Autonomous Reactive Control for Simulated Humanoids

Petros Faloutsos[1] Michiel van de Panne[2] Demetri Terzopoulos[3]

[1]University of California at Los Angeles, Department of Computer Science
[2]University of British Columbia, Department of Computer Science
[3]New York University, Courant Institute, Computer Science Department

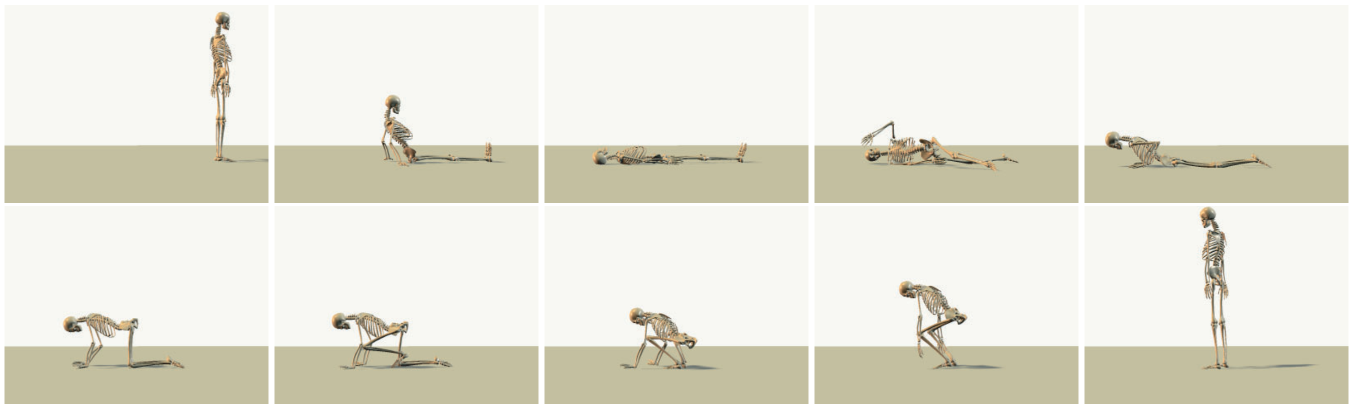


Figure 1: A dynamic humanoid falls to the ground, rolls over, and rises to an erect position, balancing in gravity.

Abstract

We present a framework for composing motor controllers into autonomous composite reactive behaviors for bipedal robots and autonomous, physically-simulated humanoids. A key contribution of our composition framework is an explicit model of the "pre-conditions" under which motor controllers are expected to function properly. Pre-conditions may be determined manually or learned automatically by algorithms based on Support Vector Machine (SVM) learning theory. We demonstrate controller composition and evaluate our composition framework using a family of controllers capable of synthesizing basic actions such as balance, protective stepping when balance is disturbed, protective arm reactions when falling, and multiple ways of regaining an upright stance after a fall.

1 Introduction

Despite the recent progress in bipedal robots [16, 17], systems with broad repertoires of lifelike motor skills remain elusive. While a divide-and-conquer strategy is clearly prudent in emulating the enormous variety of controlled motions that humans may perform, little effort has been directed at how the resulting control solutions may be integrated to yield composite controllers with signi£cantly broader functionalities. The technical challenge is not only to develop motor control strategies for speci£c actions, but also to integrate these controllers into a coherent whole.

In this paper we present to the robotics community our ongoing work [7], which demonstrated a family of composite controllers for dynamically simulated anthropomorphic characters. As an example, Figure 1 illustrates a humanoid, whose physical parameters are consistent with a fully-¤eshed adult male, autonomously performing a complex motor sequence composed of individual controllers. The upright balancing humanoid is pushed backwards by an external force; its arms react protectively to cushion the impact with the ground; the £gure comes to rest in a supine position; it rolls over to a prone position, pushes itself up on all fours, and rises to its feet; £nally it balances upright once again. Autonomously controlled sequences of such intricacy are unprecedented in the humanoid simulation literature.

Our control composition framework is implemented within DANCE, a portable, extensible object-oriented modeling and simulation system [18].[1] DANCE provides a platform that researchers can use to implement animation and control techniques with minimal design and implementation overhead.

1.1 Related Work

The simulation of anthropomorphic £gures is a challenging problem in many respects. Comprehensive solutions must aspire to distill and integrate knowledge from biomechanics, robotics and control. Not surprisingly, a divide-and-conquer strategy is evident in most approaches, focusing efforts on reproducing particular motions in order to yield tractable problems conducive to comparative analysis.

The biomechanics literature is a useful source of predictive models for speci£c motions, typically based on experimental data supplemented by careful analysis. These models target applications such as medical diagnosis, the understanding and treatment of motor control problems, the analysis of accidents and disabilities, and high-performance athletics. Computer simulation is becoming an increasingly useful tool in this domain as the motion models evolve to become more complex and comprehensive [20, 21, 24]. Given the challenge of achieving high-£delity motion models for individual motions, there have been fewer efforts towards integrated solutions applicable to multiple motions. Reference [20] is one such example. A *digital biomechanics laboratory* is proposed by Boston

[1]DANCE is freely available for non-commercial use via the URL: www.cs.ucla.edu/magix/projects/dance.

Dynamics, Inc., [13] as a tool for simulating a wide range of human motion. This currently remains ambitious work in progress.

