Synthesis of Constrained Walking Skills

Stelian Coros Philippe Beaudoin KangKangYin Michiel van de Panne*

University of British Columbia

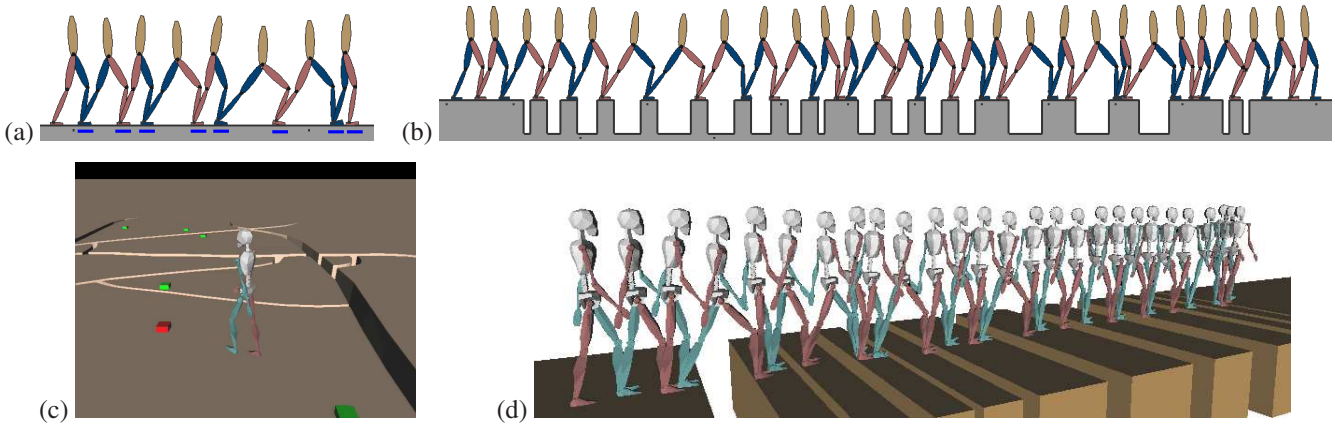

Figure 1: *Constrained walking skills. (a) Offline synthesis is used to generate physically-simulated motions for example problems. The example motions are used to develop a dynamics model that can make accurate step-to-step predictions. (b) This model can then be used by an online planner to navigate across constrained terrain. (c) A 3D physics-based character simulation plans steps to avoid stepping in crevasses. (d) A challenging terrain being navigated by the 3D model.*

Abstract

Simulated characters in simulated worlds require simulated skills. We develop control strategies that enable physically-simulated characters to dynamically navigate environments with significant stepping constraints, such as sequences of gaps. We present a synthesis-analysis-synthesis framework for this type of problem. First, an offline optimization method is applied in order to compute example control solutions for randomly-generated example problems from the given task domain. Second, the example motions and their underlying control patterns are analyzed to build a low-dimensional step-to-step model of the dynamics. Third, this model is exploited by a planner to solve new instances of the task at interactive rates. We demonstrate real-time navigation across constrained terrain for physics-based simulations of 2D and 3D characters. Because the framework sythesizes its own example data, it can be applied to bipedal characters for which no motion data is available.

1 Introduction

The creation of flexible models of motion, such as the skills needed by a character to move through a constrained environment, remains a challenging problem in computer animation. A common approach is to resequence example kinematic motion data, as done in motion graphs and their variants. Another idea is to model the processes that gives rise to motions, which is the approach adopted in physics-based animation. This type of model has the potential to be more flexible in that it can allow for direct interaction with the environment, as mediated by physics and forces. In character animation, physical models are commonly used to generate rag-doll effects, but creating skilled motions using simulations remains problematic because of the required control. The simulation of skilled walking needs to address issues of balance and control in high-dimensional action spaces. Skilled walking further requires planning in order to cope with constraints in the environment, such as gaps that need to be stepped over. Because walking is just one of many skills that we may wish simulated characters to have, it is important to have methodologies that automate the development of skills.

In this paper we propose automated techniques for developing constrained walking skills for bipedal characters. As shown in Figure 1(b,c,d), the problem is defined as one of walking across a level terrain with gaps, such that the gaps partially or fully constrain where the character can step. A *stepping stone problem* represents a fully-constrained scenario where the sequence of desired foot-step locations has been fixed in advance. The problem is particularly interesting in that there is no obvious parameterization for this task. Modeling the control required for a given step as a function of the length of the next desired step is insufficient because the action also needs to take the starting state into account. For example, when taking a 90cm step, the character needs to take different actions for an initial forward velocity $v = 0.3\ m/s$ as compared to $v = 1\ m/s$. The important role of the character state makes planning and control strongly coupled problems for this task. A planner for walking across a terrain with gaps needs to know what the controller is capable of in any given situation, e.g., in a particular state is it possible to take a large step over an upcoming gap?

*e-mail: {scoros|beaudoin|kkyin|van@cs.ubc.ca}

Our method is characterized by its *synthesis-analysis-synthesis* approach. First, *offline synthesis* (§3) is used to generate physically feasible control solutions to a set of sample problems from the given task domain. The generated solutions consist of the motions, derived from a forward-dynamics simulation, and the control inputs that created the motions. Figure 1(a) shows an example problem sequence of target stepping locations and the simulated motion resulting from the offline synthesis. Second, *offline analysis* (§4) is used to develop a low-dimensional step-to-step dynamical model that is based on the family of motions computed in the first step. Third, this dynamical model is exploited during *online synthesis* (§5) to plan and control new task instances from the same task domain.

The contributions of the paper can be summarized as follows. First, a solution is developed for the difficult problem of controlling physically-simulated characters in real time to walk in environments having significant stepping constraints. Second, a synthesis-analysis-synthesis approach is introduced as an effective method for creating a more general skill from a simple walk cycle controller. Third, real-time motion planning is demonstrated for physically-simulated characters with continuous action spaces.

2 Related Work

A variety of previous work touches on aspects of our problem and serve as inspiration for the method proposed in this paper. Because of the ubiquitous nature of walking, many data-driven kinematic models of walking have been developed to generate flexibly-parameterized walking gaits [Kwon and Shin 2005; Mukai and Kuriyama 2005; Wang et al. 2005]. Several methods demonstrate forms of constrained walking based on resequencing or interpolation of kinematic motion data [Choi et al. 2003; Mukai and Kuriyama 2005; Reitsma and Pollard 2007; Safonova and Hodgins 2007].

