CONTROL FOR SIMULATED HUMAN AND ANIMAL MOTION

Michiel van de Panne [1]

Department of Computer Science
University of Toronto

Abstract: The intelligent and graceful motion control exhibited by animals and humans remains a great challenge to replicate. We summarize some of the disciplines and applications related to human and animal motor control. We give a computer science perspective on some of the successes and failures in this area and provide a list of questions which need to be addressed. This is followed by an overview of several methods we have used for controlling simulated human and animal motion.

Keywords: motion control, locomotion, control synthesis, computer animation

1. INTRODUCTION

The topic of animal and human motion control is an interdisciplinary one, as reflected in the wide range of conferences and journals which address some part of this subject. The disciplines include control theory, biomechanics, robotics, artificial intelligence, as well as computer animation. Each of these disciplines provides a unique perspective on the problem.

While the relationship of control and robotics to animal and human motion control does not require much elaboration, the ties to artificial intelligence (AI) and computer animation are worthy of further explanation. Motion control problems share many of the key characteristics often associated with AI problems, namely learning, planning, choice of representation, and determining strategies for interacting with a real world that is complex and dynamic. Motion control can also be considered as a key component of a *bottom up* approach to AI, in which problems of interacting with the real world are considered to be at least as interesting as abstract reasoning tasks.

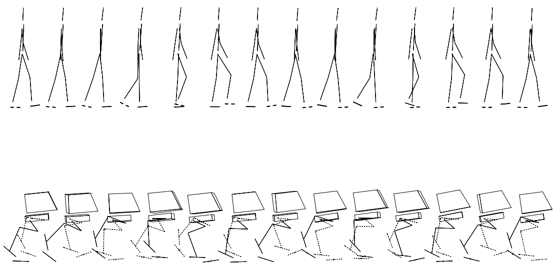


Fig. 1. Simulated movements for a human model and robot.

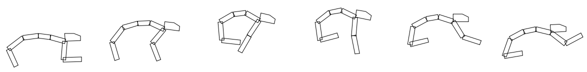


Fig. 2. A bounding cheetah model.

Within the computer graphics community, computer animation has proved to be a surprisingly fertile ground for experimentation in motion control strategies. Graceful gymnasts, leaping lions, and lumbering dinosaurs are all the product of physics and control. They are also the kinds of motions which animators wish to reproduce and manipulate. A considerable amount of effort has been focussed on using physically-based simulations in conjunction with control methods in order to produce natural character movement. Figures 1 and 2 are example results of these types of experiments.

---

[1] `van@dgp.utoronto.ca`
We gratefully acknowledge the financial support of NSERC (Canada) and ITRC (Ontario).

1.1 *Computer Animation and Control*

Traditionally, computer animations have been created using *keyframing*, which has the animator specify the position of a character in selected 'key' frames, which are then smoothly interpolated in order to yield the desired motion. Keyframing can be a tedious process, however. The motion must be specified down to the last detail and multiple iterations are often required to get a motion just right.

An alternative animation technique has been the use of *motion capture* techniques. A variety of technologies exist for tracking and recording the real-time motion of multiple markers placed on humans or animals. This data can then be used to generate the movement of an equivalent computer-animated model. Motion capture is not a panacea, however, because it is not clear as to how a captured motion can be generalized to construct novel motions or how to fit motions to characters of different proportions.(Witkin and Popovi'c, 1995; Bruderlin and Williams, 1995; Unuma *et al.*, 1995)

The use of physical simulations offers the promise of producing realistic animated motion in a more automated fashion. With the improvement of physical simulation packages, the interactive dynamic simulation of character motion has now become a reality. The catch, however, has been that of solving the associated control problems. Just how do we control the muscles in a character in order to achieve a desired motion such as walking?

Several different flavors of the motion control problem occur in the context of computer animation. Applications such as feature films require tightly-scripted control over the motion. A particular motion can be reworked over a period of days in order to get it just right. At the other end of the spectrum, characters in computer games require intelligent autonomous motion to be generated in real time. Ideally, the digital representations of such autonomous characters should embody the motion skills of the character, as well as the more traditional descriptions of geometry and surface properties.

The formulation of the motion control problem for computer animation is different in some ways than that used in robotics or control. A first key difference is that a motion trajectory having open-loop control is often a satisfactory solution for computer animation. The representation and algorithms for synthesizing open-loop motions are in general quite different from those developed for closed-loop motions. It should be noted that some form of closed-loop solution remains ideal, as it more closely reflects the abilities of humans and animals to take corrective actions during motions. A second difference is that animation solutions need to provide 'handles' for control over the general motion characteristics. Thus, one should not only be able to specify a walk, but also the desired style of walk. This is a particularly challenging problem because of its subjective nature.

Lastly, animators are often interested in manipulating the design of characters. One might choose to increase the weight of the body or the location of hips in order to give a motion a particular desired look. Thus, algorithms which are very specific to a particular model are not as interesting as more general control algorithms. Of course, the relationship between the physical design of an animal and the ways it can move is also of interest strictly as a scientific endaevor.

1.2 *A Desired Solution*

It is useful to have a description of an ideal motion control method in mind, if only to use as a measuring stick for proposed solutions. The system described below gives an example of an ideal motion control system, as seen from the point of view of an animation application.

