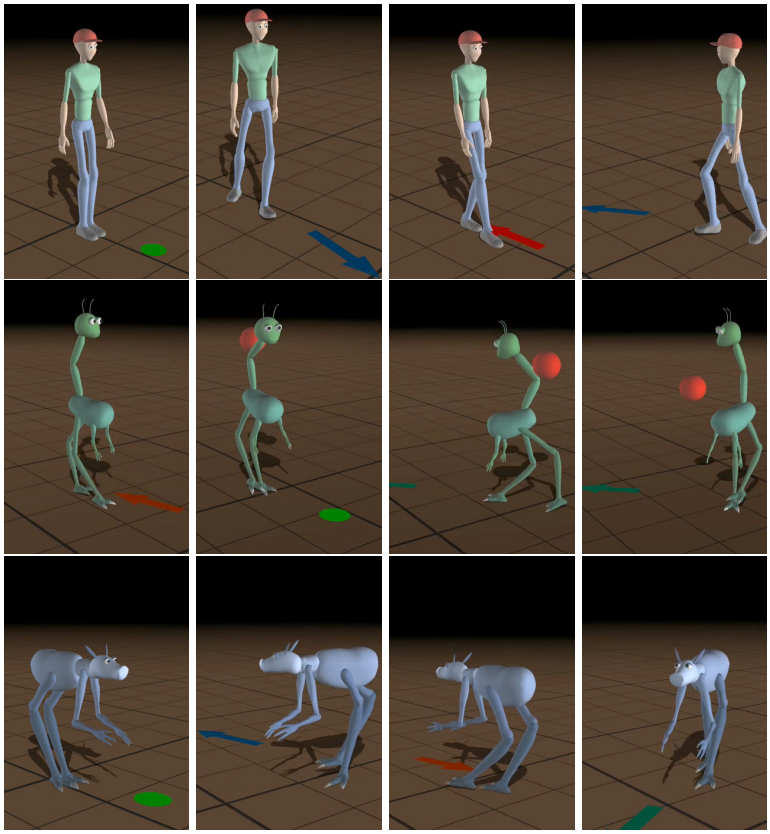

Fig. 11: Direction Following

4.4 Results

We apply the GENBICON framework to a variety of 3D characters, including one humanoid, one robot, and two beasts. More details of the kinematic and dynamic parameters of these models can be found in Coros et al. (2010). Related source code and demonstration videos are available from the project webpage[2].

Generalization across gait parameters We first show that the walking control generalizes across forwards-and-backwards walking, walking speeds, stepping frequencies, and turning towards a desired direction. We also connect these with stopping and starting behaviors. We demonstrate these basic behaviors across different characters and for different styles of motion. No character-specific or style-specific parameter tuning is required. Figure 11 shows frames from the animations. A commanded forward walk is indicated by a blue arrow, a backwards

[2] http://www.cs.ubc.ca/~van/papers/2010-TOG-gbwc

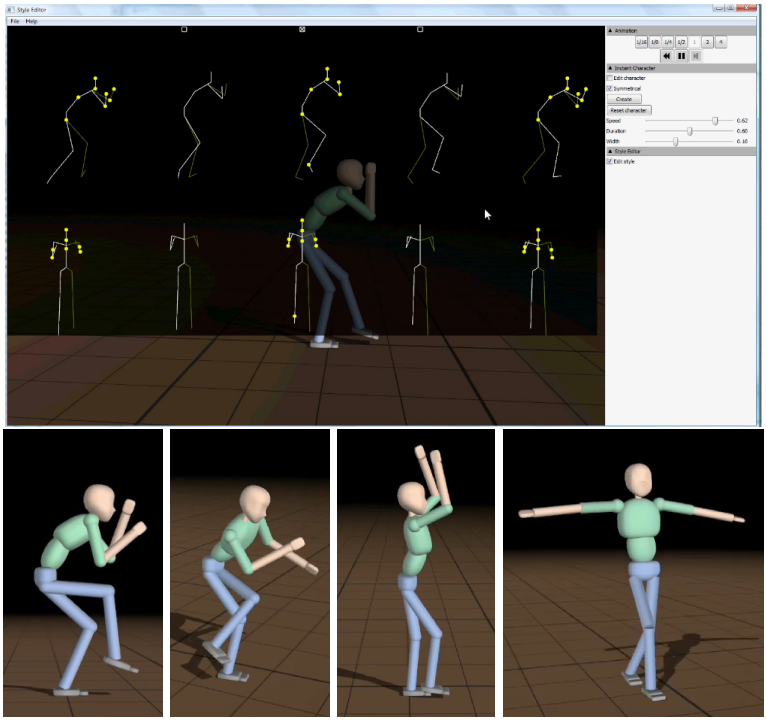

Fig. 12: Motion style editing and example motion styles.

walk by a red arrow, and a green dot indicates a command to stop walking. Our characters walk at speeds ranging from -0.6 to $1.7m/s$.

