Self–Stabilizing Running

by

Robert P. Ringrose

B.A. Physics and Computer Science,
Cornell University, Ithaca, NY (1990)
S.M. Computer Science,
Massachusetts Institute of Technology, Cambridge, MA (1993)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 1997

Author . . . . [illegible]
Department of Electrical Engineering and Computer Science
October 8, 1996

Certified by . . . . [illegible]
Marc Raibert
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by . . . . [illegible]
Arthur C. Smith
Chairman, Departmental Committee on Graduate Students

Self–Stabilizing Running

by

Robert P. Ringrose

Submitted to the Department of Electrical Engineering and Computer Science
on October 8, 1996, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science

Abstract

Legged robots can sustain stable dynamic locomotion without sensors or feedback. It is possible to construct a running robot that is inherently stable and needs no sensing to reject minor perturbations; the robot is therefore self-stabilizing. In contrast, most previous attempts to make robots run have used active, high bandwidth feedback control systems. I mathematically analyze the behavior of a simple self-stabilizing monopod, determine a running pattern, and generalize the motion to two and four legs. Using physics-based simulations, I demonstrate the stability of these motions as applied to robots. The quadruped can stably trot, pace, bound, and gallop. I verify the simulations' behavior with a physical monopod robot. Low bandwidth control systems layered on top of the self-stabilizing running motion could allow the robot to respond to the operator's desires or changes in the environment.

Thesis Supervisor: Marc Raibert
Title: Professor of Electrical Engineering and Computer Science

Contents

Chapter 1

Introduction

Running motions can be *self-stabilizing.* That is, with proper design the structure and motion of a robot can automatically cause it to recover from minor disturbances even if it cannot detect them. Similarly, many items in common use, such as tables, automobiles, and airplanes are stable about their standard operating conditions. Tables remain standing, automobiles drive forward, and airplanes fly straight and level (Pan 1972). This reliable default behavior makes them easier to control. In contrast, most current legged robots rely on high bandwidth sensing and feedback for stability. Self-stabilizing running could eliminate most of the bandwidth normally required to control running. I demonstrate some of the possibilities of self-stabilizing running by using it to control simulated one, two, and four legged robots in a variety of gaits, including a gallop.

Controlling most legged robots is like trying to balance a ball on top of a mountain: you must continuously check which way the ball is rolling and push it back to the top. Self-stabilizing running is like trying to balance a ball at the bottom of a valley. Unless the ball is completely out of the valley it will roll back to the bottom, where you want it. There is no need to actively sense which way it is rolling or push it back to the desired position.

As a demonstration of self-stabilizing running, I have created simulations of a monopod, a biped, and a quadruped (Figure 1-1). The quadruped is a pair of bipeds connected by a back. Each biped is a pair of connected monopods. By building the monopods so that they

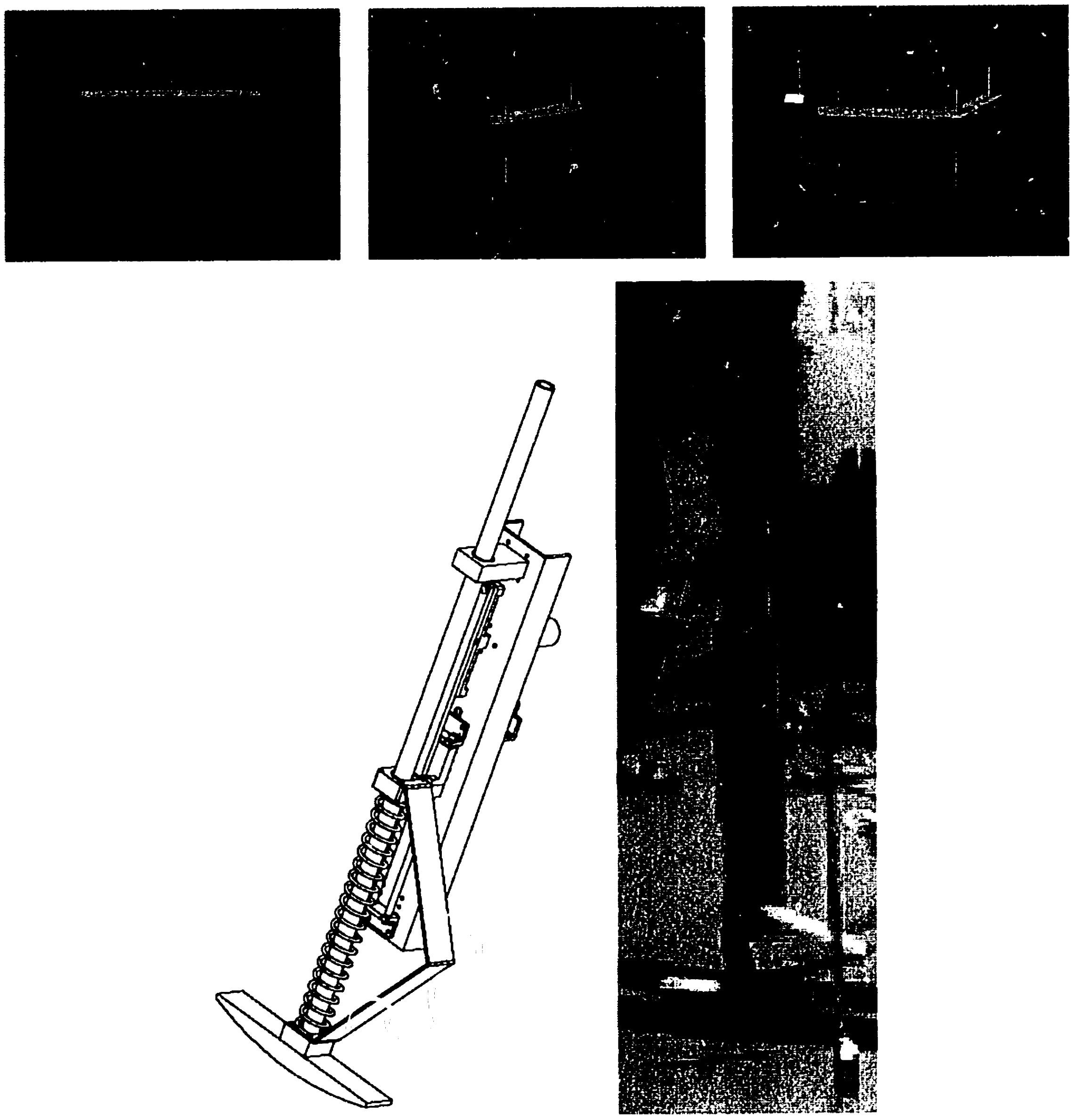

Figure 1-1: Self-stabilizing robots. Top row, left to right: monopod, biped, quadruped. Bottom row: diagram of physical monopod, picture of physical monopod in the laboratory.

are self-stabilizing and correcting for the interactions between the monopods, the bipeds run stably. Similarly, by correcting for the interactions between the two bipeds the resulting quadruped runs stably without any sensors or feedback. Each joint has an actuator in series with a spring. Over the running cycle, these actuators go through a predetermined pattern. There are no sensors, there is no conditional logic, and the actuator pattern does not change. With properly chosen actuator patterns and physical structure, the simulated robots run stably as a consequence of their design and the laws of physics.