Physics-based models have the advantage of using an explicit model of the physics and thus have the potential to be more general. A variety of feedback-based control strategies have been developed for walking and running characters [Raibert and Hodgins 1991; Laszlo et al. 1996; Hodgins and Pollard 1997; Sharon and van de Panne 2005; Sok et al. 2007; Yin et al. 2007; da Silva et al. 2008b; da Silva et al. 2008a]. However, it is not obvious how they can be adapted to handle constrained locomotion scenarios, especially since foot placement is often a key element for regulating balance and speed.

The use of trajectory-based optimization techniques to compute optimized motion sequences has a long history in animation. One of the challenges has been to develop techniques that can cope well with the high number of degrees of freedom of human motion. This can be done with simplified physical models [van de Panne 1997; Popović and Witkin 1999], data-driven dimensionality reduction of the state [Safonova et al. 2004], or leveraging existing motion data [Liu et al. 2005]. Our offline optimization step is similar to what can be accomplished with trajectory optimization methods in that we desire to modify a default walking gait subject to new stepping constraints. Our proposed method uses the results of the offline synthesis as a point of departure. By observing that there is significant structure in the example solutions, results can be cheaply estimated using regression instead of using an expensive optimization[1]. Unlike the offline optimization problem, solving partly-constrained walking problems requires exploring discrete alternatives in footstep placement, such as whether or not to take an extra step before crossing a gap. As a result, this type of optimization falls outside the scope of gradient-based optimization techniques, even if the stepping location and stepping time are also treated as free parameters. Two last points of distinction of the proposed method are that it can provide real-time control and that it works with standard black-box forward dynamics simulators for both the offline optimization and the online simulation.

Planning physics-based motions across terrain with constraints is a problem that has been studied in the context of hopping robots [Hodgins and Raibert 1991; Zeglin and Brown 1998], where it can be simplified in ways that are specific to the structure of this kind of robot. A discrete search strategy is applied to a dynamically simulated hopping lamp character in [Huang and van de Panne 1996]. Recent work has shown very promising results for terrain-specific policies for planar compass gaits, including terrains with sequences of gaps [Byl and Tedrake 2008]. Our work is also motivated by footstep planning for humanoid robotics. Kinematic models of robot capabilities have been used very effectively to compute optimized footfall sequences that avoid obstacles [Kuffner et al. 2003; Chestnutt and Kuffner 2004]. Kinematic capability models assume that a fixed range of stepping lengths is always achievable. They do not model the fact that a successful large step may be impossible for initial states that are moving too slowly or that a small step may be impossible or undesirable when moving too quickly.

More recent work has demonstrated footstep planning for the Honda ASIMO robot that does consider the state-dependent nature of actions [Chestnutt et al. 2005]. The planning strategy treats the robot's pre-existing balance control and stepping strategies as a black box that responds to body displacement commands. The key insight is that for the given robot and its controller, the current state of the robot can be accurately predicted by knowing only the last two commanded actions. The planner considers a fixed set of seven possible discrete actions for each step. A sequence of all $7 \times 7 \times 7 = 343$ command sequences is then used to create a discrete model of the space of all possible motions using a directed graph. Motion planning then uses A* search.

Also closely related is the work of [Hofmann 2006], which demonstrates a motion planning algorithm for a 3D humanoid model for a task involving a prescribed irregular foot placement. The motion is controlled using a user-developed *qualitative state plan* which is a finite state machine abstraction with specified constraints and goals associated with each state. The center of mass position and velocity are used as the state of an abstracted virtual model, which allows for some flexibility when executing the plan.

3 Offline Synthesis

Our approach begins by computing dynamically-simulated solutions to example stepping-stone problems, as shown in Figure 2. Given a sequence of target foot locations, the goal is to find a control sequence that results in the character stepping precisely at the desired locations. Both the synthesis and the subsequent analysis use individual steps as the basic motion primitive. Steps are defined based on foot-strike events, i.e., one step ends and the next one begins when the swing foot strikes the ground. The output of the process is a sequence of example steps where, for each step i, we know the starting state s_i, the applied action A_i, the resulting state s_i', and the resulting step length, l_i. The sequence of actions are the unknowns for the example problem sequences. The step length is measured as the heel-to-heel distance.

Finding solutions to a given stepping-stone problem is cast as an optimization problem. Given a base controller defined by parameters A_{base} that produces a steady-state walking gait, the goal of the optimization is to find modified parameters for each step, A_i,

[1] We note, however, that the regression model we use is in fact not a direct substitute for the offline optimization step because we treat the action as an input rather than an output. This issue is discussed further in Section 7.

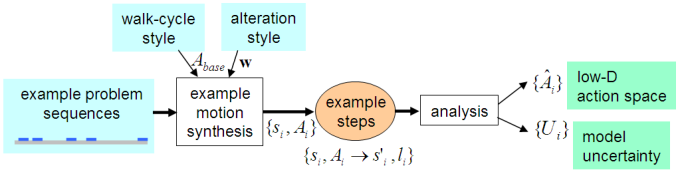

Figure 2: *Offline Synthesis and Analysis.*

such that the target stepping sequence is achieved. Implementing these ideas requires defining the example problems, the parameterized base controller, the optimization function, and the optimization method. We now discuss each of these aspects in turn, focussing on their application to planar models. The specific extensions to 3D control are deferred to §3.3.

Example stepping-stone problems are generated using a uniform step-length distribution $l \in [0.1m, 0.9m]$ for the humanoid biped and $l \in [0.1m, 1.0m]$ for the big-bird character to be introduced later. We use multiple 100-step sequences instead of a single long sequence because it is possible for the optimization to fail to find good solutions for difficult or perhaps impossible sequences of steps. While such failure is rare, it can compromise solutions for the remainder of the steps in a sequence. The character geometry, control representation, joint limits, and torque limits will all impose constraints on the types of step sequences that are feasible, as will the optimization technique itself.

3.1 Actions

The applied actions, A_i, for our method are defined in terms of the four-state finite-state machine (FSM) control structure described in [Yin et al. 2007], which we build on because of its robustness and simplicity. Two states are used to model each of left-stance and right-stance. The first state is maintained for a fixed amount of time T_{hold}, after which there is a transition to the second state. The second state terminates upon footstrike, thereby ending the step and transitioning from left-stance to right-stance or vice-versa. Each state provides fixed target angles for proportional-derivative (PD) controllers, which then compute the internal joint torques to be applied to the forward dynamics simulation. On top of this, a balancing strategy that adds feedback to the torso and swing-leg hip is active at all times, as per [Yin et al. 2007]. The biped parameters, feedback gains, and PD-gains are identical to those used in [Yin et al. 2007].