Robotics research has made remarkable progress in the successful design of a variety of legged robots [23] and, more recently, bipedal robots with anthropomorphic aspirations [16]. Despite their limited motion repertoires and rather deliberate movements, these robotic systems are truly engineering marvels. The work in [1] provides a good summary of behavioral architectures explored in the context of robotics. A 3 DOF ball-juggling robot is described in [3] which uses a theory of behavior composition, although the practicality of extending the method to high-DOF dynamic models of human motions is unclear.

Computer animation has presented interesting results towards the simulation of humanoid characters. Controllers have been successfully designed for speci£c human motions such as walking, running, vaulting, cycling, etc. [11, 15, 31]. Although dynamically simulated articulated characters equipped with an integrated, wide-ranging repertoire of motor skills currently remain an unachieved goal, some positive steps in this direction are evident. Examples include an integrated repertoire of motor controllers for biomechanically animated £sh [25], a methodology for controller design and integration applicable to simple £gures [27], and a technique for transitioning between planar gaits [10]. The work of Wooten [31] is the most relevant, as an example of a sequence of successive transitions between several controllers for human motions such as leaping, tumbling, landing, and balancing. Transitions are realized by including the end state of some controllers in the starting states of other controllers.

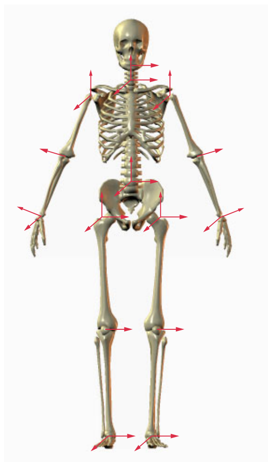
Our work is aimed at creating dynamic humanoids with broadly integrated action repertoires. Unlike previous work focusing on speci£c athletic movements or gaits, our methodology is to begin with a core set of simple actions, including balancing, small steps, falling reactions, recovery from falls, standing up from a chair, and others. Then, we contribute a framework for composing individual controllers, however they may be designed, into more capable control systems for dynamic characters. An interesting technical contribution within our controller composition framework is the introduction of a learning approach for automatically determining controller pre-conditions.

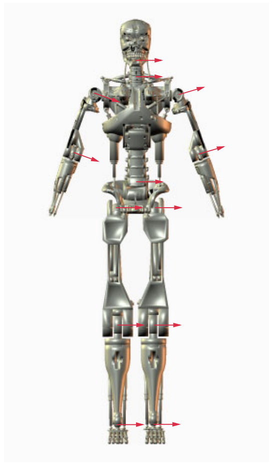
1.2 Overview

The remainder of the paper is organized as follows. Section 2 presents the anthropomorphic models that we use for experimental validations of our results. Section 3 describes a representative set of controllers. Section 4 reviews the pre-condition learning methodology. Section 5 presents the supervisor algorithm for composing controllers. Section 6 discusses our results and their applications. Section 7 concludes the paper and discusses avenues for future research opened up by our work.

2 Humanoid Models

Figure 2 illustrates our experimental dynamic models. The arrows indicate the positions of the joints and their rotational degrees of freedom (DOFs), which are also enumerated in the table. The skeleton model, which is capable of full 3D motion, has 37 DOFs, six of which correspond to the global translation and rotation parameters. The 16 DOF "Terminator" robot model is limited to producing 2D (planar) motion. The leftmost table in the £gure lists the DOFs of the models. The physical properties, such as mass and moments of inertia, of both models are consistent with anthropometric data for a fully-¤eshed adult male, as found in the biomechanics literature (see [30]). In particular, the overall mass of each model is 89.57 kilograms.


(a)


(b)

Joint	*Skeleton model DOFs*	*Robot model DOFs*
Head	1	1
Neck	3	1
Shoulder	2	1
Elbow	2	1
Wrist	2	-
Waist	3	1
Hip	3	1
Knee	1	1
Ankle	2	1

(c)

Joint	Axis	Lower Limit	Upper Limit
Head	x	-45	45
Neck	x	-50	90
	z	-60	60
	y	-80	80
Shoulder	z	-90	90
	y	-80	160
Elbow	y	0	120
	x	-90	40
Wrist	z	-90	90
	y	-45	45
Waist	x	-45	90
	z	-55	55
	y	-50	50
Hip	x	-165	45
	y	-120	20
	z	-20	20
Knee	x	0	165
Ankle	x	-45	50
	z	-2	35

(d)

Figure 2: Anthropomorphic models. (a) 3D-motion skeleton model and (b) 2D-motion "Terminator" robot model, (c) their rotational degrees of freedom (DOFs), and (d) lower/upper joint limits for the skeleton model.

The movement of the rotational degrees of freedom of the models is restricted by the physical limits of the human body. After researching the literature, we have decided to use the joint limits indicated (for the skeleton model) in the rightmost table in the £gure. To ensure that rotations of the £gure's body parts do not exceed the user speci£ed limits, we use a method based on exponential springs, which is widely used in a variety of control problems. If any rotational degree of freedom q_i, exceeds its allowable range of ($q_i^l < q_i < q_i^u$), where the superscripts designate "lower" and "upper" limits, respectively, the exponential springs produce the forces:

$$\begin{aligned} &\texttt{if} \quad (q_i^l - q_i) > \epsilon \quad \texttt{then} \quad f_i^l = k_s(e^{k_s^e(q_i^l - q_i)} - 1) - k_d\dot{q}_i, \\ &\texttt{if} \quad (q_i - q_i^u) > \epsilon \quad \texttt{then} \quad f_i^u = k_s(e^{k_s^e(q_i - q_i^u)} - 1) - k_d\dot{q}_i, \end{aligned}$$

depending on the limit that has been violated. We have determined that the spring constants k_d = 10.0 and k_s^e = 1.0 produce satisfactory behavior.