Generalization across style We develop an interface that allows animators to easily author unique motion styles. Figure 12 shows the interface and example motion styles that were created in 5-10 minutes by novice users, including children as young as 7. Different styles can be authored interactively by modifying the lean of the upper body in the sagittal plane, the target angle of the stance knee, the plane of rotation for the swing-leg IK, and the arm motions. Our walking control strategy instantly provides robust dynamically-simulated gaits. The interface is constrained to symmetric trajectories for the sake of simplicity.

Generalization across characters The control strategy generalizes well across a variety of character dimensions and proportions, as shown in Figure 11. We further develop an interface that allows a user to interactively edit the dimensions and proportions of a character and immediately see the resulting motions. The interface and examples of resulting characters are shown in Figure 13. Characters can have asymmetric arms and legs, which will introduce asymmetries in the motion even with symmetric controls from the motion generator. Asymme-

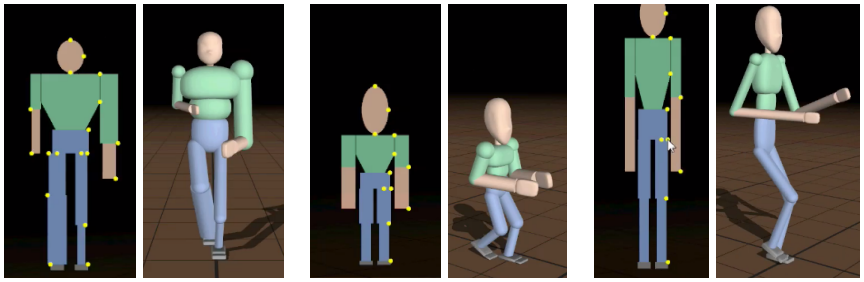

Fig. 13: Interactive editing of character proportions and the resulting walking gait.

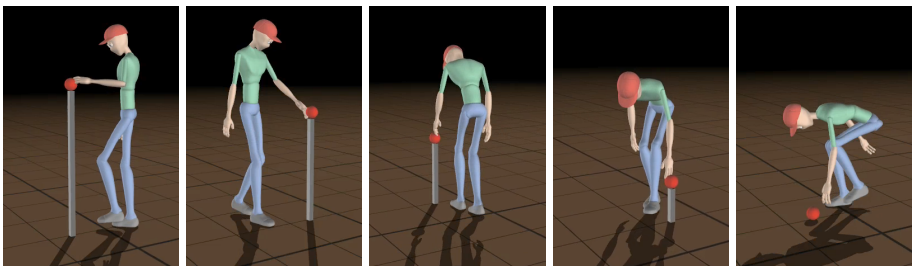

Fig. 14: Reaching for objects

try in the placement of the knee joint, such as for our robot character, is largely accommodated by the inverse kinematics used for tracking the desired swing ankle trajectories.

Generalization across tasks Walking controllers should also generalize across tasks and be able to interact with the environment in various ways. We demonstrate this on several different tasks.

Reaching: Figure 14 shows the ability to reach and grasp an object placed at an arbitrary location and at an arbitrary height. A desired heading direction is computed once per step for the character based on the location of the object. Most reaching motions can be carried out while walking at a regular pace. Objects near the ground require the character to slow down, which is implemented as a linear function of the remaining distance, d, to the object. Low reaches also require bending down, as achieved using a combination of forward upper body lean and bending of the stance knee. The reaching motion is triggered when $d < \epsilon$, at which point the arm treats the final object position as a target. Inverse kinematics provides required joint angles, which are then tracked using low-gain PD-controllers and augmented by the gravity-compensation torques.

Pulling and pushing a crate: The walking control can be used to pull and push objects, as shown in Figure 15. The crate weighs 80 kg and has a coefficient of

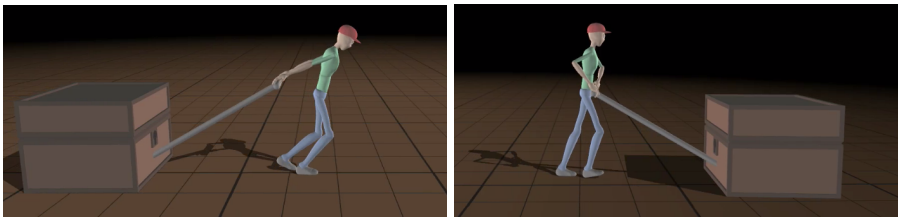

Fig. 15: Pulling and pushing to the right.