Consider a motion controller for human walking. Such a controller should ideally be capable of many things. It should be capable of walking at various speeds, turning, and walking up and down stairs. It should be able to walk across variable terrain, cross a river by stepping from rock to rock, and avoid bumping into people in a crowd. It should adapt to walking on soft and hard terrain, slippery and sticky terrain, and walking using different types of footwear. It should also be able to adapt to carrying some extra weight on any part of the body, or receiving a sudden push by a person or a gust of wind. It should be able to walk in various styles, including those reminiscent of Charlie Chaplin. The resulting walks should be indistinguishable from real walks. The controller should be able to reproduce a limping gait, a tiptoe gait, an angry stomping gait, or a walk across hot sand. The gait should adapt appropriately to the character carrying a full glass of water, a large stack of plates, or a squirming baby.

The representation of the controller should allow for multiple levels of detail, to match different levels of detail in the physical simulation. This would be to allow for efficient simulation of walking characters as seen at a distance or in a crowd. The controller should also be able to estimate the risk of falling for a given motion – imagine needing to judge the speed at which to walk down an icy hill.

Given this rather idealistic system description, many choices and restrictive assumptions need to be made in order to design a workable control solution. The following section attempts to define some of the choices. It also attempts to list current successes and failures in human and animal motion control.

2. PERSPECTIVES ON THE PROBLEM

2.1 *Some Fundamental Choices*

As with many problems, the assumptions used in establishing a solution are as important as the solution techinique itself. The following list of questions explores the wide range of choices that need to be made in developing control algorithms for human animal motion.

To what extent can a motion be open-loop?

Human and animal motions are open-loop in nature when examined at a small time scale. At a larger time scale, the closed-loop mechanisms involving spinal-cord reflexes and the cerebellum come into play. The question of how to best partition the control of a motion between open-loop and closed-loop control mechanisms is an intriguing one.

How can sensory information be best exploited?

While a state vector description offers mathemetical convenience, it can in some cases hide alternative ways of exploiting sensory information. Consider a 'touch sensor,' which records the state of contact of a foot with the ground. While such a sensor clearly provides useful information, extracting the equivalent information from a state-vector description would be cumbersome.

What kind of objectives are necessary to achieve natural motions?

It is convenient to assume that natural motions minimize an effort metric. However, defining the right metric and avoiding local minima are problematic.

How can the relative difficulty of motions be categorized?

Most of us would agree that crawling, walking, and hopping backwards on one leg are motions of increasing difficulty. How then should we define a metric for ranking the difficulty of various human and animal motions? The time and space complexity analysis of numerical control algorithms might provide some answers.

How can a priori information about motions be incorporated?

In the case of humans and animals, we have a potential wealth of motion data available to us. How can we best use this data towards constructing control algorithms capable of reproducing similar motions? Sophisticated motion capture systems used in computer animation mean that this question is now feasible to address.

How can techniques from Computer Science be applied to motion control problems?

While the use of controllers for driving simulated humans in computer animations is ostensibly an application of control science to computer science, there are also possibilities for transfer in the other direction. Techniques in numerical representation, optimization, learning, and search can all be brought to bear on complex control problems. We further espouse the merits of numerical approaches in a later section.

How can the state of the environment be incorporated into the control problem?

Environments occur with infinite variety. The control system for any kind of motion must extract the relevant information and represent it with a finite description. A task-dependant representation is probably called for, as one can imagine that the useful environmental characteristics for motions such as walking, skiing, and skating are all different.

What kind of control representation will be used for the solution?

The choice of control representation impacts on all aspects of the motion control. What is the right mix of continuous and discrete control? Which offers the most flexibility in terms of learning and adaptation? Which is easiest to initialize using a priori knowledge of the desired motion? To what extent should the controller maintain internal state?

How can multiple motion skills be integrated together?

The ability to compose simple control components together in order to create a complex controller is of obvious benefit. In the context of computer animation, one can imagine users wanting to develop an interchange format for control skills, such as different walking styles or sports abilities. The ability to import, export, and combine controllers is a key to allowing large number of developers of control algorithms to cooperate and extend each others designs.

What is the interaction of planning and control?

Control systems are typically thought of as operating in an online context, while planning systems are thought of as operating offline. However, actions such as crossing a river by jumping across a sequence of rocks, as shown in Figure 3 require tight integration of planning and control.

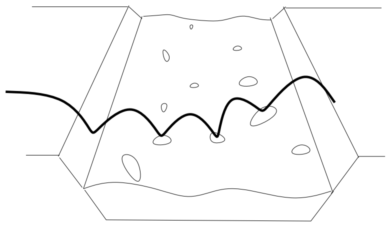


Fig. 3. The crossing-the-river problem.

2.2 *Successes and Failures*

Much progress has been made towards human and animal motion control. A proper review of the previous work cannot be done in the limited space available here. However, it is interesting to try to identify some successes and failures.

Simulation

As a first example of a success, we can consider the development of efficient simulation techniques for complex articulated bodies. Forward and inverse dynamics of $O(n)$ complexity have been explored in depth and can now be seen in commercially-available simulation packages such as SD/FAST(Symbolic-Dynamics, 1990). The ever-increasing speed of microprocessors means that complex articulated figures can now be simulated at interactive rates.