Motor controllers need information about the state of the £gure, where it is facing, whether it is balanced, etc. Controllers also need

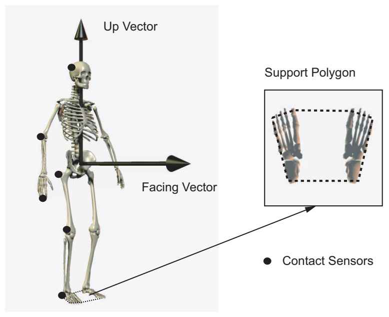


Figure 3: Common sensors.

to have information about the environment, such as body/ground contact points, the slope of the terrain at contact points, the position of obstacles, etc. Most of the information on the £gure can be computed from the state parameters; however, it is often more convenient to use higher-level sensors that are more intuitive, can be computed once per time step, and can be shared among controllers. In our current implementation, each controller has full access to the internal data structures of the system, including all the information associated with any £gure or object in the system. This allows the controllers to de£ne arbitrary sensors that keep track of necessary information such as state parameters for feedback loops and the state of the environment. Common sensor values include:

- Support polygon. The support polygon $\mathcal{S}$ is de£ned by the convex hull of the feet that are in contact with the ground, and it is crucial for the balance of the £gure.
- Center of mass information. The position $\mathbf{c}$, velocity $\dot{\mathbf{c}}$, acceleration $\ddot{\mathbf{c}}$, and relative position of the center of mass with respect to the support polygon.
- Pelvis center of mass information. The position $\mathbf{c}^h$, velocity $\dot{\mathbf{c}}^h$, acceleration $\ddot{\mathbf{c}}^h$, and relative position of the pelvis center of mass with respect to the support polygon.
- Contact information. An indication of whether the feet, head, pelvis and thighs are in contact with the ground.
- Orientation. The *facing* vector $\mathbf{v}^f$ and *up* vector $\mathbf{v}^u$ u of the pelvis, indicating the direction that the pelvis faces and how far it leans, respectively.

Figure 3 shows the support polygon, the facing vector and the up vector relative to the skeleton model.

Most of the computational burden in our approach lies in the numerical simulation of the equations of motion. The computations associated with the controllers and our composition framework are negligible in comparison. In general, the reduced- DOF, 2D-motion robot model simulates in real time on a 733 MHz Pentium III computer system, whereas the 3D-motion skeleton model runs between 5 and 9 times slower than real time.

3 Composable Controllers

We have proposed a simple but effective framework for composing specialist controllers into more capable systems for simulated £gures [7]. In our controller composition framework, individual controllers are black boxes which are managed by a simple supervisor controller. Regardless of their encapsulation, our method requires individual controllers to de£ne *pre-conditions*, *post-conditions*, and *expected performance*. Pre-conditions, denoted $\mathcal{P}$, are a set of conditions over the state of the £gure and the environment. Pre-conditions are determined either manually, as in the examples below, or they are learned automatically, Section 4. If the pre-conditions are met, then the controller can operate and possibly enable the £gure to satisfy the post-conditions, denoted $\mathcal{O}$, the range of states that the £gure may be in after the execution of the controller. Thus, the controller realizes a transition between a domain of input states to a range of output states for the £gure. Because of unexpected changes in the environment, however, this transition may not always succeed, which motivates the notion of expected performance, denoted $\mathcal{E}$; the controller should be able to evaluate its performance in order to detect failure at any point during its operation. To do this, the controller must continually be aware of the current and expected state of the £gure or the environment. Any controller that de£nes pre-conditions, post-conditions and expected performance can be part of our composition scheme explained in Section 5.

Most of the controllers for our models are based on pose control, which has often been used both for articulated objects [26] and soft objects [6]. Pose control is based on cyclic or acyclic £nite state machines with time transitions between the states. Each state of the controller can be static or can depend on feedback parameters. For some of our controllers, we use continuous control, in the sense that the control parameters are tightly coupled with some of the feedback sensors. The balance controller presented below is an example of this.

We now present a few of the individual, specialist controllers that we have implemented for our humanoid characters and we describe in detail their analytical, composable APIs. Let us £rst de£ne the following quantities and symbols: The state $\mathbf{q} = [\mathbf{x} \quad \dot{\mathbf{x}}]$ of an articulated £gure is the vector of generalized joint angles $\mathbf{x}$ and angular velocities $\dot{\mathbf{x}}$, where the dot indicates a time derivative. The position and velocity of the center of mass are denoted as $\mathbf{c}$ and $\dot{\mathbf{c}}$ respectively. The support polygon of a £gure is denoted as $\mathcal{S}$.

3.1 Default Controller

The default controller is activated when no other controller requests control of the biped. Its goal is to perform a sensible action in any given situation. In the absence of a better understanding of the situation, the most sensible thing to do is to keep the £gure in a comfortable position. We currently distinguish between two different situations, standing in place and lying on the ground. In the £rst case, the controller attempts to maintain the £gure's upright stance using moderate force while keeping the arms loose. If the £gure is leaning by more than a given threshold slant, then it is considered to be in a lying position, in which case the controller makes the character assume a relaxed pose. Thus far, these two strategies have worked well, in the sense that they bring the £gure smoothly into a perceived comfortable position. The default controller faces the dif£cult task of encompassing all situations for which we have not yet designed appropriate controllers. It therefore represents only a starting point for future improvements.