Our action vector, A, consists of a set of six parameters of the above controller which can then be modified as needed for each walking step, as modeled by two successive states of the FSM. These parameters are: the trunk target angles (first state, second state), the swing hip target angles (first state, second state), the stance ankle target angle (first state), and T_{hold}. The controls for each step are initialized to $A_i \leftarrow A_{base}$, where A_{base} produces a regular walking gait with $36cm$ steps for the humanoid biped. The big-bird character uses the stance knee angle in the optimization instead of the stance ankle angle, and takes $66cm$ steps.

3.2 Optimization

Given an example stepping stone problem, the cost function to be minimized by the offline optimization assigns a cost to both stepping-location errors and deviations from the original control parameters:

$$f(A_1 \ldots A_n) = \sum_{i=1}^{n} (||x_i - x_i^d||^2 + \gamma (A_i - A_{base})^T \mathbf{W} (A_i - A_{base}))$$

where n is the number of steps, x_i^d is the desired stepping location for step i, x_i is the actual stepping location for step i, $\mathbf{W}$ is a diagonal weighting matrix, and $\gamma = 0.1$. We currently use $\mathbf{W} = diag(1,1,1,1,0.05,4)$, where these weight the 6 control parameters in the order described above. Distances are measured in metres, angles in radians, and time in seconds.

A multitude of optimization methods can now be considered. Simultaneous optimization of the many steps in a long sequence of steps is impractical; a sequence of 100 steps will have 600 free parameters for a problem where analytical gradients are not readily available. Sequentially optimizing one step at a time does not provide sufficient anticipation – the state resulting from a step may be incompatible with being able to achieve the following step. As a compromise, we optimize over a three-step sliding window that thus has a total of $3 \times 6 = 18$ free parameters. The sliding window is moved forward one step at a time, meaning that any given action A_i for step i will be optimized three times in succession, each time with a different placement in the sliding window. The action for a given step is not finalized until the sliding window has completed all three passes, and only the final resulting action is retained for later use. After a given window optimization is complete, the final parameters for the last two steps in the window become the initial parameters for the first two steps of the next window placement. The state at the end of the first step of the optimization window becomes the new immutable starting state for the next optimization window.

For each position of the sliding window, gradient-descent optimization is used. Although the optimization window spans events that introduce state discontinuities such as foot strikes, the objective function generally varies smoothly as a function of the optimization parameters for our problem domain. We note that unlike the problems tackled in [Yin et al. 2008], our terrain is flat, has constant friction, and is obstacle free. The failure cases that do arise are discarded, as will be discussed shortly. The gradient is numerically computed using centred finite-differences, and thus makes use of 18×2 simulations spanning the duration of the sliding window. A bisection line search is used to find the local minima in the direction of the gradient, and the gradient-descent operation then repeats. The process stops when the objective function ceases to improve or after 15 iterations have elapsed. The optimization time grows linearly with the number of steps.

It is possible to pose stepping stone sequences that are impossible for the optimization to solve to a desired degree of accuracy. This can result in two possible outcomes. The character may end up losing balance and falling, in which case we terminate the stepping sequence and remove the data associated with the five previous steps. Alternatively, the character may end up performing badly in terms of the proposed objective function. We note however that these latter cases still result in valid training data samples. In the data collection phase we are interested in the actual outcome of an action and not necessarily in the desired outcome that was used to generate it. The optimization is nevertheless important because it shapes all the stepping actions in a consistent fashion, thereby giving the action space a considerable amount of structure which we later rely on.

As shown in Figure 2, the user has three avenues by which to influence the stepping behavior. The problem stepping sequence specification determines the range of step lengths to be accomodated. The base-control parameters A_{base} determine the walking style. Lastly,

it is possible to influence the way in which step adaptations should be made by altering **W** or by choosing a different parameterization of the base controller.

3.3 3D Control

The offline synthesis process applied to our 3D character is largely identical. The goal for the 3D problem is to achieve constrained stepping as seen in the character's sagittal plane. Our 3D character has human-like proportions and mass distribution. The control strategy closely follows that presented in [Yin et al. 2007]. A swing-leg placement strategy is used as a balance strategy in both the sagittal and coronal planes. The base controller uses three states per step. States one and two have fixed-duration dwell times, and state three ends upon the foot strike that demarcates the beginning of the next step. As before, action space A consists of a subset of the target angles used in the states. The specific nine target angles that comprise A are: the sagittal torso angle, with respect to the vertical, in all three states; the sagittal swing hip angle, in all three states; the sagittal stance ankle angle, in states one and two; and the dwell times of states one and two, which are assumed to be identical.

In the objective function, x and x_{targ} are measured in their projection to the character's sagittal plane, as defined by the root link coordinate frame. We use $\mathbf{W} = \mathbf{I}$. We add one additional term to the objective function that measures lateral step deviation, i.e., $(z_i - z_{targ})^2$, where z_{targ} defines a desired lateral foot spacing of $15cm$. While there are no explicit parameters in A to directly affect lateral stepping, this term discourages the use of subspaces of A which introduce unnecessary lateral disturbances. Single-sided finite differences are used for the 3D case. The 3D offline optimization requires 2.5 minutes per example step. The 3D simulation runs $3\times$ faster than real time. For comparison, the 2D simulation runs $5\times$ faster than real time.

4 Motion Analysis

Motions are planned on a step-by-step basis using an abstract model of the step-to-step dynamics. Given the current state, the planner evaluates the state resulting from each of many possible actions for the current step. This can be applied recursively to look two or more steps into the future. In support of this, the example data is used to build a *step-to-step dynamics model (SSDM)*. As shown in Figure 3, the model predicts the state at the start of the next step, $\tilde{s}'$, as a function of the state at the start of the current step, s, and the applied action during the step, A. It also predicts the resulting step length, $\tilde{l}$, and the uncertainty, $\tilde{U}$, of its prediction. The offline data will also be used to predict the subspace of reasonable actions that can be taken from a given state s, which we refer to as the *capabilities model*.