Several indications exist, in nature and in the laboratory, that self-stabilizing running should be possible. Thompson and Raibert, limiting themselves to springs and a mechanical system, found reentrant running motions in a simulated hopping machine (Thompson & Raibert 1989). McGeer's demonstrates a stable system walking downhill under the influence of gravity (McGeer 1989) (McGeer 1990). McGeer's passive dynamic robots are pairs of mechanical legs which walk downhill stably without any motors, sensors, or computers. Also, a recent pattern-based cockroach simulation generated stable dynamic running without feedback. It is actuated in the same manner as the self-stabilizing running simulations I use, and runs stably. However, its stability is also due to the fact that a cockroach's long legs give it an extremely wide base of support. Finally, animals run easily. Some animals can run within minutes of birth. Chickens can run with their heads cut off. It makes sense that their control should be simple and stable. These examples suggest that running should be achievable with low bandwidth and minimal tuning, as with self-stabilizing running.

Self-stabilizing running has several benefits.

- It does not rely on sensors. There will not be errors from a foot sensor falsely registering touchdown, or from sensor noise.
- Aside from the initial configuration of the robot, there is no stored state to update, have corrupted by erroneous code, or reinitialize if you restart the robot.
- The actuator positions are calculated locally, based solely on the current time. There is no need for a large number of control signals from a central "brain" to the actuators.

- Should an external force disturb the robot, it will recover naturally. There is no need to constantly check for disturbances, figure out how to counteract them, and avoid overcorrecting.

- One can expand the capabilities of a self-stabilizing robot by layering more complex behaviors on top of the self-stabilizing motions. For example, with the addition of a terrain sensor, a higher-level control could appropriately retract or extend the legs to counteract rough terrain.

- For the self-stabilizing quadruped I have created, the phases between its legs are arbitrary. It is capable of gaits, including transverse and rotary gallops, which have not been demonstrated by other controllers, such as the virtual leg controller (Raibert, Chepponis, & Brown 1986).

On the other hand, there are some significant drawbacks to self-stabilizing running.

- There is no mechanism for input. An operator cannot command speed or direction. Self-stabilizing runners cannot react actively to the environment; they simply keep going and hope that the disturbances from it are relatively small.

- The basin of attraction is smaller for a self-stabilizing robot than for a computer-controlled one. It cannot deviate as far from a stable running motion and still return to it.

- It is conceivable that the running motion could be stable, but not allow the robot to start. In this case, the motions required to keep running will not make it start running. It is a result of the size of the basin of attraction, which may not include the robot at rest.

- Self-stabilizing robots could be used as a base upon which one would layer lower-frequency feedback controllers which would take care of more complicated tasks. However, there is no guarantee that the feedback controllers will be able to do their jobs without disturbing the robot's motion far enough that it leaves its basin of attraction and falls.

- It is not necessarily easy to find a self-stabilizing motion for a given robot. The robots I use happen to be constructed in such a way that they break down into smaller, stable pieces, but that is not the case for all robots.

- Given a supposedly self-stabilizing running motion, it is difficult to prove stability. It is also hard to measure how much its motion is disturbed by any given external influence, or how much it could be disturbed without falling over.

It is possible to address most of the drawbacks associated with self-stabilizing running. Low bandwidth feedback control can allow an operator to modify the self-stabilizing motions to change the behavior (for example, altering the speed), and take care of large deviations from which the self-stabilizing robot might not otherwise recover. The only way I know to determine if the low bandwidth feedback will disrupt the self-stabilizing motions is to try them. Previous dynamic legged robots, such as those presented in (Raibert *et al.* 1992) (Ringrose 1992b) (Murthy & Raibert 1983), did not take advantage of self-stabilizing running because the conditions required for stable running were not known.

I have created simulations of one, two, and four legged robots. In accordance with the principles of self-stabilizing running, their control systems have no sensors, just a timer and a set of carefully chosen actuator paths. I attempt a mathematical solution for the robot's behavior and stability where possible. When mathematical solutions are impractical, I use simulations of the machine and make approximations as necessary to demonstrate stability. The quadruped can trot, pace, bound, and gallop. As a test of the simulations' validity, I had a mechanical self-stabilizing monopod built (Appendix A). The behavior of the mechanical monopod is very similar to that of the equivalent simulated monopod. It recovers well from disturbances.

The next chapter discusses several types of legged locomotion and how they relate to self-stabilizing locomotion. Chapter three uses a simplified model to develop an intuitive understanding of why the monopod, from which I create the biped and quadruped self-stabilizing runners, hops stably. Chapter four presents several tools which I use to find self-stabilizing motions and demonstrate their stability. The fifth chapter describes the self-stabilizing mechanisms for the monpod, biped, and quadruped. Chapters six through eight present results for simulations of monopod, biped, and quadruped robots. Finally, there is a brief summary and discussion of future work. The two appendices contain a description of a physical self-stabilizing monopod robot and digitized frames from the videotape which should accompany this work. The videotape itself has the physical robot in use and computer generated animations of the various self-stabilizing simulations.

Chapter 2

Legged Locomotion

There are several different types of legged locomotion, differentiated by how the robot are stabilized (Figure 2-1). Statically stable locomotion refers to robots which move slowly and have a broad base of support, so they can essentially ignore momentum in their motion. Dynamic robots, on the other hand, have significant momentum and a small base of support. If their joints were to lock in place they would tip over. Most dynamic robots are actively stabilized by a computer which calculates how they should move. Those dynamic robots which do not use a computer I divide into passively stable and self-stabilizing robots. Passively stable robots are typically powered by gravity. The focus of my research is on self-stabilizing robots, which have powered actuators and high momentum, but no computers or other feedback control mechanisms. Finally, robot running has some similarity to juggling, although when juggling the actuator is on the "ground" rather than part of the robot.