3.2 Balancing

Balancing in a quiescent, upright stance is a complex biomechanical control phenomenon that depends on different factors, such as the distance between the feet, and the presence of (or lack of) visual feedback [4]. A considerable body of research aims to understand the sensory information [28] and re¤ex responses that humans use to maintain quiet stance [8]. The strategies that people employ as a response to disturbances during quiet stance are generally divided into hip strategies and ankle strategies depending on whether the hips or the ankles are the dominant regulators of the postural stability. A comprehensive analysis of balance strategies during quiet stance focusing on ankle control can be found in [9]. Most researchers in biomechanics seem to agree that ankle strategies are more likely to occur in response to small disturbances, while hip strategies occur in response to larger disturbances.

Our balance controller is responsible for maintaining a natural standing posture. It is based on an inverted pendulum model that uses the ankles to regulate the body sway [8]. Despite the fact that the body of the £gure is not as rigid as the inverted pendulum hypothesis suggests, the approximation works well in practice. Our balance controller uses an ankle angle of 0.06 radians as the equilibrium position. For this controller, the articulated body must be in a balanced upright position, the velocity and acceleration of the center of mass should not exceed certain threshold values as explained by [19], and both feet must maintain contact with the ground at all times. The controller can tolerate small perturbations of the posture and the velocity/acceleration of the center of mass by stiffening the ankle joints. For larger accelerations of the center of mass, the controller actively actuates the ankle joint to reduce the acceleration of the center of mass. The post-conditions are similar to the pre-conditions. In mathematical form:

$\mathcal{P}$:

Velocity: $|\dot{\mathbf{c}}| < 0.3$ m/sec.
Balance: projection$(\mathbf{c}) \in \mathcal{S}$.
Posture: (upright) $(1/n)\sum_i \sqrt{(q_i - q_{0,i})^2} < 0.1$ rad,
where $i =$ (thigh, knee, waist), $\mathbf{q}_0 = \mathbf{0}$,
and n is a normalization parameter.
Contact: feet on ground.

$\mathcal{O}$:

Velocity: $|\dot{\mathbf{c}}| < 0.05$ m/sec.
Balance: projection$(\mathbf{c}) \in \mathcal{S}$.
Posture: (upright) $(1/n)\sum_i \sqrt{(q_i - q_{0,i})^2} < 0.1$ rad,
where $i =$ (thigh, knee, waist), $\mathbf{q}_0 = \mathbf{0}$,
and n is a normalization parameter.
Contact: feet on ground.

The expected performance $\mathcal{E}$ is identical to the pre-conditions. We enhance the behavior of our balance controller in a simple fashion by kinematically simulating the biped's visual attention. In particular, we apply Perlin noise [22] to the degrees of freedom of the neck that makes the biped look around in its environment.

Because of the relatively simple task that this controller has to accomplish and the inherent stability of the simple ankle strategy that we employ, the balance controller can be used successfully on slightly different terrains and £gures. Nevertheless, the controller could be enhanced to employ more complex strategies, especially as responses to larger external disturbances. For example, a simulated biped should attempt to maintain balance by shifting its weight, or bending at the waist. If the biped cannot maintain balance, it must then resort to taking a step or even initiating a fall behavior.

3.3 Falling

The manner in which people fall depends upon a number of factors, such as their physique, their age, and their training. Involuntary falling reactions are very common in everyday life, especially among young children and the elderly. They are probably the most common reason behind fracture injuries among the elderly. The work in [12] shows that, during a fall, the elderly are more likely to impact their hip £rst as compared to younger adults falling under the same conditions. Our fall controller is designed with the average adult in mind. Its main action is to absorb the shock of the impact using mostly the hands. The work in [32] provides a way to distinguish falls from normal activities based solely on velocity characteristics. The pre-conditions of our fall controller de£ne a larger acceptable region in velocity space than the one speci£ed by [32] because they are de£ned in accordance with those of the balance controller. All situations that are beyond the capabilities of the latter should be handled by the fall controller:

$\mathcal{P}$:

Vertical Velocity: $\dot{c}_v < 0.3$ m/sec.
Balance: projection$(\mathbf{c}) \notin \mathcal{S}$.
Contact: hip not on ground, hands not on ground.

$\mathcal{E}$:

If falling forward, face down $v_y^f < 0.1$.
If falling backward, face up $v_y^f > -0.1$.
Contact with the ground in 3 seconds.

$\mathcal{O}$:

Either
Velocity: $|\dot{\mathbf{c}}| < 0.3$ m/sec.
or
head on ground.

The pre-conditions ensure that if the £gure is not balanced, then the fall controller bids to take over. The fall controller succeeds when the velocity and acceleration of the biped are brought close to zero or when the head touches the ground. The expected performance ensures that the biped keeps on falling in the same direction. In addition, it requires (a) that the £gure's facing direction does not reverse, something which might happen when falling from a great height, and (b) that the £gure touches the ground within 3 seconds in order to ensure that the fall was from a short height. Our implementation of the fall controller computes the direction of the fall and responds accordingly. It can therefore handle a variety of pushes. Figure 4 shows snapshots of falls in different directions. The second frame in Figure 1 also demonstrates the action of the fall controller within a fall-and-recover sequence. The controller is relatively robust and it can be used on different bipeds and ground models.