The SSDM makes its predictions based upon the example steps computed during the offline synthesis. We employ k-NN interpolation as a simple form of non-parametric regression. Direct application of this in the high-dimensional state space of $s \times A$ yields poor results and ignores the fact that both the states and actions of the example steps exhibit significant structure. To this end, we manually define a lower-dimensional state space, and use PCA to define a lower-dimensional action space. The low-D action space also plays an important role in the planning process by providing a compact action space to sample from, i.e., that defined by $\hat{A}$, as opposed to having to draw samples from the original high-D action space. We now describe in more detail how each aspect of the SSDM is defined.

Low-dimensional state space: The state s is 18-dimensional for planar characters and much higher for the 3D human model. In order to simplify functions that operate on the character state, we define a reduced dimensional representation of the state given by $\hat{s}_i = (d, v, \theta_{torso}, \theta_{Lhip}, \theta_{Rhip})$, where d, v are the position and velocity of the center of mass as measured with respect to the stance foot, and the remaining parameters are the torso, left-hip, and right-hip angles. These features are motivated by the need to model the state in a compact fashion but still capturing the essence of the state. The specific joint angles that we therefore use are those that drive the heaviest links in the character. It should also be feasible to use an automatically-computed low-dimensional state space representation, although we have not yet explored this. We define the distance between two different states s_i and s_j according to $d^2(s_i, s_j) = d^2(\hat{s}_i, \hat{s}_j) = (\hat{s}_i - \hat{s}_j)^T \mathbf{X} (\hat{s}_i - \hat{s}_j)$ where $\mathbf{X}$ is a diagonal weighting matrix. We use $\mathbf{X} = diag(2, 1, 0.5, 0.5, 0.5)$ for all our characters and styles. For the case of 3D characters, all parameters are identical, but taken in a sagittal projection.

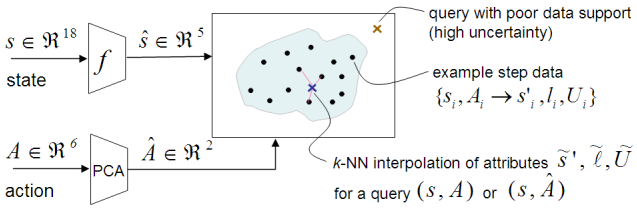

Figure 3: *The step-to-step dynamics model (SSDM). The non-parametric (example-based) model makes predictions using the results of the offline synthesis. The given dimensions for the state and actions spaces are for the 2D bipeds.*

Low-dimensional action space: A principal component analysis (PCA) of all the example actions reveals that there is significant structure in the control actions computed by the offline optimization. 66% of the variation is contained in the first two principal components. We use these first two principal components to define a latent 2D action space $\hat{A}$, whose purpose is to define a unique 2D parameterization for the 6D action space (9D for the 3D character). We project from A down to $\hat{A}$ using the PCA matrices. Where necessary, we estimate A from $\hat{A}$ using kNN interpolation, as will be described shortly. In practice, this gives reconstructions that are better than the 66%-of-variation PCA reconstruction.

***k*NN regression for SSDM:** During planning, the outcomes of many different actions are explored for the current state. In order to efficiently support repeated queries involving the same state s, the regression process first finds the subset of K example steps that have a starting state s_j most similar to s, as measured by $d(s, s_j)$. This subset can be reused for subsequent queries involving the same state. We use $K = 25$. A kD-tree is used to efficiently find the K examples, yielding $3\times$ speedup over straight linear search. The second stage of the regression prediction selects the k nearest neighbors from K based on their distance in reduced action space, $||\hat{A}_i - \hat{A}||$. We use $k = 3$. Lastly, we compute the weights for each of the k samples using $w_i = 1/(d(s_i, s) + \alpha||\hat{A}_i - \hat{A}||)$, followed by a normalization step. The final weights thus take into account distances in both state and action space. We use $\alpha = 1$. Interpolation is carried out using $\tilde{v} = \sum_i w_i v_i$, where v is the value we wish to see interpolated as a function of the query $(s, \hat{A})$. We expect that alternative regression procedures such as Gaussian process latent variable models would likely produce similar results.

During planning, the SSDM is used to predict the resulting state, the step length, and the uncertainty of its own prediction. Once the planner has committed to an action, the SSDM is also used to estimate the full dimensional action, A, that corresponds to $\hat{A}$. This

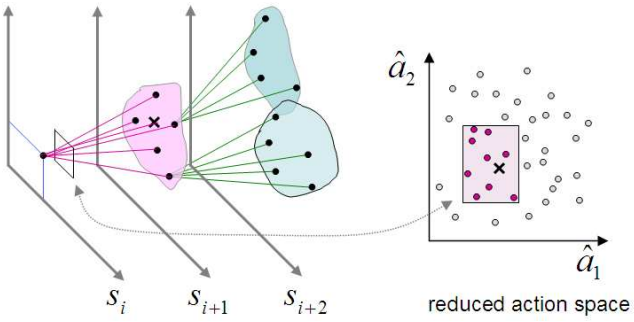

Figure 4: *Two-step finite horizon planning using the reduced action space. Abstractions of future states considered by the planner are shown.*

then becomes the action to be applied. Estimating A using $\hat{A}$ as a latent variable has two advantages over the alternative of using linear reconstruction from the related PCA matrices. It allows A to be modeled as a curved manifold in the high-dimensional space, and it ensures that the model always interpolates and never extrapolates.

Uncertainty model: The uncertainty U provides a way for the SSDM regression to express doubt about the values that it is being asked to estimate. As described in the following section, the planner avoids actions that have uncertain predicted outcomes, either by eliminating them from consideration, or, for fixed-stepping scenarios, adding a penalty cost. For each example step s_i, A_i, s'_i, l_i, we associate an uncertainty estimate U_i, which is computed using leave-one-out cross-validation. We temporarily remove the data for example step i from the set of examples and then use the SSDM to estimate the step resulting from $(s_i, \hat{A}_i)$. This produces $\tilde{s}'_i$ and $\tilde{l}_i$ as estimates of the resulting state and step length, respectively. A comparison of these with their known values is used to compute the uncertainty, which we define as $U_i = d(s_i, \tilde{s}_i) + \beta |l_i - \tilde{l}_i|$. We use $\beta = 1$.