2.1 Statically Stable Locomotion

A legged robot has a polygon of support, defined by taking the convex hull of the feet touching the ground. The robot is statically stable if its center of mass remains above the polygon of support (Figure 2-2). Statically stable robots are only stable at low velocities; if

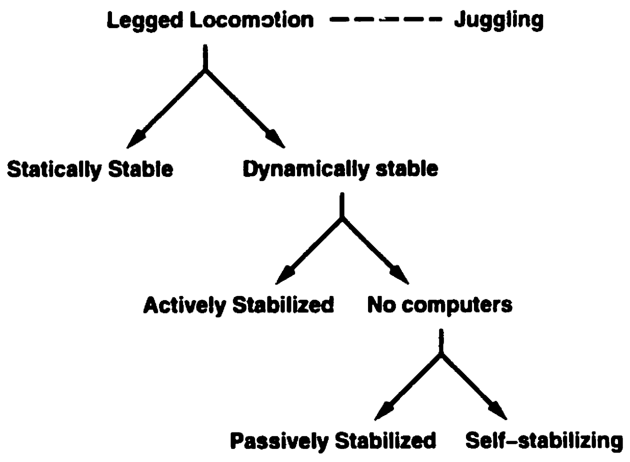

Figure 2-1: Types of legged locomotion. Statically stable robots move slowly enough that momentum is not a problem. Dynamically stable robots are either actively stabilized by a computer, passively stabilized by physics, or self-stabilizing. Juggling is similar to legged locomotion, except the actuator is attached to the ground in juggling, while it is part of the leg for locomotion.

the robot has sufficient forward velocity and the center of mass is near the forward edge of the polygon of support, the robot may tip over even though it is statically stable (Figure 2-3). As a result, statically stable walkers have fairly low maximum speed. Some of the principles of balance, applied foot force, and terrain usage carry over to dynamically stable robots.

The polygon of support is defined by the touchdown points for the feet. For the polygon to have nonzero area, there must be at least three feet on the ground at all times, including when a foot is being repositioned for the next step. Therefore, a statically stable walking robot must have at least four legs.[1] Many walking robots have six legs because the additional legs allow more gaits, and having more legs on the ground can increase the size of the polygon of support. Additional legs also lead to greater fault-tolerance when walking outdoors, where legs break or feet slip.

When a legged robot has multiple feet on the ground, the legs may waste energy working

[1]Large feet, rather than three legs on the ground, could also stabilize the robot.

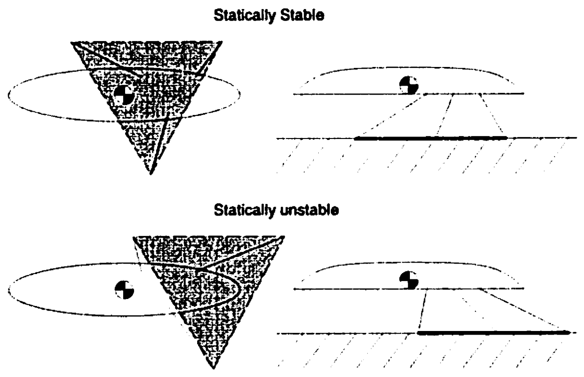

Figure 2-2: Illustration of a three-legged machine in and out of its polygon of support. The top figure demonstrates statically stable support. The bottom figure demonstrates statically unstable support.

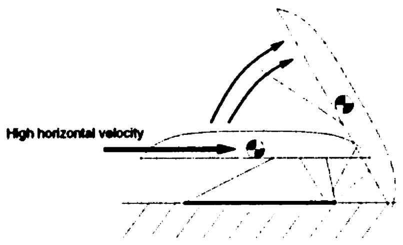

Figure 2-3: Momentum changes the stable positions of the center of mass. With enough momentum, a machine is able to tip over and may be unstable, even though it is statically stable.

against each other. The front leg could push back, while the rear leg pushes forward, and because the ground is generally inflexible the result is wasted effort and mechanical stress. In a less extreme example, if one is attempting to rotate the robot body and does not carefully calculate the forces applied at the feet, different feet are likely to work against each other. Song and Waldron addressed this problem while working on the Adaptive Suspension Vehicle by mathematically calculating the forces which need to be exerted at the feet in order to have the desired resultant force and torque applied to the body (Song & Waldron 1989).

Additionally, in rough terrain one must carefully chose where the foot will fall next. Many people have addressed this question. Some of the "pitfalls" include deciding what footholds are stable, recovering from slipping feet, and choosing footholds such that all the legs will have places to put their feet as the robot moves forward. Many researchers simply ignore these problems, assuming the ground is stable, flat, or otherwise well-behaved. Other possibilities for finding footholds include laser range finders, human assistance, video analysis, and "feeling" the ground with the feet to find appropriate footholds. The Ambler deals with slipping feet by using a conservative support polygon (Krotkov & Simmons 1996). If a robot has m legs on the ground, the conservative support polygon is the intersection of the $m-1$ support polygons created by removing any one leg. As long as the robot keeps its center of mass over the conservative polygon of support, any individual leg could slip or fail without the entire robot tipping. Finally, the positioning of the various feet given a set of valid and invalid terrain positions is a computational problem which has been addressed by many researchers. The problems with getting a map of valid and invalid terrain positions, as well as the relatively large computation required, has kept most work with extremely rugged terrain in simulation.

Many researchers have built statically stable walking robots. A few examples are:

- Robert McGhee built a hexapod at Ohio State University (McGhee 1983). It walked with various gaits over level ground, negotiated obstacles and rough terrain (with some human assistance choosing footholds). The basic control commanded foot velocities, rather than positions.

- Gurfinkel et al. have a superficially similar hexapod, built at about the same time as McGhee's robot (Gurfinkel *et al.* 1981). Its controller was a hybrid, partly digital and partly analog. The low level analog control commanded foot positions, unlike McGee's hexapod, but the results were similar.

- Hirose designed a pantograph mechanism which decouples the horizontal and vertical motions of the foot, allowing easy, efficient leg control. The Perambulating Vehicle II (Hirose 1984)(Hirose & Umetani 1980) was a quadruped with four of these pantograph legs. Using a set of reflexes, it walked over rough terrain, and achieved speeds of 2 cm/sec on flat terrain.