4 SVM Learning of Pre-Conditions

In this section, we introduce an automatic, machine learning approach to determining pre-conditions, which is based on systematically sampling the performance of controllers. Our method uses a machine learning algorithm attributed to Vapnik [29] known as *Support Vector Machines* (SVMs), which has recently attracted much attention, since in most cases the performance of SVMs matches or exceeds that of competing methods.

SVMs are a method for £tting functions to sets of labeled training data. The functions can be general regression functions or they can be classi£cation functions. In our application, we use simple classi£cation functions with binary outputs which encode the success or failure of a controller. Burges [2] provides an excellent tutorial on SVMs.

To apply the SVM technique to the problem of determining controller pre-conditions, we train a nonlinear SVM classi£er to predict the success or failure of a controller for an arbitrary starting state. Thus, the trained SVM demarcates the boundary of regions in the £gure's state space wherein the controller can successfully do its job. Training sets comprising examples $\{\mathbf{x}_i, y_i\}$ are generated by repeatedly starting the dynamic £gure at a stochastically-generated initial state $\mathbf{x}_i$, numerically simulating the dynamics of the £gure under the in¤uence of the controller in question, and setting $y_i = +1$ if the controller succeeds or $y_i = -1$ if it fails.

The distribution of the stochastically-generated initial states is of some importance. The sample points should ideally be located close to the boundaries which demarcate the acceptable pre-condition region of state-space. However, these boundaries are in fact the unknowns we wish to determine and thus we must resort to a more uniform sampling strategy. Unfortunately, the high dimensionality of the state-space precludes regular sampling. We thus adopt the following stochastic process to generate a suitable distribution of initial states: First, a nominal initial state is chosen, based upon the designer's knowledge of the controller. A short-duration simulation (typically 0.3s) is then carried out from this initial state while a randomized perturbation process is executed.

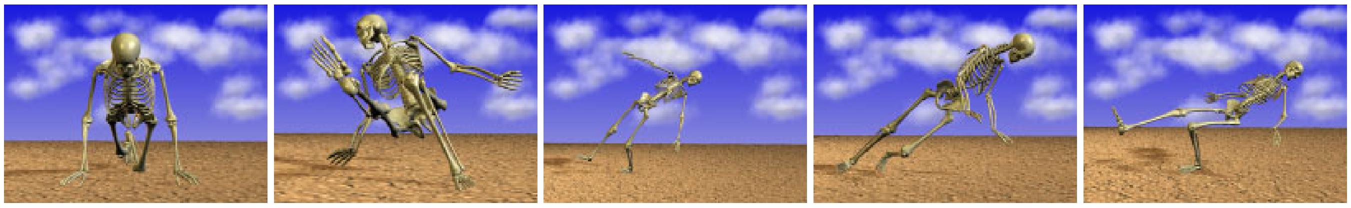


Figure 4: Falling in different directions

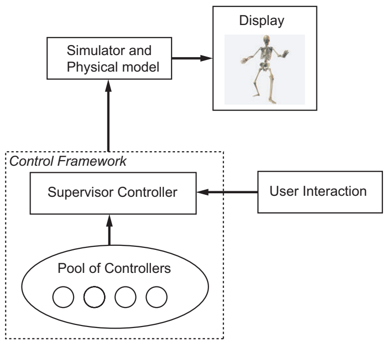


Figure 9: System overview

This currently consists of applying an external force of random (but bounded) magnitude and random direction to the center-of-mass of the pelvis. Simultaneously, the biped's joints are perturbed in a stochastic fashion by setting randomized offset target angles for the joints and using the biped's PD joint controllers to drive the joints towards these perturbed positions. While the perturbation strategy is admittedly ad-hoc, we have found it to be effective in sampling the pre-condition space, as is validated by the online use of the learned pre-condition models.

We employ T. Joachims' SVM^{light} software which is available on the WWW [14]. The software can accommodate large training sets comprising tens of thousands of observations and it ef£ciently handles many thousands of support vectors. It includes standard kernel functions and permits the de£nition of new ones. It incorporates a fast training algorithm which proceeds by solving a sequence of optimization problems lower-bounding the solution using a form of local search. It includes two ef£cient estimation methods for error rate and precision/recall.

The SVM training phase can take hours in our application, but this is done off-line. For example, on a 733 MHz PIII computer, the SVM training time for a training set of 8,013 observations is 2,789 seconds using the polynomial kernel, 2,109 seconds using the linear kernel, and 211 seconds using the radial kernel. For a training set of 11,020 observations, the training time is 8,676 seconds using the polynomial kernel, 3,593 seconds using the linear kernel, and 486 seconds using the radial kernel. Once trained, the SVM classi£er can provide answers on-line in milliseconds.

5 Supervisor Controller Algorithm

Figure 9 presents an overview of our control system. At each time step of the physics-based simulation, the supervisor controller £rst checks whether it needs to initiate a bid process, and proceeds to do so if the user-speci£ed target state has changed or if there is no active controller (other than a default controller). During the bidding process, all available individual controllers determine whether their pre-conditions are satis£ed and, if so, they bid for control over the dynamic £gure by returning a priority number (detail in [5]). The supervisor controller selects from among the collection of bidding controllers the one that returns the highest priority, registers it as the active controller, and invokes a method associated with the controller which implements its control strategy. The method returns to the supervisor controller a status parameter. If the status parameter indicates that the controller has failed, then a new bidding process is initiated.[2] Along with the status parameter, the method returns target values for some or all of the dynamic £gure's degrees of freedom along with associated stiffness and damping parameters, which are used by a set of proportional-derivative controllers to calculate the actual control torques. Alternatively, the active controller can choose to apply torques directly to the £gure and return no values for the supervisor's proportional-derivative controllers.