At the same time, there remains a need for a spectrum of dynamical simulation techniques which provide a continuous tradeoff between computation time and simulation accuracy. One might also expect a difference in the cost of simulating predictable motions, such as a walk cycle, and simulating an unexpected fall. Because the walk cycle occurs within the confines of a small subset of the state space, it should be possible to build a more efficient dynamics simulator for such restricted regions of state space.

Motion Capture

In the past ten years, motion capture technology has made great advances. It is now possible to accurately record human or equine motion within a large staging area and at a high frame rate (100Hz). Both optical and magnetic capture technologies have seen significant improvements. A recognizable failure, however, has been a dearth of control algorithms which can effectively apply this motion data towards the synthesis of controllers. While techniques exists for using motion data as controller reference trajectories, it should be possible to extract significantly more useful information from motion data sets.

Optimization

A variety of motion optimization techniques continue to attract a great deal of interest in control, biomechanics, robotics, and animation. However, determining a proper optimization metric which captures the natural qualities of human and animal motions still remains enigmatic and elusive.

Benchmark Problems

The use of benchmark problems such as the cart-and-pole, the acrobot, and others has provided the benefit of being able to compare and contrast the performance of alternative control algorithms. Given the availability of sophisticated simulation tools, however, it would be interesting to establish complex benchmarks involving simulated humans and animals. One can easily imagine a kind of virtual olympics with competitions among various control algorithms under appropriate restrictions of the types of acceptable control inputs.

Analytic vs Numerical Methods

Numerical synthesis methods for control provide powerful tools that complement analytic control solutions in many ways. Much of our work, to be described in the following sections, can be regarded as taking a numerical approach to control synthesis. Controllers are synthesized using numerical optimization procedures, which require repeated simulations to guide the synthesis process. There are several strong arguments to be made for numerical approaches to control synthesis over purely analytic methods. Although analysis by itself leads to important insights(Alexander, 1984), it often does not reveal enough to build a working controller.

Numerical methods are well suited to cope with the complexity of many dynamical systems, such as the articulated figures which are used to model humans and animals. When the generation and use of complex sensory information must also be considered, the use of a simulation-based approach is perhaps the only feasible path for control synthesis.

Designing and testing learning or adapative techniques requires extensive use of simulations. Learning or adaptation is a key part of any realistic model of a motor skill, and learning strategies are difficult to evaluate analytically.

The current availability of sophisticated physical simulation software means that simulation-based approaches are more tractable than ever before. It is relatively easy today to build reasonably accurate physical simulations of humans or animals with given physical parameters. Simulations are also easily adapted to reflect different terrain conditions, as well as to simulate any type of sensory input. Experimental approaches allow for the evaluation of many alternative choices for representing and learning control functions.

2.3 *A Sampling of Results*

We have explored a variety of control techniques which make extensive use of numerical simulation and numerical optimization techniques. The availability of rich, efficient simulation environments provides new opportunities for exploring control techniques and control representations. At the same time, autonomous behaviours for simulated characters can potentially benefit from the rich family of well-studied control techniques.

The following three sections represent different approaches to solving control problems for autonomous simulated creatures. We focus on describing the fundamental ideas behind each approach and refer the reader to the original papers for a full description.

3. SENSOR-ACTUATOR NETWORKS

Sensor-actuator networks (SANs) are a simplistic type of control structure that allows for the 'discovery' of modes of locomotion(van de Panne and Fiume, 1993). The user supplies the configuration of a mechanical system that has been augmented with simple sensors and actuators. A stochastic algorithm then explores various ways in which the sensors and actuators can be wired together in order to produce modes of locomotion.

There are a few noteworthy features of SANs. First, the building blocks of the control system are a rather unlikely collection of parts: binary sensors, PD actuators, and summation nodes with time delays. Surprisingly, these can be assembled together to control unstable, dynamic movements. More surprisingly, the job of building a controller from these parts can be automated. Second, stochastic techniques are used to perform the controller synthesis. This is illustrative of the potential power of numerical optimization techniques applied to control problems. Although the following discussion summarizes some of our own work, others in animation have also pursued similar lines of thought(Auslander *et al.*, 1995; Ngo and Marks, 1993; Sims, 1994; Grzeszczuk and Terzopoulos, 1995).

3.1 *Constructing and simulating a creature*

The types of creatures used for our experiments are mostly 2D articulated figures, built out of rigid links connected with joints. Associated with each joint is an actuator that produces a torque to drive the joint towards a desired reference position using a local PD controller. An example creature is shown in Figure 4, representing a desklamp which becomes 'alive'. The equations

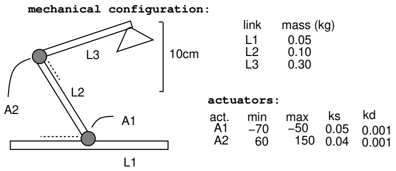


Fig. 4. The Luxo creature.

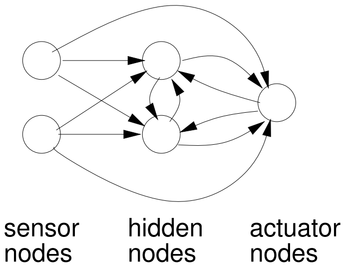


Fig. 5. Topology of a sensor-actuator network.

of motion for the creatures are generated automatically by a *dynamics compiler* of our own construction, although commercial programs such as SD/FAST(Symbolic-Dynamics, 1990) provide even more sophisticated capabilities. Ground reaction forces are generated by a penalty method which includes a friction model.