- Song and Waldron at Ohio State University have built the Adaptive Suspension Vehicle (Song & Waldron 1989), a 17-foot long self-contained hexapod capable of carrying a 500 pound payload at 5 mph. It traversed rough terrain, with a human operator on board.

- The Ambler, built at Carnegie-Mellon University, was another self-contained hexapod (Krotkov & Simmons 1996). An interesting feature was that the legs have identical work spaces, allowing it to use a circulating gait. That is, it could consistently pick up whichever leg happened to be in the rear and place it in front. The Ambler was designed to operate completely autonomously in situations where humans would be unable to assist. Therefore, the design sacrificed a lot of potential mobility for fault-tolerance, stability, and power savings.

2.2 Actively Stabilized Locomotion

When animals, people, or robots run, they bounce on their legs as if they were springs. A running human will land on one leg, store part of his kinetic energy in the leg's tendons and muscles, and then recover it as he takes off again. Normally, the pitch of this bouncing

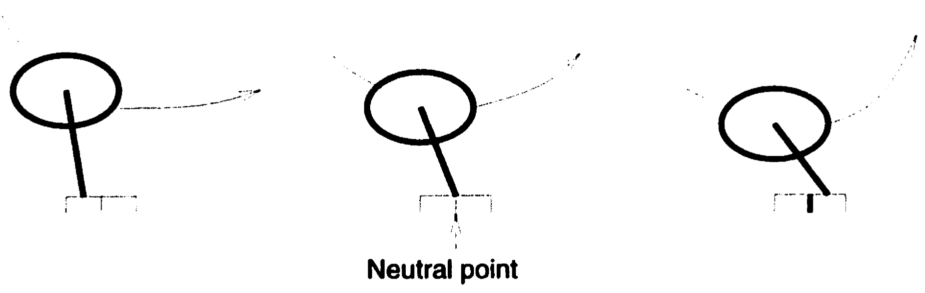

Figure 2-4: Active stability: when the foot is positioned at the neutral point, the body travels along a symmetric path that leaves the robot unaccelerated in the forward direction. Displacement of the foot from the neutral point accelerates the body by skewing the symmetry of the body's trajectory. When the foot is placed closer to the hip than the neutral point, the body accelerates forward during stance and the forward speed at liftoff is higher than the forward speed at touchdown (left). When the foot is placed further from the hip than the neutral point, the body decelerates during stance and the forward speed at liftoff is slower than the forward speed at touchdown (right) (Figure and caption from (Raibert & Hodgins 1991), adapted from (Raibert 1986)). [3]

motion is not stable. A robot bouncing on one leg will gradually tilt, and eventually tip over.

Actively stabilized robots use a control system to counteract the instability present in most rapid running motions. One method of actively stabilizing a running robot is to consider its pitch, velocity, and hopping height separately. Hip torques applied while the foot is on the ground control the body pitch. Foot placement at touchdown controls the velocity (Figure 2-4). Finally, the leg consists of a spring in series with an actuator, and extending or retracting the actuator during stance alters the hopping height as the robot bounces on the spring (Murthy & Raibert 1983).

Koditschek and Bühler, taking a simplified monopod and mathematically analyzing it, show that period 1 and period 2 gaits exist (Koditschek & Buhler 1991). Burdick, Vakakis,

[3]Permission to copy without fee all or part of this figure and caption is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.

and Caughey extend Koditschek and Bühler's model, relaxing some assumptions (Burdick, Vakakis, & Caughey 1989). They discover that given the proper parameters the monopod will exhibit hopping motion of any desired period, even or odd, or chaotic motion. Their analysis relies upon the fact that the monopod, confined to vertical motion, has essentially one degree of freedom. The maximum height of any given hop completely determines the rest of the hop, including the time at which the leg actuator extends.

A hopping strategy for a monopod extends fairly readily to a biped. If the biped's legs are attached close together, use the monopod hopping strategy on the stance foot while keeping the other leg out of the way. During flight, switch feet so that the old stance foot is out of the way, while the monopod hopping control is used on the new stance foot. An alternating biped has several advantages over a monopod. The leg which is staying out of the way can position itself to be in the right place for the next hop, allowing faster running. Prepositioning the unused leg also results in the two legs moving in opposite directions, reducing the pitch disturbances caused by the leg inertia. On the other hand, if the two legs are in phase one can control them using a virtual leg strategy (Raibert, Chepponis, & Brown 1986). After both feet touch down, control the legs to emulate a single virtual leg placed between them, and use the monopod hopping strategy on the single virtual leg.

A quadrupedal running strategy emerges naturally from bipedal running. Split the quadruped's legs into two pairs. Control the pairs as you would a biped in phase, by having them emulate virtual legs in the center of each pair. Having reduced the quadruped to a pair of virtual legs, control them as you would a biped out of phase. Use the monopod running algorithm on whichever virtual leg is on the ground, and keep the other virtual leg out of the way. So, the overall monopod running algorithm controls the two virtual legs, each of which controls two physical legs, to make the quadruped run as directed by an operator.

Several aspects of my self-stabilizing monopod are inspired by Raibert's actively stabilized monopod. Both machines have a similar telescoping leg, with a spring and actuator. Raibert's monopod has an air spring in series with a hydraulic actuator, rather than a linear spring in series with a "perfect" actuator, but the principle of bouncing on a spring in the leg is the same. Also, Raibert's monopod controls stability by altering the touchdown position of the foot. The shape of my self-stabilizing monopod's foot effectively makes the touchdown position of the foot a function of pitch, stabilizing the monopod in pitch.

It is worth noting that, by bending the knees, it is possible to run without having a ballistic phase. This type of running is known as groucho running (McMahon, Valiant, & Frederick 1987). Active control systems can stabilize groucho runners in a manner similar to normal running.

2.3 Passively Stabilized Locomotion

By properly designing the robot, is it possible to passively locomote by a simple interaction of gravity and inertia, with no actuators, no sensing, and no computers. (McGeer 1989) describes a simulation of a passive dynamic running machine; (McGeer 1990) also describes a physical passive dynamic walking machine. I illustrate both machines in Figure 2-5. In some respects, McGeer's work distills walking to its essence: a simple set of motions generated by a simple mechanism.

McGeer's passive dynamic walking machine walks downhill using an inverted pendulum gait, illustrated in figure 2-6. The walking machine is two legs connected by a pin joint (the hip), confined to a plane. It can tip forward and pitch, but it cannot turn or tip to either side. Standing upright, it topples forward. On a shallow downhill slope, given appropriate initial conditions, it will topple from one leg onto the next. The momentum will push it up onto the new leg while the leg previously used swings forward and into place for the next toppling motion. The gradual descent down the slope replenishes the energy lost to damping. With the correct leg lengths and mass properties this motion is stable. If the masses and leg lengths are chosen properly, unactuated knees can serve to raise the foot and prevent the robot from stubbing the swing leg's toe, or motors can lift the feet.