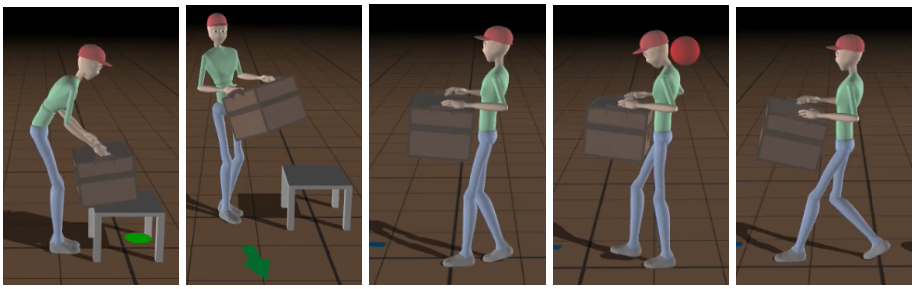

Fig. 16: Lifting and moving heavy crates.

friction of 0.2 with the ground. The high level behavior has the character walk towards the crate, slow down when near the crate handles, and reach for the handles when within reach. Springs and dampers are used to connect the hands to the handles. The inverted pendulum model is adapted to account for the weight to be pushed or pulled by using $\alpha = 0.2$ instead of $\alpha = 0.05$.

Lifting and moving a crate: The simulated characters can lift and move heavy crates (5–25 kg) using the generalized walking control, as shown in Figure 16. The high level behavior used to approach the crate is similar to that used in the pulling and pushing skill. Once the hands are attached to the box with stiff springs and dampers, the crate mass is incorporated into the computations for the overall COM position and velocity. It is also incorporated into the gravity compensation by adding a compensation force corresponding to half the weight of the box to each hand.

Navigation over and under obstacles: A high level behavior that steps over obstacles and ducks under other obstacles is also easy to create. Of particular interest is that the stepping-over behavior and the ducking behavior are controlled independently as they do not interfere with each other, as shown in Figure 17. The step-over and duck-under obstacles are paired together with varying offsets in order to demonstrate that the factored control works well for all combinations of these two skills. The control for ducking under obstacles consists of bending the body as a linear function of the obstacle distance, yielding a smooth ducking motion.

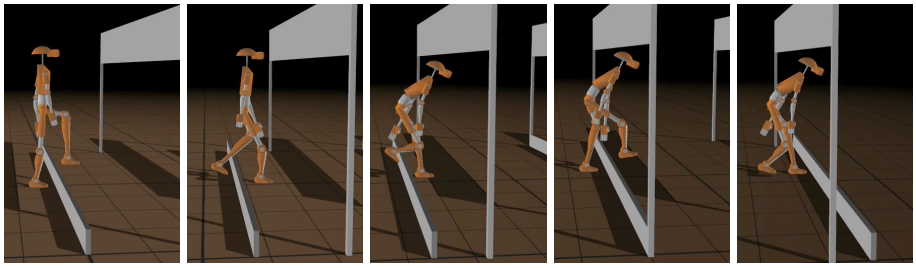

Fig. 17: Stepping over a sequence of obstacles combined with ducking under obstacles, with varying offsets.

The high level control for stepping over an obstacle involves slowing down upon approaching the obstacle. It is then important to arrange for a footstep that is planted near the obstacle itself before stepping over it. This footstep is done by adapting the step length once within a threshold distance of the obstacle. Because the location of the swing foot is determined by the inverted pendulum model, we implicitly adjust the step length by manipulating the desired velocity. Once in position to step over the obstacle, a trajectory is planned that allows the swing foot to clear the obstacle in the upcoming step. This trajectory guides the swing ankle and is constructed using four Catmull-Rom spline segments. The swing foot follows this trajectory until the heel is past the obstacle, after which control of the forward component once again reverts to the inverted pendulum in order to produce a final foot placement that is suitable from the point of view of balance. The motion of the trailing foot over the obstacle is controlled in an analogous fashion. We also note that the same high level control can also be used for climbing a step, except for the way in which the vertical trajectory of the swing ankle is computed.

5 Future Directions

In conclusion, biped control has seen significant progress in terms of quality, agility, and robustness. Tracking controllers can achieve high quality of motion with minimal human design effort, while controllers with explicit balance strategies are more robust and easier to stylize. Both methods have no direct knowledge of the equations of motion, which are usually required by model-based methods such as optimal control.

There remain many open problems for further investigation in the near future. We are particularly interested in further pushing forward the robustness and generalization ability of biped controllers through deep reinforcement learningPeng et al. (2016, 2017) . Such controllers should require minimal *a priori* knowledge to build and solve higher-level tasks such as navigation and manipulation. We conjecture that better control representations can still be devised, and more powerful learning frameworks need to be explored.