3.2 *Sensor-actuator networks*

The control structure is a sensor-actuator network: a small non-linear network of weighted connections between a collection of binary sensors and the actuators (the muscles of our creatures). An example topology is shown in Figure 5. The network has internal delays, thereby giving it dynamic properties, most notably the ability to oscillate and for sensors to influence the oscillation pattern. SANs are similar in nature to many artificial neural networks, but are different in several respects. The nodes of SANs have associated time delays which are a key property governing their behaviour. More importantly, our synthesis method does not employ any derivative-based learning methods.

Each node in a SAN computes the weighted sum of its inputs and sets the output to 1 or 'on' after a given time delay if the input sum exceeds a fixed threshold value. The actuators are directly controlled by the output *actuator nodes* in the network. The input sum for these nodes provides a value proportional to the desired reference position for the PD controller driving the associated joint.

Each SAN is defined by a set of parameters which govern its behaviour. These include the weights connecting outputs of nodes to inputs of other nodes, the internal time-delays of nodes, and the parameters defining the mapping which provides

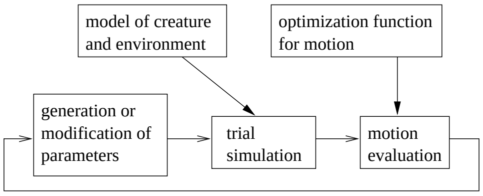


Fig. 6. Generate-and-test control synthesis.

reference positions to the actuator joints. The collection of these parameters defines the search space for the control synthesis technique.

3.3 *Control Synthesis*

We use a *generate-and-test* technique to synthesize appropriate SAN controllers, as described in Figure 7 and depicted in Figure 6. While this is a 'blind' technique in that it has no *a priori* knowledge about the problem, it is capable of first finding and then fine-tuning many suitable modes of locomotion, some of which are quite unexpected. Two phases are involved: a *global* search, followed by a *local* search through the parameter space. The sole purpose of the global search is to quickly determine a small set of promising regions in the search space. While most randomly generated controllers will result in little or no useful motion, 1-5% do in fact produce useful motion. The subsequent local search then refines the parameter values using a greedy local optimization procedure. The most common optimization criterion we have used is the distance travelled as measured during T seconds of simulation.

Interesting results have been achieved for a reasonable variety of creatures. A controller can be automatically generated which makes a simulated fish swim forward and even chase a target when equipped with two binary eyes. A mechanical model of a desk lamp can be taught to move in leaps and bounds. Objects as simple as two or three link chains can move in varied and curious ways.

Needless to say, there are limits to the complexity of motion that can be expected to emerge using SANs and the stochastic optimization procedure. It would be futile, for example, to expect to use our method directly to synthesize a SAN for controlling a human performing a high jump. Another major problem is that the operation of a SAN is not readily understandable. To overcome some of these problems, the next technique applies numerical optimization to a more recognizable type of control system.

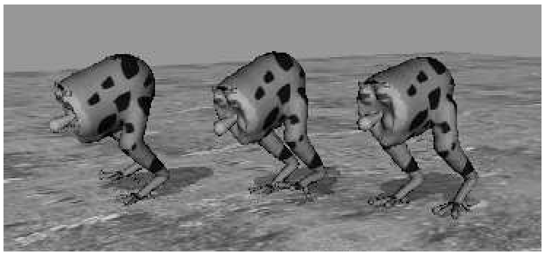


Fig. 8. A simulated windup toy: the monster

4. PARAMETERIZED STATE MACHINES

It is obvious that most forms of locomotion have a periodic nature. In this section we describe a control synthesis technique that exploits this, and which also relies on the surprising passive stability of many motions. In many ways, this work answers the question 'To what extent can we control gaits without the use of sensory feedback?' The results discussed below are collated from (van de Panne, 1996) and (van de Panne *et al.*, 1994).

The best point of departure for explaining the approach is to begin with the motion of a windup toy. A mechanical windup toy goes through a periodic sequence of positions in order to achieve forward motion, driven by a toy motor or spring. There is typically no feedback – the toy is oblivious to its environment. A walking windup toy stays on its feet because of the inherent stability of its motion. Some windup toys are capable of more impressive manoevres, such as back-flips.

Now suppose we imagine providing designers of windup toys with a sophisticated suite of simulation tools, which would let them carefully optimize the design of windup toys. What types of open-loop motions are possible? Surprisingly, some very dynamic motions, such as the simulated bounding of the 'cheetah' creature shown in Figure 2, can be controlled as a *virtual windup toy*. It is also possible to automatically optimize dynamic actions such as the *leaps* of a Luxo lamp (see Figure 12) or to produce a variety of walking and running gaits for creatures like simulated cats and 'monsters' (see Figure 8).