Capabilities model: The example data is also used to provide a model of the subspace of reasonable actions that should be considered when in a given state. This subspace of actions is then used in the planning process. The subset of K example steps having s_j closest to s is used for this, i.e., the same set used in the first stage of the kNN regression. The feasible subspace of actions is defined by the axis-aligned bounding box placed around the K actions in the reduced action space, as illustrated in Figure 4. More generally, the convex hull could also be used.

5 Motion Planning and Execution

Given a good step-to-step dynamics model, a planning algorithm can use this model to accomplish its task. The goal of the planning can be to return the first satisfactory solution, or, alternatively, to return the best solution according to an optimization criterion. Given the constraint of wanting to control simulated characters in real-time, we opt for either finding the first satisfactory solution, or using the best solution that is found within a fixed number of samples of the action space. We explore three types of planning algorithm, each of which samples from the feasible space of actions as defined by the capabilities model. Each algorithm can plan over a multiple-step horizon, as illustrated in Figure 4. Unless otherwise noted, we use a two-step planning horizon. Replanning occurs after each step.

Samples drawn from the space of feassible actions can still result in a number of unacceptable outcomes. The planners reject samples having too much uncertainty, $U \geq U_{max}$ ($U_{max} = 0.75$), or which have an associated predicted step length which results in stepping into a gap. For stepping-stone scenarios, a sample can be rejected for stepping too far from a desired location. We use a threshold of $6cm$ for this. In order to make a final choice among multiple acceptable options, we use a weighted sum of the predicted step error (as summed over the planning horizon) and the uncertainty associated with the immediate action. We weight the uncertainty with a constant $c_u = 0.1$.

We consider three planning techniques.

Regular Sampling: A first possible planning algorithm is to regularly sample the feasible action space, which can be done recursively until a satisfactory plan is constructed for the next n steps. This technique thoroughly examines the action space and is thus our method of choice for fully-constrained stepping-stone problems, as shown in Figure 8.

Random Sampling: For less-constrained problems, such as terrains with sequences of gaps, exhaustive sampling of the action space is typically not required. In such cases we resort to random sampling. For any given step, the random sampling terminates when a satisfactory solution is found or a maximum number of samples for that step has been reached. When applied in the context of building an n-step finite horizon planning, the search operates in a depth-first fashion, and the first solution to successfully achieve n steps is used. The traversal shown in Figure 6 is produced using this planning technique.

All planning for the 3D character uses the random sampling approach with a two-step horizon and an upper bound of 500 samples in order to achieve real-time planning. In order to converge to a regular gait in the absence of obstacles, the base controller A_{base} is used whenever the nearest impending gap is more than $1m$ in front of the character. Otherwise, random sampling with a two-step horizon is employed, using an upper bound of 500 samples. If this fails to produce a solution that strictly avoids the gaps, the solution that best avoids the gap is chosen.

Hybrid: One aspect missing thus far from our planner is the notion of a preferred step length. In order to incorporate this, we develop a hybrid model. A simple footstep planner is first used to determine the lengths of the next two steps to be taken. Steady-state step lengths are preferred ($36cm$ for the 2D and 3D humanoid bipeds, and $66cm$ for big-bird). If a planned step location falls within a gap, it is moved to either before the gap or after, depending on which edge is closer. Given the footstep plan, a first attempt is made to match the planned steps using a sparse regular sampling in the reduced action space. In the absence of an acceptable solution being found, the footstep planning is abandoned and the random sampling planner is invoked.

6 Results

Parameter settings: Our 2D simulation uses optimized Newton-Euler equations of motion and a damped-spring penalty method ground contact model ($k_p = 100000N/m, k_d = 6000Ns/m$) with a time step of $0.0005s$. We compute up to 2000 example steps in order to be able to evaluate the effects of the number of example steps on the quality of our solutions. An average of 11.9 optimization iterations were required per step. For a randomly selected run of 100 steps, the average error in the desired foot placements is $4.3cm$, with 77% of the step errors being less than $6cm$, and 9% more than $12cm$. It is worth noting that not all sequences of steps can be satisfied. The 3D simulation uses the Open Dynamics Engine (ODE) physics simulator and we generate 1000 example steps. All of the parameters described above for the 2D bipeds are quali-

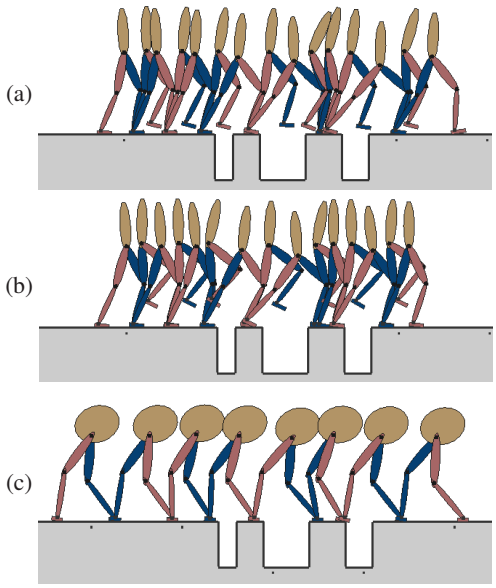

Figure 5: *Walking Style Examples. (a) Regular walk. (b) High-stepping walk. (c) Big-bird walk.*

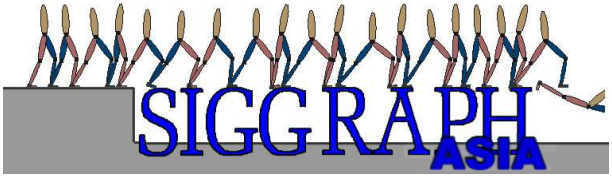

Figure 6: *Constrained terrain walking.*

tiatively similar for the 3D biped. Examples computed using the random-sampling and hybrid planners run in real time, which encompasses most of our examples. Two exceptions are the result shown in Figure 1(b) and Figure 8, which each use a higher sampling rate during planning in order to find feasible solutions to these highly-constrained problems.

Highly constrained walking: The 2D and 3D simulated characters can plan their way across highly constrained terrains. Animations of our results are best seen in the video that accompanies this paper. Figure 8 shows the result of a stepping stone traversal, which fully constrains the desired foot locations for each step. Smooth walking involving a mix of small and large steps requires anticipation and this is provided by the planner. The sequence of steps is different from any it has seen in the example data. The error for the last step is 8.9cm, which is almost twice our average stepping error of 5cm. The regular-sampling planner is used for this particular example, which does not run in real-time.