McGeer also demonstrates the stability of this walking motion, using a simulation of his walking machine. First he finds a reentrant walking cycle. Next, he perturbs the walking

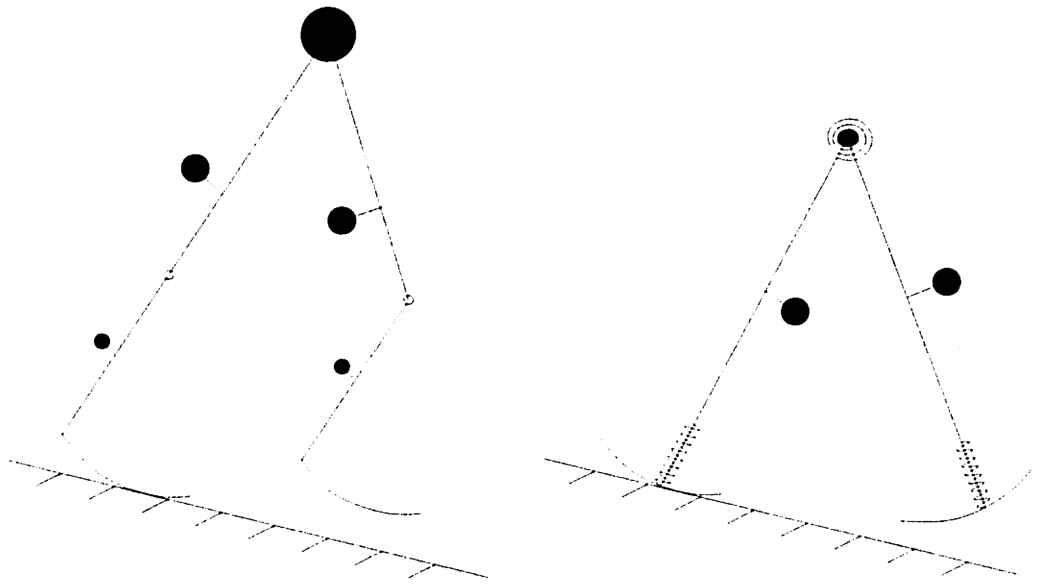

Figure 2-5: Passively stabilized walking (left) and running (right) machines.

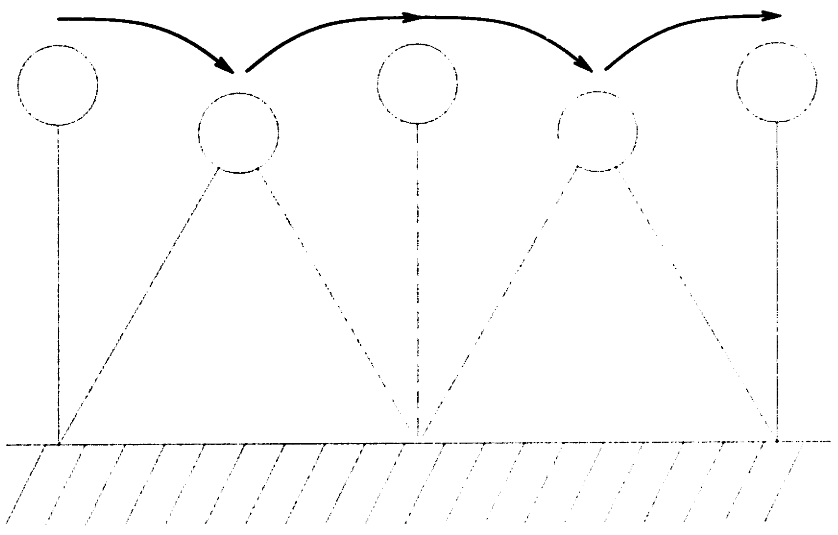

Figure 2-6: Inverted pendulum walking. The robot is initially moving horizontally. It falls forward, catches itself on the next leg, and uses its momentum to rise up on that leg. The process repeats.

machine around that reentrant cycle. Finally, using the perturbations he linearizes the walking machine's step to step motion and mathematically demonstrates that the linearized system is stable.

When running you bounce on your leg rather than falling as an inverted pendulum. McGeer's passively stabilized running simulation is similar to his passively stabilized walking machine, except that the legs and hips have springs. The leg springs allow the robot to bounce on its feet, in much the same way as Raibert's hopping biped. Springs at the hips bring the proper leg forward to continue running. Again, with the proper spring constants, leg lengths, and mass properties this type of mechanism can run stably. However, some modes are extremely difficult to stabilize reliably and the set of valid initial conditions is small (McGeer 1989).

2.4 Self-Stabilizing Running

A self-stabilizing running robot has no computer, but does have actuators which go through a fixed, repeating cycle. For my studies of self-stabilizing running I assume each actuator operates in series with a spring at its joint. I also place a damper in parallel with the actuator-spring combination. There is a cycle timer and there are no other inputs to the actuator. The actuator has a fixed function from cycle time to actuator length. A properly constructed robot, with appropriate paths for the actuators over time, can run in a stable, dynamic manner. The first time I observed this phenomenon was with a simulation of a cockroach. The cockroach's control was a single digitized cockroach stride, played back as desired positions for the actuators. Examination of the simulated results indicated that it was bouncing on its legs as would be expected for a running robot, and it was stable despite the lack of feedback in the controller.

The self-stabilizing quadruped I simulate consists of two bipeds, connected by a back. Each biped is two self-stabilizing monopods. The actuators in the hips connecting the biped's legs, and those in the back connecting the quadruped's bipeds, counteract the interference between the different monopods. Thus, the stability of the monopod determines the stability of the more complex creatures, in much the same way as Raibert's more complicated robots build upon his hopping monopod. Additionally, the monopod bounces on its legs in a manner similar to McGeer's and Raibert's robots.

As mentioned in section 2.2, a monopod with feedback can exhibit any period gait, including chaotic motion (Koditschek & Buhler 1991) (Burdick, Vakakis, & Caughey 1989). The relevant analysis depends on the fact that the maximum hopping height of the monopod completely determines the motion of the next hop. For a self-stabilizing monopod, unlike an actively stabilized monopod, the maximum hopping height does not determine the motion over the next hop unless you also include information about the position of the actuator within its cycle. The analysis does not readily extend to the addition of a second variable, and is not applicable to a self-stabilizing monopod. In fact, the behavior of a self-stabilizing monopod is simpler than that of an actively stabilized monopod.