4.1 *Pose-control graphs*

The basis of our simulated wind-up toys is a simple state machine, which we refer to as a *pose control graph* (PCG). An example is shown in Figure 9 for Luxo, the jumping desk lamp. Each state in the PCG has a set of reference positions for the actuated joints of a creature. The poses of Luxo in Figure 9 illustrate the reference (i.e., desired) shape of Luxo in the three states. The state transitions are based upon fixed time intervals or simple sensory information, such as a particular point making or breaking

```
generate and evaluate 200 random controllers
for each of the 10 best
  for 1000 trials
    randomly choose a parameter to vary
    perturb the parameter value by +delta or -delta
    evaluate the new controller by simulation
    if movement has improved then
      keep change
    else
      reject change
  end for
end for
output: 10 controllers, each providing a potentially
        unique mode of locomotion
```

Fig. 7. Pseudocode for generate-and-test optimization

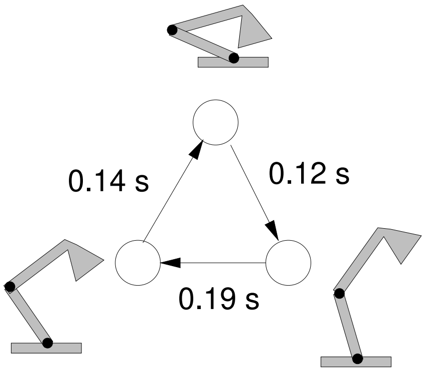


Fig. 9. A pose control graph.

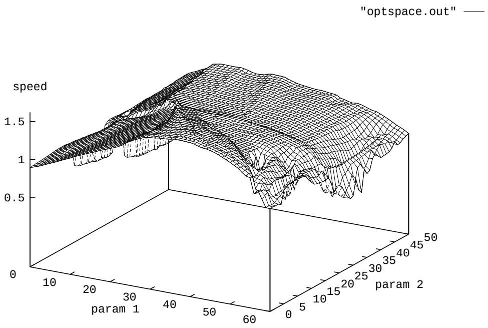


Fig. 10. Optimization space for a cat trot.

ground contact. The PCG itself defines the *desired* internal shape of the creature over time. Its real shape and motion are determined by interactions with the environment and the reaction forces that are produced.

PCGs are defined by a large set of parameters which represent a suitable search space for numerical optimization techniques. These include the joint reference positions in each state (the 'poses') and the state transition times. Any chosen set of parameter values represents a controller whose performance can be evaluated through the use of trial simulations. Speed of locomotion once again represents a straightforward optimization function, which is measured by recording the distance travelled during a fixed simulation time. Other functions may be useful, however, in ensuring that a particular type of gait is produced, or ensuring that a gait remains robust(van de Panne, 1996). Figure 10 shows a visualization of the optimization function for a two-dimensional projection of the parameter space, in this case for a 3D simulated model of a cat attempting to trot. It is clear that any optimization technique must be able to escape from local minima.

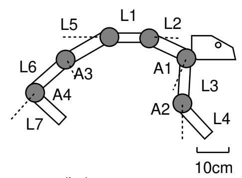
Several varieties of numerical optimization technique seem to succeed at searching for near-optimal parameter values in the search space. We have experimented with versions of simulated annealing and other hill-climbing algorithms. These optimization algorithms each have several parameters of their own which need fine-tuning, but in general tend to perform well. The basic optimization scheme remains effectively that shown in Figure 6.

One of the best results obtained using this type of control synthesis is for the 'cheetah' creature, shown running in Figure 2. This creature has four actuated joints, as well as two passive, springy joints in the back, connected to link L1 (see Figure 11). The numerical optimization yields running gaits which can take full advantage of the passive springs in the back to store and release energy during a fast bounding gait. The resulting motion, which makes use of no feedback from the environment, is surprisingly fluid. Fully 3D motions such as the walking monster shown in Figure 8 can also be controlled using an automatically-synthesized PCG. Both the cheetah and the monster must rely on passive stability, however.

4.2 *Parameterizing motions*

The optimization process described thus far is good at producing a single set of control parameters which results in a stable periodic gait. However, how could we control the speed of a gait?

mechanical configuration:



link	mass (kg)
L1	0.30
L2	0.35
L3	0.15
L4	0.10
L5	0.35
L6	0.15
L7	0.10

actuators:

act.	min	max	ks	kd
A1	−45	45	0.4	0.01
A2	−90	0	0.4	0.01
A3	40	70	0.4	0.01
A4	−120	−30	0.4	0.01

Fig. 11. The cheetah creature.

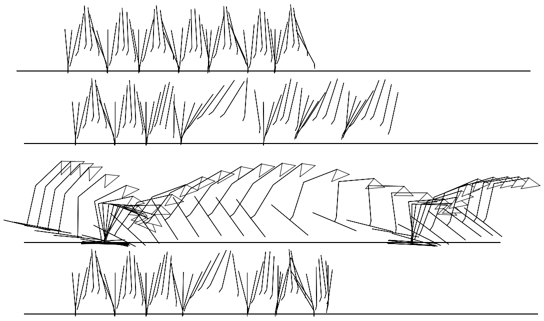


Fig. 12. Optimizing a Luxo leap. In A, B, and D only the middle link is shown for clarity.

One can show that locomotion speed can often be controlled by suitable interpolation between the control parameters for a slow gait and a fast gait(van de Panne, 1996). By optimizing a given controller X with respect to some criterion to yield a new controller Y, the controllers obtained by interpolating between the control parameters of X and Y often result in well-behaved, interpolated motions. This is an immediately useful characteristic for animators, and also shows a promising direction towards building complex, parameterizable motor skills.