For the highly-constrained terrains shown in Figures 1(b) and (d), the planner must decide where and how to step. The hybrid planner is used for both examples and runs in real-time for the 3D model terrain traversal (Figure 1(d)). The largest gaps for this latter example are 50cm wide and therefore require at least a 70cm step in order to safely cross with a 20cm foot. The 2D result shown in Figure 1(b) runs slower than real time because of the high sampling rate needed in order to find a feasible solution to this particular problem.

Styles and Characters: Figure 5(a) shows a result for our primary model, a 7-link planar biped. A significant feature of our method

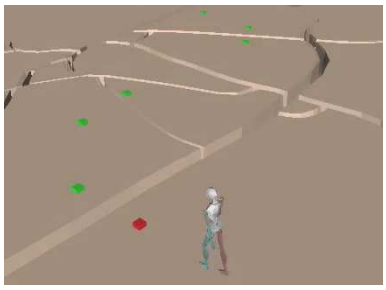

Figure 7: *Following a path while avoiding crevasses.*

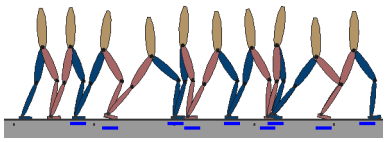

Figure 8: *Results for a stepping-stone problem.*

is that it can be readily applied to alternate walking styles and alternate physical models without making any changes to the synthesis pipeline or any of its parameters. Figure 5(b) shows the terrain traversal simulation that results from using a base walking style that lifts the swing leg much higher during mid-stance. This same style is preserved in the resulting simulated motions. Similarly, we can apply the synthesis pipeline to a new character, such as the big-bird character shown in Figure 5(c). The same synthesis-analysis-synthesis steps are applied, with no changes to any of the parameter settings. Interestingly, the strategy that emerges to deal with constrained foot-placements for the big-bird character is quite different from that of the human-like biped. Qualitatively, it accomplishes much of the required constrained stepping by having the body move at a relatively constant speed and stepping faster when constraints require taking short steps. This is an effective strategy and we speculate that it may result in part from the small foot of this creature. Figure 6 shows a highly constrained walk that can be planned in real time.

3D Path Following: Figure 7 is an example of following a path while using real-time planning to step across crevasses. The path is defined using a sequence of way-points. Turning towards the way point is accomplished on any given step using the stance hip, as described in [Yin et al. 2007]. Once within 50cm of the current waypoint, the next waypoint becomes the goal.

Interaction and Replanning: Replanning at every step allows for interactive unplanned physical interaction of the characters (planar and 3D) with their environment. Figure 9 shows how a mid-stride push will affect the resulting motion. The use of a continually-active balance mechanism [Yin et al. 2007] adapts the placement of the swing foot without delay, although of course an unfortunately timed push could in this way cause a step into a gap. Upon foot-strike, the planning process takes the current state into account when developing its subsequent plan. The video also demonstrates robustness to small changes in terrain height (4cm) while stepping over gaps, as well as an example of the 3D model adapting to a push. While the pre-existing balance mechanism provides the immediate response, it is the step-by-step motion planning that results in the required adaptation with respect to upcoming gaps.

Effect of number of example steps: The locomotion skill of the character is in part a function of the number and span of the motion prototypes that are computed offline. Figure 10 shows the distribu-

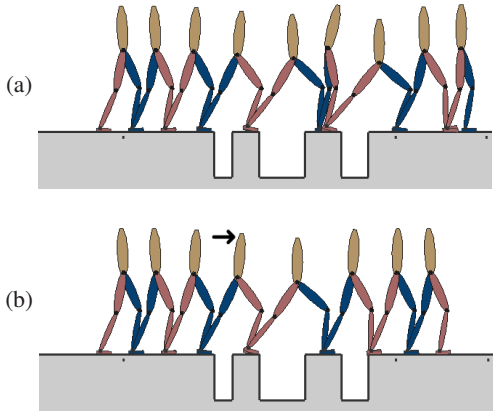

Figure 9: *The effect of a push on the result of a terrain traversal simulation. (a) Resulting motion with no push. (b) Resulting motion with a push. The extra forward speed from the push results in only a single step being taken on the terrain before the last gap.*

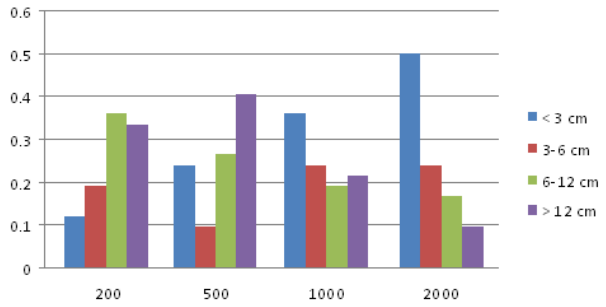

Figure 10: *Performance as a function of the number of example steps. The colored histograms give the distributions of stepping length errors when using the given number of synthesized example steps.*

tion of foot-placement errors for a fully-constrained stepping stone walk across a new stepping-stone sequence for the planar human model. The ability to precisely follow a given stepping-stone sequence improves with more example data. The performance figures are for a terrain that has the same uniform random distribution of requested step lengths as was used for generating the example data.

Effect of planning horizon: The effect of the planning horizon is shown in Figure 11 as evaluated on the planar human biped. A set of fully-constrained stepping stone problems is solved using one, two, and three-step planning horizons. The distribution of stepping errors is shown. A one-step planning horizon performs poorly as it aims to accurately achieve the next foot placement while disregarding subsequent steps. The three-step planning horizon yields results that are qualitatively similar to the two-step planner, having slightly fewer large errors and fewer small errors. We hypothesize that the limitation on the quality of predictions made three steps into the future may be too low to yield an advantage over a two-step time horizon plan. A two-step time horizon further seems to allow sufficient flexibility for the posed problems.

Effect of sampling density during planning: The quality of the motion is a function of the number of samples used per state during the planning process, as shown in Figure 12. We collect stepping-length error data for stepping-stone sequences as a function of the number of samples used by the planner. Planning with regular sampling is used for this test. The solution quality improves as a function of the number of samples used.