In the case where no available controller bids for control, the supervisor controller activates the default controller, a generic controller, which we will describe in more detail later, that tries to do something sensible with the £gure when no specialist controller is able to assume control.

Some controllers automatically bid for control over the £gure when their pre-conditions are met; hence, many controller transitions occur autonomously, such as taking a protective step in response to a loss of balance. However, other actions are initiated voluntarily, and the associated controllers become active only at the request of the user. For example, a £gure balancing upright can be instructed to remain standing, to sit-down, to walk, or to take a dive. Currently, the user directs voluntary actions by interactively entering command strings to the supervisor controller. These commands increase the suitability score of the designated controller and forces invocation of the arbitration process which selects and activates the designated controller. The control of voluntary motions could be delegated to a high-level planner, but motion planning is beyond the scope of our current work.

6 Results

We now present two sequences of autonomous and user-instructed actions that our simulated humanoids are able to perform. The increased simulation ef£ciency of the 2D-motion robot model permits a more productive controller design phase. In addition, the two dimensional case is more robust than the three dimensional one and it depends less on the speci£c biped and ground model. However, the sequence we are able to achieve for the full scale three dimensional skeleton model shows that our method can be used successfully for both cases. Inevitably, robust controllers for complex motions will be developed by ourselves or by others. Our system can integrate these controllers as they become available and produce a powerful composite controller.

The reduced dimensionality of the robot model allowed us to develop a relatively large number of controllers. The sequence shown in Fig 10 involves 13 controllers: balance, prone-to-

[2] An additional check avoids an in£nite loop when a badly designed controller bids for control and immediately fails.

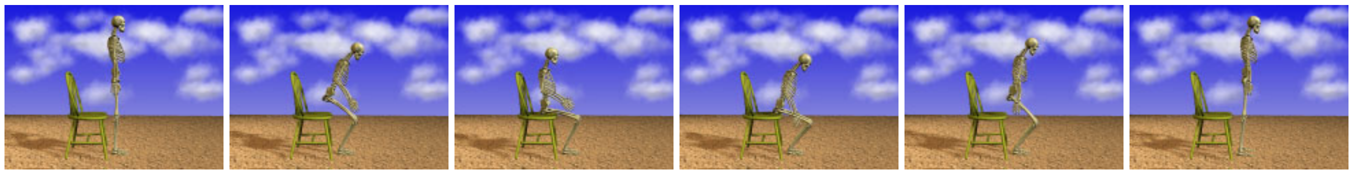


Figure 5: Sitting and rising from a chair

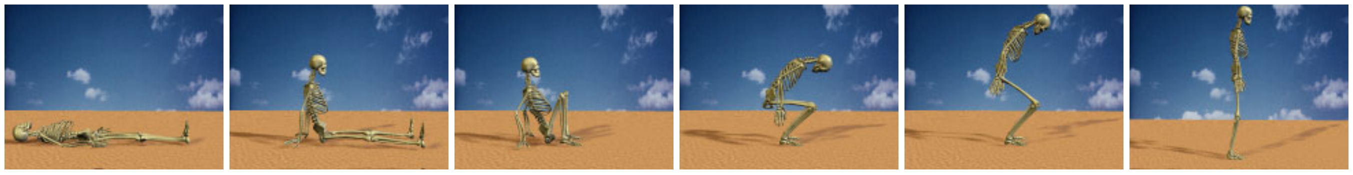


Figure 6: Rising from a supine position on the ground and balancing erect in gravity.

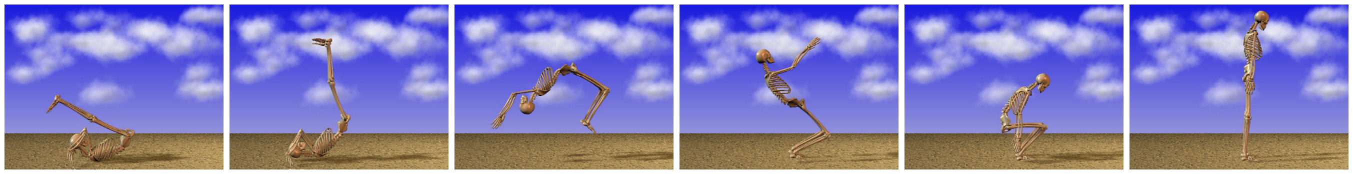


Figure 7: Kip move: A more vigorous way of getting up from the supine position as in the £rst frame of Fig. 6.

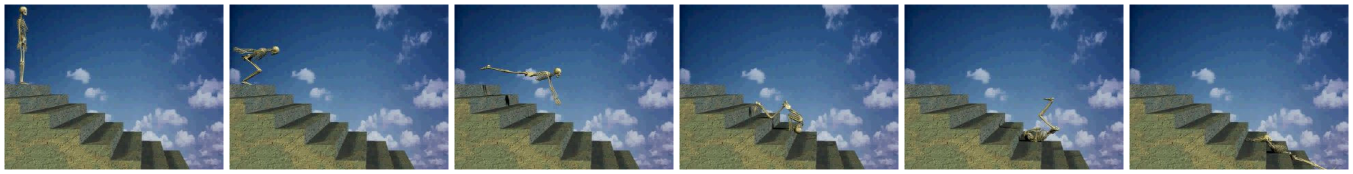


Figure 8: Ouch!