A different type of parameterization allows us to also work with aperiodic motions. In Figure 12, the goal is to begin with a periodic hopping motion (A), and to use an optimization procedure to change the middle hop into a large leap (B). This can be accomplished by optimizing for the total distance travelled over all the hops, while restricting the optimization to only alter the parameter values associated with the control of the middle hop. Note that it is important to have each trial simulate the three hops following the leap. This ensures that a large leap followed by a fall will always be a suboptimal solution that is rejected. Figure 12(C) illustrates a detail of the final leaping motion, with horizontal displacement being scaled for clarity. Figure 12(D) shows the result of interpolating between the control parameters for a regular hop (A) and those of the leap in (B). As one might hope, the result is an intermediate size hop.

5. LIMIT CYCLE CONTROL

Simulating human walking has proved to be a surprisingly difficult task, and has long been a subject of fascination(Furusho and Maubuchi, 1987; Furusho and Sano, 1990; Katoh and Mori, 1984; Hmam and Lawrence, 1992; McGeer, 1990*a*; McGeer, 1990*b*; Saito *et al.*, 1994; Stewart and Cremer, 1992; Vukobratovic, 1990; Miura and Shimoyama, 1984). The difficulty arises from its instability, combined with the non-linear and underactuated nature of the dynamics. As a result, a large variety of simplifications are often introduced to make it a tractable control problem. These include planar models with planar dynamics, the use of reference trajectories and linearization, and adding unrealistic constraints which keep the feet glued in place during a stance phase. While these assumptions have been useful for understanding the nature of walking control, they are all unrealistic in some aspect.

Progress in the area of controlling hopping and running gaits has been better. The plenary work of Raibert(Raibert, 1986) and Raibert and Hodgins(Raibert and Hodgins, 1991) in controlling hopping robots (both real and simulated), has shown that hopping can be decomposed into three separable control problems, namely those of (1) controlling hop height, (2) controlling forward speed, and (3) controlling body attitude. It has also been shown that a full dynamical simulation of a human can be made to run using this type of control model(Hodgins, 1994).

The work we briefly summarize here is that of a control method for a complete 3D dynamical simulation of human walking(Laszlo, 1996; Laszlo *et al.*, 1996). The human model has realistic proportions, masses, and moments of inertia. The articulated skeleton used for our simulations is shown in Figure 13. The speed, direction, and style of the walk can be controlled. The walks are unfortunately not mistakeable for a real human walk because they still have certain robotic qualities to their motion. Nevertheless, we believe that the parameters can be appropriately tuned to achieve a class of simulated, 'natural-looking' walks.

Our approach to controlling a walking gait begins with a cyclic state machine (pose control graph), much like the virtual windup toys. This provides for the basic open-loop stepping actions for a walking motion. An example is shown in Figure 14. The walking motion resulting from this open-loop control has the human figure taking several steps and then falling over. Note that the use of *open loop* is somewhat of a misnomer because PD controllers are in fact being used for individual joints. Designing the control consists of

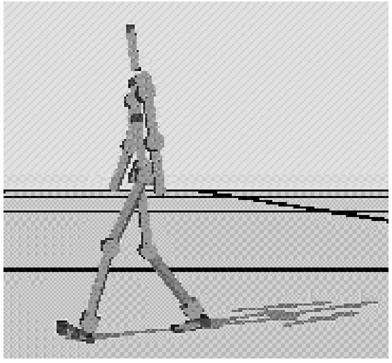


Fig. 13. The human model.

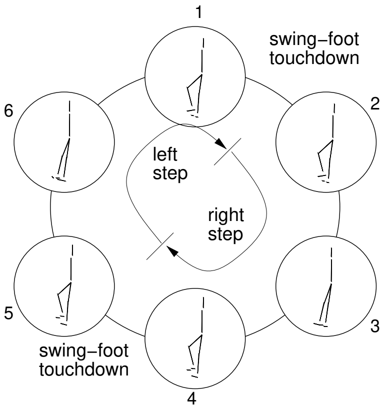


Fig. 14. Finite state machine for walking.

supplying suitable reference positions to the PD controllers used at each joint.

5.1 *Linearizing over a limit cycle*

The limit cycle method takes a basic open-loop motion and adds the necessary control to produce a stable limit cycle. There exist several possibilities for introducing control into this type of cyclical motion. The key insight of our method is the realization that while the system dynamics are non-linear and potentially discontinuous at many points in time, they are quite well behaved over a complete step. Thus, we discretize the limit cycles as shown in Figure 15.

Given the discrete dynamical system, it is possible to define a new set of input and output variables for this system. The output variables we desire to control are a projection of the full system state. In fact, it is sufficient to provide some type of control over forward-backward pitch and left-right pitch. These types of pitch can be measured using a *pitch-vector* defined in terms of the body coordinates. Several possibilites can be shown to work, as illustrated in Figure 16. We refer to the outputs as *regulation variables*, which in this case correspond to the sagittal and lateral components of the pitch-vector. The motivation of using a smaller, projected version of the state

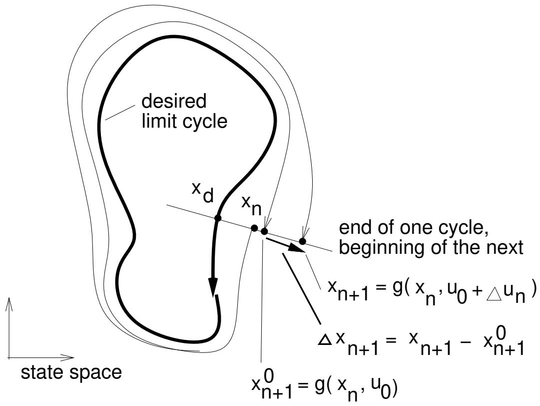


Fig. 15. A limit cycle as a discrete dynamical system.