2.5 Juggling

Research on juggling is also applicable to understanding running. Running is juggling where you have an actuator attached to a bouncing body, rather than a bouncing body hitting a moving actuator. (Buhler 1990) demonstrates that simple feedback control can stabilize the hopping height of a juggled ball. Stefan Schall and Christopher Atkeson take this one step further and show that if the paddle path is properly designed one does not need feedback to juggle a ball vertically (Schaal & Atkeson 1993).

Schall and Atkeson's work deals with an oscillating paddle bouncing a single ball in the air, in the same way that an oscillating leg actuator can lift a hopping robot. They assume the paddle motion is constant, unaffected by the ball. Schall and Atkeson also model the paddle impact as instantaneous, with damping added by a coefficient of restitution.

They find "focusing trajectories" which correspond to stable vertical bouncing motions. They also managed to create a period 2 juggling oscillation, a behavior which I was unable to stably reproduce with a monopod simulation. I believe this difference comes from the fact that the ground contact times are much shorter for juggling. Additionally, since their period 2 juggling results were from a physical juggling robot it is possible that they are not completely stable, but will naturally diverge after a long time. This is a behavior I have observed in my monopod, something which looks like a period 2 hop but actually converges slowly on a single period oscillation, or eventually tips over.

2.6 Summary

Self-stabilizing running is a subset of dynamically stabilized running and builds upon many other forms of running. Passively stabilized running suggests that it is possible to run stably, without a computer actively controlling the robot. Work with actively stabilized running introduces the actuator-spring combination for control, as well as the general leg structure. Statically stable walking contributes some more complex behaviors which may be layered on top of the self-stabilizing motion, such as obstacle avoidance or compensation for rough terrain.

In the future, it would be interesting to combine different aspects from the various types of legged locomotion. It should be possible to make a self-stabilizing robot which, if the actuators were to stop moving, is passively stabilized and will continue running, albeit a bit less stably. On top of that, one could layer an active control to allow user control and to take care of disturbances which the self-stabilizing robot cannot handle. Finally, in circumstances where the terrain is too rough to check quickly, speed is not desired, or a smoother motion is necessary a statically stable walking algorithm could take over, assuming that the robot has enough legs for statically stable navigation. The result should be an extremely versatile robot which, when running normally, should be fairly efficient.

Chapter 3

Simplified Models

I have created simulations of self-stabilizing quadrupeds, bipeds, and monopods. I will show that individual implementations of these simulations are mathematically stable, but the methods I use to show stability do not give a good intuitive understanding of why the self-stabilizing robots work. In this chapter, I use a simplified monopod to develop an intuitive understanding of why the monopod hops stably, rather than going into a chaotic or higher period oscillation. I also back up the intuitive explanation with mathematical calculations.

My self-stabilizing quadrupeds are stable because they are formed by pairs of self-stabilizing bipeds, with the back counteracting disturbances and stabilizing the quadruped fore and aft. The bipeds are stable for similar reasons: each biped is two self-stabilizing monopods, with the pelvis and hips counteracting the limited offset between the two monopod's heights. The monopods have shaped feet which stabilize their pitch. The feet tend to push the monopods back toward vertical when they pitch. Finally, the monopod vertical motion is stabilized by the construction of the leg and the path of the leg actuator. Because the multi-legged robots depends on the monopod's stability, most of this chapter is dedicated to the monopod.

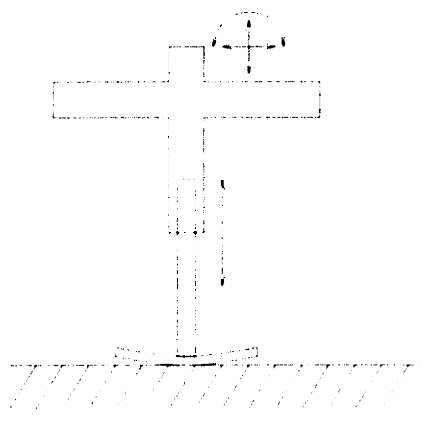

Figure 3-1: Monopod with curved foot. The model is planar, but can pitch. There is no joint at the hip.

3.1 Monopod

The monopod is the simplest of the self-stabilizing robots, and is vital to understanding the other robots. The monopod consists of a mass, attached to a foot by a linear actuator in series with a spring (Figure 3-1). There is a damper in parallel with the actuator and spring, representing friction in the vertical leg motion. The actuator is driven through a periodic motion with a fixed cycle time λ. The monopod can hop vertically, move horizontally, and pitch side to side. The actual monopod's foot has a mass which is small compared to the body. For this chapter, I simplify the model by having a massless foot.

I divide the monopod's motions into hopping height, touchdown phase, and pitch. First, I show that if the touchdown phase is constant and the monopod can only move vertically there is a stable hopping height. The stable hopping height is the point at which the actuator adds the same amount of energy as the damper removes over the course of a single cycle. Second, I show that the phase is stable. The stable touchdown phase is such that when the height stabilizes the time on the ground plus the time in the air is one complete actuator cycle. If the monopod touches down late, it will lose a little energy and remain in the air less, bringing the next touchdown closer to the stable touchdown phase (vice versa for an early touchdown time). Finally, I allow the monopod to pitch side to side and move horizontally, and show that a properly shaped foot can stabilize the additional degrees of motion. The pitch stabilization relies on an extension of the restoring forces you get from a wheel with an off-center mass.

Experimentally, there is a remarkably large set of actuator motions which will cause the monopod to hop stably, including sine waves, triangle waves, and square waves. For this chapter I simplify the leg actuator's motions into a single impulsive extension at time t_{extend}, of distance Δx, and a single retraction at some later point when the monopod is in the air.

3.1.1 Height Stabilization

The hopping height reaches equilibrium when the energy added to the hop by the actuator is balanced by the energy removed by the damper, and the total time in the air and on the ground is the same as the actuator's cycle time. I will address the relationship between hopping frequency and actuator frequency later; for now I am assuming that the time of touchdown t_{td} precedes t_{extend} by a constant amount of time, and that t_{extend} occurs before maximum spring compression.