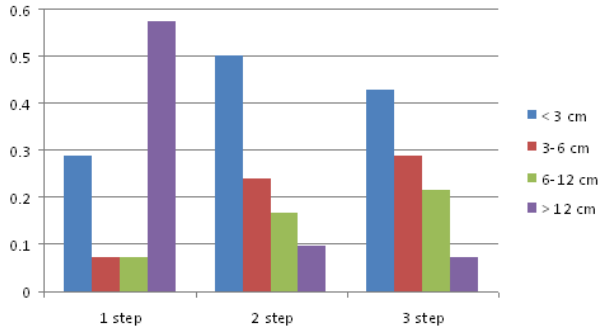

Figure 11: *Effect of varying the planning horizon for a stepping-stone problem. The colored histograms give the distributions of stepping length errors.*

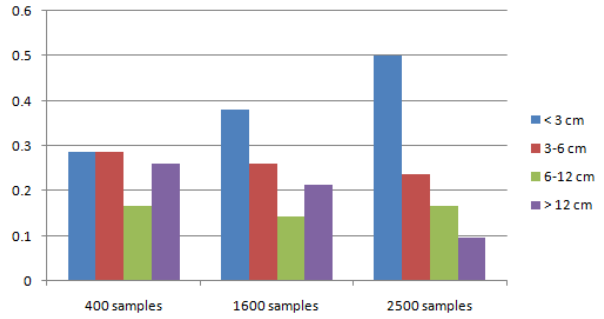

Figure 12: *Effect of the number of interpolated sample actions used for exploring the action space during planning, applied to the stepping-stone problems of the type shown in Figure 8. The colored histograms give the distributions of stepping length errors. Regular sampling planning is used.*

Terrain stress test: Terrains can vary in difficulty and this can affect the ability of our simulated walking to successfully traverse it. A systematic characterization requires defining classes of terrain. As a simple test we develop a set of regular terrains that have a set of 5 gaps of width w and an inter-gap spacing of length s. We then record data for 3 simulated walks across the terrains, with each of these walks beginning at 3 different random distances from the first gap. We also need to define success. A successful walk can sometimes be obtained even without a solid foot placement on the far side of a gap. We define any footfall where less than half of the foot is on the ground to be a failure even if it does not result in a fall. Errors significant enough to cause a fall also count as a failure. Table 1 shows the results for various values of w and s and evaluated for the planar human model. As might be expected, the harder terrains are the ones with wider gaps and less space between the gaps. We note that traversing a 50cm gap requires taking a 70cm step, as measured heel-to-heel, and that our character has 90cm legs.

7 Discussion

The demonstrated technique shares ideas with kinematic data-driven methods such as motion graphs and their many variants. It is perhaps most similar with methods that develop continuously-parameterized kinematic models of motion. However, our work differs in several key respects.

		w			
		0.2	0.3	0.4	0.5
	1	1	1	1	0.93
s	0.75	1	1	1	0.93
	0.5	0.86	0.93	0.86	0.6
	0.25	0.8	0.93	0.66	0.3

Table 1: *Effect of gap width and gap spacing on successful traversal. The fraction of successful steps is given for terrains with gap width w and inter-gap spacing s, as measured in metres.*

First, the model synthesizes its own example data to work from, which allows the method to work in the absence of motion capture data. Much of the power of computer graphics as a medium has always been its ability to portray new worlds and this requires abstract models that do not rely on large quantities of real-world data. The motion models developed in this paper are the product of the physical structure of the given biped, an initial cyclic stepping motion, physics, and the optimization objective function used to compute the offline example steps.

Second, while our approach is data-driven, it is applied to compute control actions that drive physics-based simulations instead of the kinematic interpolation of motions. Many kinematic techniques assume that it is possible to blend or transition between all pairs of stepping motions. We note that the analogous result does not hold in the dynamic setting. For example, a fast long step is infeasible without sufficient initial momentum. Applying a planning strategy analogous to that of motion graphs can be trivially implemented in our setting by considering the set of all stepping actions that begin from a state that is 'close enough' to the current state. In our framework this amounts to considering only the K actions beginning from similar states and which we use to define our action-space bounding box. We have experimented with this type of discrete action space and found it to be consistently inadequate. This motivates the sampling in a continuous action space that is used by our planner, which effectively allows for interpolation between previously observed actions.

Third, it is not obvious how to parameterize the dynamics of the example step data because it involves high-dimensional actions that govern transitions between high-dimensional states. Specific states and actions are unlikely to repeat. The 2D action space manifold, which we parameterize using the first two PCA coordinates of the high-dimensional actions, introduces the necessary structure that makes sampling the action space a tractable proposition.

The results demonstrate that skills which anticipate features of the environment can be developed for real-time, reactive physics-based character animation in a largely automated way. Taken as a whole, the synthesis-analysis-synthesis process aims to automatically create a complete interconnected family of motions rather than individual motions. It establishes close connections between the constraints and objectives that shape a skill and the resulting patterns of action. The technique could likely be extended to other problems such as stepping over objects by using continuation methods to develop the required solutions to the example problems [Yin et al. 2008].

An interesting alternative to the current planning approach is to directly use regression to compute the next desired action. First, a desired sequence of target foot placements can be constructed using a simple fixed model of the minimum and maximum step lengths that the character can take. The action required for any given step could then be predicted directly from the example data set using regression, i.e., $A = q(s, l_1)$ or $A = q(s, l_1, l_2)$, where s is the current state of the character, l_1, l_2 are the next two desired step lengths, and q defines an appropriate regression-based estimator. Experimentation with this scheme revealed a number of limitations. One issue is that the planner needs to be very conservative in its placement of planned steps. Simply assuming that all step sequences satisfying the minimum and maximum step-length bounds of the example problems are equally feasible results in poor performance. A second issue is that it is not obvious how many future steps should be included in estimating the current action. Only considering the imminent step provides insufficient anticipation of upcoming steps. Considering the next two or three steps results in regression queries that have sparse (poor) data support, given that it is unlikely that the example data will contain a sufficiently similar example.

Our work has a number of limitations. The motions synthesized for our human-like 2D and 3D bipeds are not as natural as we would like. In particular, the 2D motion does not make significant use of the ankles and thus does not achieve the toe-off behavior expected in a human gait. Motion capture data could be used in two ways to help correct this. The base cyclic motion could be designed to more accurately mimic motion capture data. Additionally, if stepping sequence motion data were available, similarity to this data could be incorporated into the objective function for the offline synthesis.