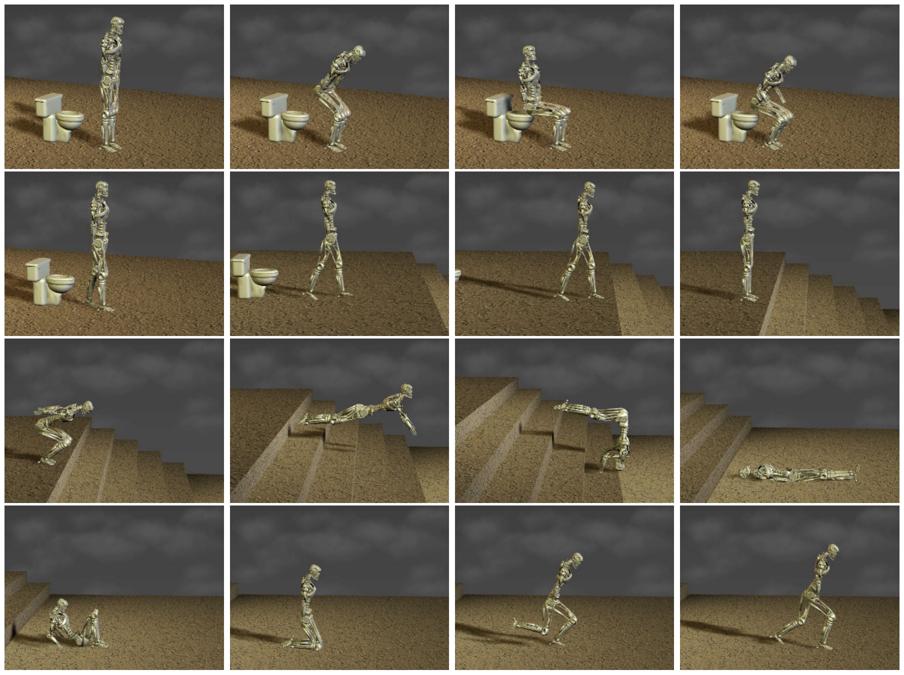


Figure 10: The terminator sequence, left to right and top to bottom.

kneel, supine-to-kneel, kneel-to-crouch, crouch-to-stand, standto-sit, sit-to-crouch, protective-step, fall, walk, plunge-and-roll, double stance-tocrouch, and the default controller. The plunge-and-roll, stand-to-sit, sit-to-crouch and walk controllers bid for control of the biped only under the direction of the user. The remaining controllers act autonomously.

The three dimensional sequence shown in Figure 1 is created interactively. The skeleton is equipped with the following controllers: balance, fall, roll-over, prone-crouch, crouch-to-stand and the default controller. All controllers are autonomous in this case; as the skeleton goes though different con£gurations, it automatically reacts to the current situation activating the most appropriate controller among those that are available. First, the user pushes the biped backwards. The composite controller activates a fall behavior that tries to absorb the shock. With the £gure in a supine con£guration, the roll-over controller brings the skeleton to a prone position which makes it possible for the prone-to-crouch controller to take over. When the £gure reaches a crouching posture, the prone-to-crouch controller succeeds and the crouch-to-stand controller brings the £gure to an upright position, which allows the balance controller to take over again. This sequence is less robust than the 2D counterpart. The prone-to-crouch and the crouch-to-stand controllers are particularly sensitive to the ground model.

Figures 5,6,7 and 8 show various sequences of actions that our models can perform. More details on our results can be found in [5]. Associated animations can be found at www.cs.ucla.edu/∼pfal/research.

7 Conclusion

This paper has presented a framework for composing dynamic controllers. Our framework has been implemented within a freely available system for modeling and animating articulated £gures. To our knowledge, our system is the £rst to demonstrate a dynamic anthoropomorphic £gure with controlled reactions to disturbances or falls in any direction, as well as the ability to pick itself up off the ground in several ways, among other controlled motions. We hope that our system will foster collective efforts among numerous practitioners that will eventually result in complex composite controllers capable of synthesizing a full spectrum of human-like motor behaviors.

Given the enormous challenge of building controllers capable of large repertoires of dynamic human-like motion, it is inevitable that the work presented in this paper is incomplete in many ways. Published control methods for 3D walking, running, and stair climbing make obvious candidates for integration into our system. Coping with variable terrain and dynamic environments are dimensions of added complexity that should provide work for years to come. Finally, intelligently integrating controllers which affect only subsets of DOFs needs to be addressed in order to allow for the parallel execution of controllers.

References

[1] Ronald C. Arkin. *Behavioral Robotics*. MIT Press, 1998.

[2] C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):955–974, 1998.

[3] R. R Burridge, A. A. Rizzi, and D. E Koditschek. Sequential composition of dynamically dexterous robot behaviors. *The International Journal of Robotics Research*, 18(6):534–555, June 1999.

[4] B. L. Day, M. J. Steiger, P. D Thompson, and C. D. Marsden. Effect of vision and stand width on human body motion when standing: implications for afferent control of lateral sway. *Journal of Physiology*, 469:479–499, 1993.