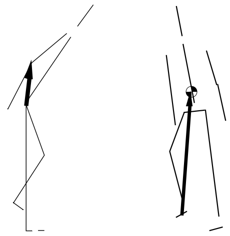


Fig. 16. Regulation variables for use in walking.

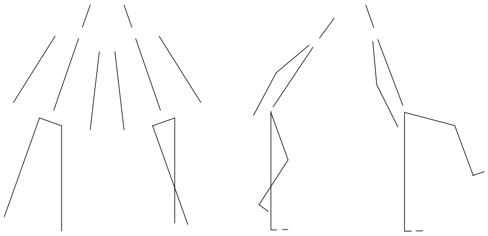


Fig. 17. Control perturbations used for walking.

is one of providing a minimal relevant description of the system dynamics. The reasoning behind the choice of a pitch-vector is intuitive: a walking figure can fall forwards, backwards, to the left, or to the right. In practice, controlling the pitch vector is quite sufficient to yield stable walking motions while still leaving the motion largely unconstrained.

Given the choice of a pair of regulation variables (front-back pitch and lateral pitch), we need to select an appropriate pair of control perturbations to yield a fully controllable system. One set of choices which works well is shown in Figure 17. The control perturbations are applied by altering the references poses in the state machine. The magnitudes of the two potential applied alterations for each step represent the control inputs.

The relationship between the regulation variables and control variables is approximately linear for any given step. However, the nature of the linear relationship may change from step to step — i.e., it is a function of the system state. The control algorithm constructs the appropriate linear model

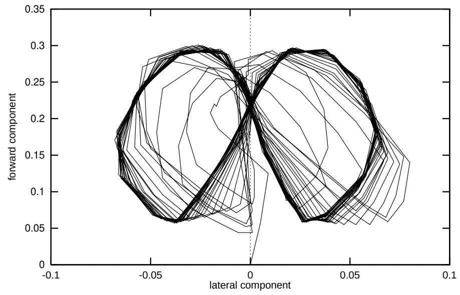


Fig. 18. Regulation variable limit cycle.

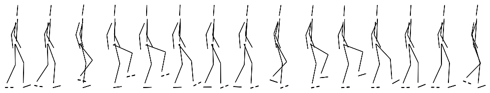


Fig. 19. Stylistic variation on a walk.

using a total of four simulations of the same step, each executed with different control parameters. Finite differences are then used to construct the linear model.

The resulting limit cycle which is traced by the projection of the pitch vector is shown in Figure 18. The left and right lobes of the figure-8 pattern correspond to the pitch vector swaying slightly to the left and the right during the walking motion. At the end of each step, the pitch vector passes through a 'target' point, which represents the reference input for the discrete control system.

5.2 *Results*

The limit cycle control technique enables the control of a full dynamic simulation of a 3D human model having realistic proportions and mass distribution. The figure can walk at a selected speed, as well following a given path. The control has also been applied to obtain walking motions for a 'scout' robot shown in Figure 1. Variations on a walk are easily obtained by changing the open-loop control. The closed-loop component of the control, as implemented by the limit cycle control method, automatically adapts. An example is shown in Figure 19.

6. CONCLUSIONS

The availability of sophisticated simulation packages is today making automated numerical synthesis techniques an increasingly flexible and useful approach. Numerical optimization techniques can work well in the face of complex, non-linear phenomena that might otherwise be difficult to cope with analytically.

Simulating human and animal motion is a challenging task from which much can be learned. How can we develop the complete sets of motor

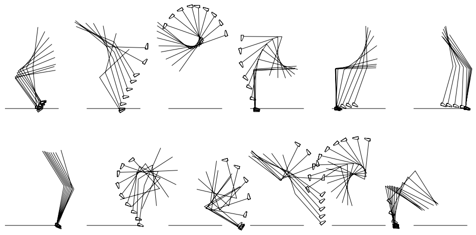


Fig. 20. A Flipping Acrobot.

skills which a squirrel uses when jumping between branches in a tree, or that a gazelle uses in turning a sharp corner? It is exciting to think of the possible control techniques which might be useful in solving these kinds of problems. Simulation-based approaches may prove to be a significant investigative tool of the future.

Animating human or animal movements naturally leads into an exploration of many facets of movement planning and execution. One which we have been exploring deals with a highly-unstable, underactuated 'acrobot'(Berkemeier and Fearing, 1994). This creature is in effect a two-link robot, which must apply continuous active control to balance itself. With some effort, control strategies for forward locomotion and even front-and-back flips can be developed, as shown in Figure 20.

Lastly, many motions require careful planning to succeed. It is not clear where a control problem ends and a planning problem begins. Figure 21 is an example of this, where Luxo, the hopping lamp, must locomote across a variety of terrain. Initial explorations in this direction using a simulation-based synthesis approach have yielded encouraging results, although there is clearly very much exciting work left.