The current method focusses on modeling the dynamics of variable length steps as seen in the sagittal-plane. This is sufficient for making the 3D model step across large gaps in the way that humans commonly do, namely stepping forwards across gaps and not sideways. It can be applied to general 3D curved paths as long as the required steps still happen predominantly in the sagittal plane. However, this does not solve the fully general version of the stepping stone problem, namely navigating across an arbitrarily-placed sequence of 3D stepping stones, also perhaps having variations in height. Significant progress has been demonstrated on developing kinematic solutions to this class of problem [Choi et al. 2003; Safonova and Hodgins 2007], although to the best of our knowledge these techniques do not yet, subjectively speaking, exhibit highly agile stepping and turning behaviors that would be indistinguishable from human motion captured directly in the same context. Extending our current technique to allow for diagonal steps would require adding one or two dimensions to the action space and adding extra dimensions to the low-dimensional state representation. We feel the technique would probably scale to accomodate diagonal steps with a small lateral component, although we have not tested such a scenario. Developing control strategies for much more arbitrary highly agile motions in highly constrained 3D environments remains an open problem, although we hope that our work may serve as a significant building block for this class of problem.

8 Conclusions

Developing skills for simulated characters is a challenging problem. We have presented an automated synthesis-analysis-synthesis pipeline for producing simulated walking skills for planar bipeds that are capable of navigating across terrains with gaps and foot-placement constraints. The pipeline supports variation in character design and walking style. It can synthesize constrained walking control strategies in the absence of prior motion data, thereby allowing physically-simulated skills to be developed as a function of what a particular biped structure will allow.

We wish to extend the capabilities of our controllers in a variety of ways. The walking skills do not yet demonstrate the agility of human walking. We wish to develop walking controllers that can stop, start, and perform rapid step adaptations in a way that mirrors human capabilities. The ability to incorporate timing constraints may be important in some situations. Better imitation of human terrain-navigation behaviors may be possible by taking observational data

into account. Incorporating an energy-based term into the offline motion optimizations may yield more plausible motions for real and imaginary characters.

Acknowledgements

We thank the anonymous reviewers for their detailed suggestions for improving the paper. Funding from NSERC (Natural Sciences and Engineering Council of Canada) is gratefully acknowledged.

References

BYL, K., AND TEDRAKE, R. 2008. Approximate optimal control of the compass gait on rough terrain. In *Proc. Int'l Conf. on Robotics and Automation (ICRA)*.

CHESTNUTT, J., AND KUFFNER, J. 2004. A tiered planning strategy for biped navigation. In *Proceedings of the IEEE - RAS / RSJ Conference on Humanoid Robots*.

CHESTNUTT, J., LAU, M., CHEUNG, K. M., KUFFNER, J., HODGINS, J. K., AND KANADE, T. 2005. Footstep planning for the honda asimo humanoid. In *Proc. IEEE Int'l Conf. on Robotics and Automation*.

CHOI, M., LEE, J., AND SHIN, S. 2003. Planning biped locomotion using motion capture data and probabilistic roadmaps. *ACM Transactions on Graphics (TOG) 22*, 2, 182–203.

DA SILVA, M., ABE, Y., AND POPOVIĆ, J. 2008. Interactive simulation of stylized human locomotion. *ACM Trans. Graph. 27*, 3.

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

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

HODGINS, J. K., AND RAIBERT, M. N. 1991. Adjusting step length for rough terrain locomotion. *IEEE Trans. on Robotics and Automation 7*, 3.

HOFMANN, A. G. 2006. *Robust Execution of Bipedal Walking Tasks from Biomechanical Principles*. PhD thesis, Massachusetts Institute of Technology.

HUANG, P. S., AND VAN DE PANNE, M. 1996. A search algorithm for planning dynamic motions. In *Proceedings of the Eurographics Workshop on Computer Animation and Simulation*, 169–182.

KUFFNER, J., NISHIWAKI, K., KAGAMI, S., KUNIYOSHI, Y., AND INOUE, H. 2003. Online footstep planning for humanoid robots. In *Proc. IEEE Int'l Conf. on Robotics and Automation*.

KWON, T., AND SHIN, S. Y. 2005. Motion modeling for on-line locomotion synthesis. In *Proc. ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 29–38.

LASZLO, J. F., VAN DE PANNE, M., AND FIUME, E. 1996. Limit cycle control and its application to the animation of balancing and walking. In *Proc. ACM SIGGRAPH*, 155–162.

LIU, K., HERTZMANN, A., AND POPOVIĆ, Z. 2005. Learning physics-based motion style with nonlinear inverse optimization. *ACM Trans. on Graphics (Proc. SIGGRAPH) 23*, 3, 1071–1081.

MUKAI, T., AND KURIYAMA, S. 2005. Geostatistical motion interpolation. *ACM Trans. on Graphics (Proc. SIGGRAPH)*, 1062–1070.

POPOVIĆ, Z., AND WITKIN, A. 1999. Physically based motion transformation. In *Proc. ACM SIGGRAPH*, 11–20.

RAIBERT, M. H., AND HODGINS, J. K. 1991. Animation of dynamic legged locomotion. In *Proc. SIGGRAPH '91*, 349–358.

REITSMA, P. S. A., AND POLLARD, N. S. 2007. Evaluating motion graphs for character animation. *ACM Transactions on Graphics 26*, 4.

SAFONOVA, A., AND HODGINS, J. K. 2007. Construction and optimal search of interpolated motion graphs. *ACM Trans. on Graphics (Proc. SIGGRAPH)*, Article 106.

SAFONOVA, A., HODGINS, J. K., AND POLLARD, N. S. 2004. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. *ACM Trans. on Graphics (Proc. SIGGRAPH)*, 514–521.

SHARON, D., AND VAN DE PANNE, M. 2005. Synthesis of controllers for stylized planar bipedal walking. In *Proc. Int'l Conf. on Robotics and Automation (ICRA)*.

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

VAN DE PANNE, M. 1997. From footprints to animation. *Computer Graphics Forum 16*, 4 (October), 211–223.

WANG, J. M., FLEET, D. J., AND HERTZMANN, A. 2005. Gaussian process dynamical models. In *Proc. Neural Information Processing Systems Conf.*, 1441–1448.

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

YIN, K., COROS, S., BEAUDOIN, P., AND VAN DE PANNE, M. 2008. Continuation methods for adapting simulated skills. *ACM Trans. Graph. 27*, 3.

ZEGLIN, G., AND BROWN, B. 1998. Control of a bow leg hopping robot. In *Proc. IEEE Intl Conf. on Robotics and Automation*, 793–798.