[5] Petros Faloutsos. *Composable Controller for Physics-based Character Animation*. PhD thesis, Univeristy of Toronto, DCS, Toronto,Canada, 2001.

[6] Petros Faloutsos, Michiel van de Panne, and Demetri Terzopoulos. Dynamic free-form deformations for animation synthesis. *IEEE Transactions on Visualization and Computer Graphics*, 3(3):201–214, 1997.

[7] Petros Faloutsos, Michiel van de Panne, and Demetri Terzopoulos. Composable controllers for physics-based character animation. In *Proceedings of ACM SIGGRAPH*, pages 251–260, Los Angeles, August 2001.

[8] R. C Fitzpatrick, J. L. Taylor, and D. I. McCloskey. Ankle stiffness of standing humans in response to imperceptible perturbation: re¤ex and task-dependent components. *Journal of Physiology*, 454:533–547, 1992.

[9] Plamen Gatev, Sherry Thomas, Thomas Kepple, and Mark Hallet. Feedforward ankle strategy of balance during quiet stance in adults. *Journal of Physiology*, 514(3):915–928, 1999.

[10] J. K. Hodgins. Biped gait transitions. *Proceedings of 1991 IEEE International Conference on Robotics and Automation*, pages 2092–2097, April 1991.

[11] J. K. Hodgins, W. L. Wooten, D. C. Brogan, and J. F. O'Brien. Animating human athletics. *Proceedings of SIGGRAPH 95, ACM Computer Graphics*, pages 71–78, 1995.

[12] E. T Hsiao and S. N Robinovitch. Common protective movements govern unexpected falls from standing height. *Journal of biomechanics*, 31:1–9, 1998.

[13] Boston Dynamics Inc. The digital biomechanics laboratory. `www.bdi.com`, 1998.

[14] T. Joachims. Making large-scale svm learning practical. advances in kernel methods. In B. Schölhopf, C. Burges, and A. Smola, editors, *Support Vector Learning*. MIT-Press, 1999. http://www-ai.cs.uni-dortmund.de/DOKUMENTE/joachims_99a.pdf.

[15] Joseph F. Laszlo, Michiel van de Panne, and Eugene Fiume. Limit cycle control and its application to the animation of balancing and walking. *Proceedings of SIGGRAPH 96*, pages 155–162, August 1996.

[16] Honda Motor Co. Ltd. www.honda.co.jp/english/technology/robot/.

[17] Sony Co. Ltd. `www.sony.co.jp/en/SonyInfo/News/Press/200011/00-057E2/`.

[18] Victor Ng and Petros Faloutsos. Dance: Dynamic animation and control environment. Software system, www.cs.ucla.edu/magix/projects/dance.

[19] Yi-Chung Pai and James Patton. Center of mass velocity-position predictions for balance control. *Journal of biomechanics*, 30(4):347–354, 1997.

[20] Marcus G. Pandy and Frank C. Anderson. Three-dimensional computer simulation of jumping and walking using the same model. In *Proceedings of the VIIth International Symposium on Computer Simulation in Biomechanics*, August 1999.

[21] Marcus G. Pandy, Felix E. Zajac, Eunsup Sim, and William S. Levine. An optimal control model for maximum-height human jumping. *Journal of Biomechanics*, 23(12):1185–1198, 1990.

[22] K. Perlin. An image synthesizer. *Computer Graphics*, 19(3):287–296, July 1985.

[23] M. H. Raibert. *Legged Robots that Balance*. MIT Press, 1986.

[24] Cecile Smeesters, Wilson C. Hayes, and Thomas A. McMahon. Determining fall direction and impact location for various disturbances and gait speeds using the articulated total body model. In *Proceedings of the VIIth International Symposium on Computer Simulation in Biomechanics*, August 1999.

[25] Xiaoyuan Tu and Demetri Terzopoulos. Arti£cial £shes: Physics, locomotion, perception, behavior. *Proceedings of SIGGRAPH 94*, pages 43–50, 1994.

[26] M. van de Panne. Parameterized gait synthesis. *IEEE Computer Graphics and Applications*, pages 40–49, March 1996.

[27] Michiel van de Panne, Eugene Fiume, and Zvonko Vranesic. Reusable motion synthesis using state-space controllers. *Computer Graphics (SIGGRAPH 90 Proceedings)*, 24(4):225–234, August 1990. ISBN 0-201-50933-4. Dallas, Texas.

[28] Herman van der Kooij, Bart Koopman, Ron Jacobs, Thomas Mergner, and Henk Grootenboer. Quanti£cation of sesonry information in human balance control. In *Proceedings of the 20th Annual International COnference of the IEEE Engineering in Medicine and Biology Society*, volume 20, pages 2393–2396, 1998.

[29] V. Vapnik. *Estimation of Dependecies Based on Empirical Data (in Russian)*. Nauka, Moscow, 1979. English translation Springer Verlag, New York, 1982.

[30] David A. Winter. Muscle mechanics. In *Biomechanics and Motor Control of Human Movement*, chapter 7, pages 165–189. John Wiley and Sons, Inc., second edition, 1990.

[31] Wayne Wooten. *Simulation of Leaping, Tumbling, Landing, and Balancing Humans*. PhD thesis, Georgia Institute of Technology, March 1998.

[32] Ge Wu. Distinguishing fall activities from normal activities by velocity characteristics. *Journal of Biomechanics*, 33:1497–1500, 2000.