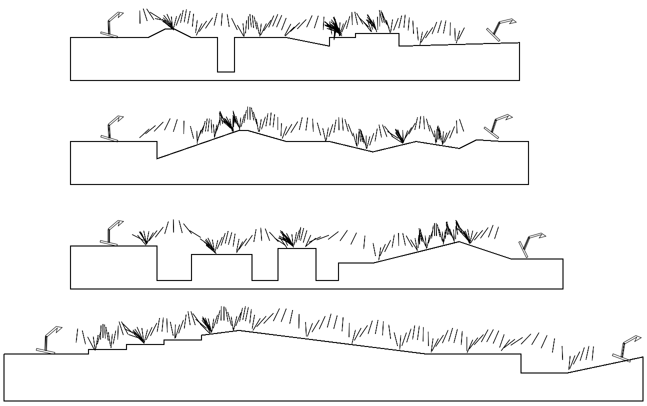


Fig. 21. Synthesized control for Luxo on cross-country runs.

7. REFERENCES

Alexander, R. M. (1984). The gaits of bipedal and quadrupedal animals. *International Journal of Robotics Research.*

Auslander, J., A. Fukunaga, H. Partovi, J. Christensen, L. Hsu, P. Reiss, A. Shuman, J. Marks and T. Ngo (1995). Further experience with controller-based automatic motion synthesis for articulated figures. *ACM Transactions on Graphics.*

Berkemeier, M. D. and R. S. Fearing (1994). Control experiments on an underactuated robot with applications to legged locomotion. *Proceedings, IEEE International Conference on Robotics and Automation* pp. 149–154.

Bruderlin, A. and L. Williams (1995). Motion signal processing. *Proceedings of Siggaph '95, ACM Computer Graphics* pp. 97–104.

Furusho, J. and A. Sano (1990). Sensor-based control of a nine-link robot. *The International Journal of Robotics Research* **9**(2), 83–98.

Furusho, J. and M. Maubuchi (1987). A theoretically motivated reduced order model for the control of dynamic biped locomotion. *Journal of Dynamic Systems, Measurement, and Control* **109**, 155–163.

Grzeszczuk, R. and D. Terzopoulos (1995). Automated learning of muscle-actuated locomotion through control abstraction. *Proceedings of SIGGRAPH '95, ACM Computer Graphics* pp. 63–70.

Hmam, H. M. and D. A. Lawrence (1992). Robustness analysis of nonlinear biped control laws via singular perturbation theory. *Proceedings of the 31st IEEE Conference on Decision and Control* pp. 2656–2661.

Hodgins, J. K. (1994). Simulation of human running. *Proceedings, IEEE International Conference on Robotics and Automation* pp. 1320–1325.

Katoh, R. and M. Mori (1984). Control method of biped locomotion giving asymptotic stability of trajectory. *Automatica* **20**, 405–414.

Laszlo, J. (1996). *Controlling Bipedal Locomotion for Computer Animation.* M.Sc. Thesis, Department of Computer Science, University of Toronto.

Laszlo, J., M. van de Panne and E. Fiume (1996). Limit cycle control and its application to the animation of balancing and walking. *Proceedings of SIGGRAPH '96* pp. 155–162.

McGeer, T. (1990*a*). Passive dynamic walking. *The International Journal of Robotics Research* **9**(2), 62–82.

McGeer, T. (1990*b*). Passive walking with knees. *Proceedings of IEEE International Conference on Robotics and Automation* pp. 1640–1645.

Miura, H. and I. Shimoyama (1984). Dynamic walk of a biped. *International Journal of Robotics Research* pp. 60–74.

Ngo, J. T. and J. Marks (1993). Spacetime constraints revisited. *Proceedings of SIGGRAPH '93* pp. 343–350.

Raibert, M. H. (1986). *Legged Robots that Balance.* MIT Press.

Raibert, M. H. and J. K. Hodgins (1991). Animation of dynamic legged locomotion. *Proceedings of SIGGRAPH '91* pp. 349–358.

Saito, F., T. Fukuda and F. Arai (1994). Swing and locomotion control for a two-link brachiation robot. *IEEE Control Systems* **14**(1), 5–12.

Sims, K. (1994). Evolving virtual creatures. *Proceedings of SIGGRAPH '94, ACM Computer Graphics* pp. 15–22.

Stewart, A. J. and J. F. Cremer (1992). Beyond keyframing: An algorithmic approach to animation. *Proceedings of Graphics Interface '92* pp. 273–281.

Symbolic-Dynamics (1990). *SD/Fast User's Manual.*

Unuma, M., K. Anjyo and R. Takeuchi (1995). Fourier principles for emotion-based human figure animation. *Proceedings of SIGGRAPH '95, ACM Computer Graphics* pp. 91–96.

van de Panne, M. (1996). Parameterized gait synthesis. *IEEE Computer Graphics and Applications* pp. 40–48.

van de Panne, M. and E. Fiume (1993). Sensor-actuator networks. *Proceedings of SIGGRAPH '93* pp. 335–342.

van de Panne, M., R. Kim and E. Fiume (1994). Virtual wind-up toys for animation. *Proceedings of Graphics Interface '94* pp. 208–215.

Vukobratovic, M. (1990). *Biped Locomotion: Dynamics, Stability, Control and Applications.* Springer Verlag.

Witkin, A. and Z. Popovi'c (1995). Motion warping. *Proceedings of SIGGRAPH '95* pp. 105–107.