Acknowledgements

We sincerely thank all our collaborators for their contributions to the work described in this Chapter, especially Kevin Loken and Philippe Beaudoin. This work was funded in part by NSERC Discovery Grant RGPIN-2015-04843.

Bibliography

Mazen Al Borno, Martin de Lasa, and Aaron Hertzmann. Trajectory optimization for full-body movements with complex contacts. *IEEE Transactions on Visualization and Computer Graphics*, 19(8):1405–1414, 2013.

Stelian Coros, Philippe Beaudoin, and Michiel van de Panne. Generalized biped walking control. *ACM Trans. Graph.*, 29(4):Article 130, 2010. doi:10.1145/1778765.1781156.

DART. Atlas simbicon, `https://dartsim.github.io/gallery.html`.

Martin de Lasa, Igor Mordatch, and Aaron Hertzmann. Feature-based locomotion controllers. *ACM Trans. Graph.*, 29(4):131:1–131:10, 2010. ISSN 0730-0301.

Thomas Geijtenbeek and Nicolas Pronost. Interactive character animation using simulated physics: A state-of-the-art review. In *Computer Graphics Forum*, volume 31, pages 2492–2515. Wiley Online Library, 2012.

Stevie Giovanni and KangKang Yin. Locotest: Deploying and evaluating physics-based locomotion on multiple simulation platforms. *Lecture Notes in Computer Science*, 7060:227–241, 2011.

Perttu Hämäläinen, Joose Rajamäki, and C Karen Liu. Online control of simulated humanoids using particle belief propagation. *ACM Transactions on Graphics (TOG)*, 34(4):81, 2015.

Jessica K. Hodgins, Wayne L. Wooten, David C. Brogan, and James F. O'Brien. Animating human athletics. In *SIGGRAPH'95*, pages 71–78, 1995.

Yoonsang Lee, Sungeun Kim, and Jehee Lee. Data-driven biped control. *ACM Trans. Graph.*, 29(4):129:1–129:8, 2010.

Libin Liu, KangKang Yin, Michiel van de Panne, Tianjia Shao, and Weiwei Xu. Sampling-based contact-rich motion control. *ACM Trans. Graph.*, 29(4): Article 128, 2010. doi:10.1145/1778765.1778865.

Libin Liu, KangKang Yin, Michiel van de Panne, and Baining Guo. Terrain runner: control, parameterization, composition, and planning for highly dynamic motions. *ACM Trans. Graph.*, 31(6):Article 154, 2012. doi:10.1145/2366145.2366173.

Libin Liu, KangKang Yin, Bin Wang, and Baining Guo. Simulation and control of skeleton-driven soft body characters. *ACM Trans. Graph.*, 32(6):Article 215, 2013. doi:10.1145/2508363.2508427.

Libin Liu, KangKang Yin, and Baining Guo. Improving sampling-based motion control. *Computer Graphics Forum*, 34(2):415–423, 2015. doi:10.1111/cgf.12571.

Libin Liu, Michiel van de Panne, and KangKang Yin. Guided learning of control graphs for physics-based characters. *ACM Trans. Graph.*, 35(3):Article 29, 2016. doi:10.1145/2893476.

Adriano Macchietto, Victor Zordan, and Christian R. Shelton. Momentum control for balance. *ACM Trans. Graph.*, 28(3):Article 80, 2009.

Uldarico Muico, Yongjoon Lee, Jovan Popović, and Zoran Popović. Contact-aware nonlinear control of dynamic characters. *ACM Trans. Graph.*, 28(3): Article 81, 2009.

Xue Bin Peng, Glen Berseth, and Michiel van de Panne. Terrain-adaptive locomotion skills using deep reinforcement learning. *ACM Transactions on Graphics*, 35(4):Article 81, 2016.

Xue Bin Peng, Glen Berseth, KangKang Yin, and Michiel van de Panne. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Transactions on Graphics*, 36(4):Article 41, 2017.

J. Pratt, C.M. Chew, A. Torres, P. Dilworth, and G. Pratt. Virtual model control: An intuitive approach for bipedal locomotion. *Int'l J. Robotics Research*, 20(2):129, 2001.

Jerry E. Pratt and Russ Tedrake. Velocity based stability margins for fast bipedal walking. In *Fast Motions in Biomechanics and Robots*, 2006.

Craig Sunada, Dalila Argaez, Steven Dubowsky, and C. Mavroidis. A coordinated jacobian transpose control for mobile multi-limbed robotic systems. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 1910–1915, 1994.

Andrew Witkin and Michael Kass. Spacetime constraints. In *SIGGRAPH'88*, pages 159–168, 1988.

KangKang Yin, Kevin Loken, and Michiel van de Panne. SIMBICON: Simple biped locomotion control. *ACM Trans. Graph.*, 26(3):Article 105, 2007. doi:10.1145/1276377.1276509.

Index