For the hopping height to be in a *stable* equilibrium, if the total energy is greater than the equilibrium value the energy removed must be greater than the energy added, and vice versa when the total energy is lower. Additionally, the total energy removed or added must be small enough that the hopper will approach the equilibrium value without overshooting too badly. The energy introduced by the actuator motion is linear with respect to the touchdown velocity, while the energy removed from the damping effects is a quadratic function of the touchdown velocity. Where these functions cross, the hopping height is stable (Figure 3-2).

The remainder of this section is a mathematical justification of the existence of a stable hopping height. I approximate the energy change due to damping and the energy change due to the leg extension, and find conditions which will allow them to be equal. I then calculate the total energy change due to an error in touchdown velocity, and show that this

Flight				
	Time	t	$=$	0
	Height	$h(t)$		
	Energy	E_{tot}	$=$	$\frac{1}{2}mV^2 + mgh$
Touchdown				
	Time	t	$=$	t_{td}
	Vertical velocity	V_{td}		
	Height	$h(t_{td})$	$=$	0
	Leg spring compression	x	$=$	0
	Energy	E_{tot}	$=$	$\frac{1}{2}mV_{td}^2$
Leg extension				
	Time	t	$=$	t_{extend}
	Fraction of maximum compression	f	$=$	$\frac{x}{x_{max}}$
	Actuator extension	Δx		
	Energy introduced	ΔE_a	$=$	$\frac{k}{2}(2\Delta x f x_{max} + \Delta x^2)$
Maximum compression neglecting leg extension (a mathematical convenience, not an actual state)				
	Time	t	$=$	$t_{td} + \pi\sqrt{\frac{m}{k}}$
	Compression	x_{max}	$\approx$	$\frac{mg}{k} + V_{td}\sqrt{\frac{m}{k}}$
True maximum compression (occurs after leg extension)				
	Time	t	$=$	t_c
	Energy	E_{tot}	$=$	$\frac{1}{2}kx_c^2 - mgx_c$
Takeoff				
	Time	t_{to}	$\approx$	$t_{td} + \pi\sqrt{\frac{m}{k}}$
	Energy	E_{tot}	$=$	$\frac{1}{2}mV_{td}^2 + \Delta E_a + \Delta E_d$
Flight				
	Time	t	$=$	λ
	Energy	E_{tot}	$=$	$\frac{1}{2}mV_{td}^2 + \Delta E_a + \Delta E_d$

Table 3.1: Some monopod states and values over a single cycle. I use these values to demonstrate the vertical and pitch stability of the monopod.

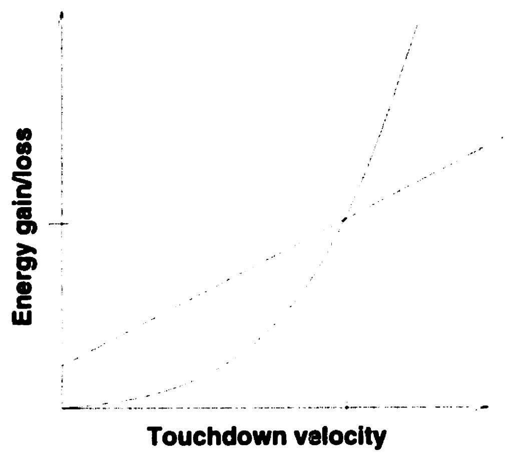

Figure 3-2: Energy removed by damping (solid) and added by the actuator (dashed) as a function of the touchdown velocity. The point where the two energies cross is the stable touchdown velocity. The total system energy ($E_{tot} = \frac{1}{2}mV_{td}^2$) is related to the touchdown velocity.

energy error decreases over successive hops.

A few approximations give a useful equation for the energy lost to damping over a single hop:

$$\begin{aligned}
\Delta E_d &= -\int_{t_{td}}^{t_{to}} bV^2(t) \\
V(t) &\approx \tilde{V}(t)V_{td} \\
\mathcal{K} &= b\int_{t_{td}}^{t_{to}} \tilde{V}^2(t) \\
\Delta E_d &\approx -\mathcal{K}V_{td}^2
\end{aligned}$$

The first equation is the formula I use for damping, in terms of velocity and a damping coefficient. Approximating velocity by saying that is scales linearly with the touchdown velocity is valid for an undamped spring. $\mathcal{K}$ is a constant which makes the final formulation for the energy loss have an obvious form.

The energy change due to the leg extension is a bit more complex. Define the height at touchdown as zero, so the total energy at touchdown E_{td} is completely kinetic. Let $x(t)$ be the spring compression, with x_{max} and E_{max} being the compression and the total energy at time t_{max}, the time of maximum compression neglecting the actuator extension.[1] At the time the actuator extends, $x_{extend} = fx_{max}$, with $f \in [0..1]$.

$$\begin{aligned}
E_{td} &= \frac{1}{2}mV_{td}^2 \\
E_{max} &= \frac{1}{2}kx_{max}^2 - mgx_{max} \\
E_{extend}^- &= \frac{1}{2}kf^2x_{max}^2 + \frac{1}{2}mV^2(t_{extend}) - mgfx_{max} \\
E_{extend}^+ &= \frac{1}{2}k(fx_{max} + \Delta x)^2 + \frac{1}{2}mV^2(t_{extend}) - mgfx_{max} \\
\Delta E_a &= \frac{k}{2}\left(2\Delta x f x_{max} + \Delta x^2\right) \\
x_{max} &= \frac{1}{k}\left(mg \pm \sqrt{m^2g^2 + kmV_{td}^2}\right) \\
x_{max} &\approx \frac{mg}{k} + V_{td}\sqrt{\frac{m}{k}}
\end{aligned}$$

Just before extension, the energy is E_{extend}^-, including spring energy, kinetic energy, and potential energy, and after extension the energy is E_{extend}^+. Subtraction gives the energy change, ΔE_a. Setting E_{max} equal to E_{td} gives a formula for x_{max} in terms of the touchdown velocity. Under most useful circumstances, the square root in the equation for x_{max} can be approximated away.[2] The energy ΔE_a added by the actuator's extension is therefore expressible in terms of f and V_{td}.

$$\Delta E_a = k\Delta x f\left(\frac{mg}{k} + V_{td}\sqrt{\frac{m}{k}}\right) + \frac{k}{2}\Delta x^2$$

Setting $\Delta E_a = \Delta E_d$ and solving for the touchdown velocity gives V_0 in terms of f.

$$V_0 = \frac{\sqrt{mk}}{2K}f\Delta x \pm \frac{1}{2K}\sqrt{mk\Delta x^2 f^2 + 4Kmgf\Delta x + 2Kk\Delta x^2}$$

[1]Note that although the time of maximum compression occurs after t_{extend}, I am not calculating x_{max} with the leg extension included. This is because x_{max} is used to determine the amount of energy introduced into the spring by the actuator extension.

[2]In a simple simulation, I had $\frac{k}{m} = 100$, $g = 9.81$, and $V_{td} = 3$. Systems where the approximation $kV_{td}^2 \gg mg^2$ is invalid correspond to hoppers which barely get off the ground.

Because all the values in the square roots are positive, V_{td} is guaranteed to exist for any given f and Δx. Additionally, to have a positive velocity at touchdown, we must use the positive square root.

Therefore, given f there is a touchdown velocity V_0 which corresponds to no energy change over a complete hop. Showing that this energy level is stable requires two parts. If you have a touchdown velocity $V_0 + V_e$, the energy change must have sign opposite the energy error caused by V_e and must result in an energy change of less than twice the energy error caused by V_e. The sign change is necessary to ensure that over time the velocity error disappears, and the maximum velocity change prevents an overall energy error increase from overshooting V_0.

Given a touchdown velocity V_0+V_e, the energy changes due to damping and the actuator can be added to get the energy change over a cycle ΔE.

$$\begin{aligned}
\Delta E_d &= -\mathcal{K}(V_0 + V_e)^2 \\
\Delta E_a &= \frac{k}{2}\Delta x^2 + \Delta x f m g + \Delta x f \sqrt{mk}(V_0 + V_e) \\
\Delta E &= \Delta x f \sqrt{mk} V_e - 2\mathcal{K} V_0 V_e - \mathcal{K} V_e^2
\end{aligned}$$

For ΔE to have a sign opposite V_e, you need $2\mathcal{K}V_0 + \mathcal{K}V_e > \Delta x f \sqrt{mk}$. Substituting the earlier equation for V_0, a lower bound on V_e is $V_e > -V_0$, and there is no upper bound.

To guarantee that the change in energy is not too great, consider the energy error ΔE_e resulting from a velocity error V_e, set the change in energy over the step to be enough that the energy error magnitude increases, and do some algebra to get a necessary condition for the energy to oscillate unstably.

$$\begin{aligned}
\Delta E_e &= \frac{1}{2} m(2V_0 V_e + V_e^2) \\
\Delta E &> -2\Delta E_e \\
\Delta x f \sqrt{mk} &> (\mathcal{K} - m)(2V_0 + V_e)
\end{aligned}$$

If $\mathcal{K} < m$, the fact that $\Delta x f \sqrt{mk} \geq 0$ results in $0 < 2V_0 + V_e$, an inequality which is true.[3] In the case where $\mathcal{K} > m$, the hopping motion is only stable if $V_e < \frac{\Delta x f \sqrt{mk}}{\mathcal{K} - m} - 2V_0$. As the damping effects increase, the maximum allowable error steadily decreases.

Therefore, as long as the damping is small ($\mathcal{K} < m$) and the touchdown velocity error is less than the touchdown velocity, given f and Δx there exists a touchdown velocity V_0 which is stable.

$$V_0 = \frac{\sqrt{mk}}{2\mathcal{K}} f \Delta x + \frac{1}{2\mathcal{K}} \sqrt{mk\Delta x^2 f^2 + 4\mathcal{K} mg f \Delta x + 2\mathcal{K} k \Delta x^2}$$

3.1.2 Phase Stabilization

If the phase of the monopod is stable, it should hop with a touchdown time t_{tds} fixed relative to the actuator motion and a period λ which is the same as the actuator motion's period. Just as with the hopping height, I will show that the phase has a fixed point, and that this fixed point is stable.

The existence and stability of the fixed point comes from the relationship between phase and the velocity at takeoff. Earlier touchdowns result in larger vertical velocities, while later touchdowns result in smaller vertical velocities. The time in the air is directly proportional to the vertical velocity, and the time on the ground is a constant determined by the spring and mass (neglecting damping and leg extension). Given an actuator oscillation period λ, there is a touchdown time such that the time in the air plus the time on the ground is λ (there are some limits on λ, which I will discuss later). If the monopod should touch down late, the smaller vertical velocity will decrease flight time, making the next touchdown closer to the fixed point, and vice versa for early touch down times.

The remainder of this section justifies this explanation of hopping phase stability. I calculate the amount of time and in the air, as a function of when the monopod touches down, assuming a stable hopping height for that touchdown time. I then create a function

[3]For this inequality to be false, the velocity at touchdown would have to be negative. That is, it would touch down while the body, with the foot attached, is flying into the air. Besides, we already have $V_e > -V_0$ as a lower bound on V_e.

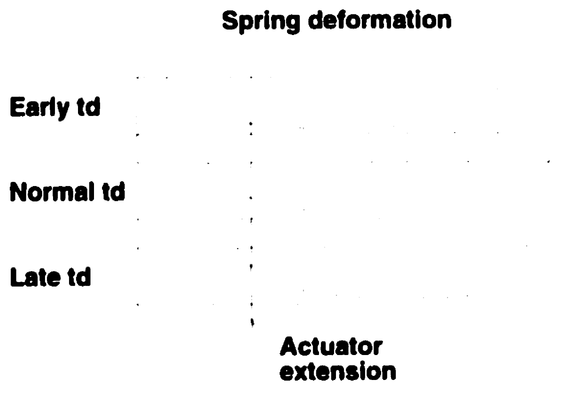

Figure 3-3: Fraction of maximum compression changes with time of touchdown.

from the touchdown time of one cycle to the touchdown time for the next cycle, show that the function has a fixed point, and show that perturbations from the fixed point will be pushed back toward the fixed point.

To show that it will achieve a stable phase, assume that the actuator has a single time t_{extend} at which it extends, as before. I have shown that, given a fraction f of maximum compression at which the actuator extends, there exists a touchdown velocity V_0 which is stable assuming f does not change from hop to hop. While the monopod is on the ground, it follows a sinusoidal path

$$x(t) = \mathcal{A}\sin\left(\sqrt{\frac{k}{m}}(t - t_{td})\right)$$

with $\mathcal{A}$ determined by the touchdown velocity (this equation neglects damping and leg extension). Therefore, as long as the leg extension occurs before the sinusoid reaches maximum compression, the fraction of maximum compression at which the actuator extends (Figure 3-3) is f.

$$f = \cos\left(\sqrt{\frac{k}{m}}\left(t_{td} - t_{extend} + \frac{\pi}{2}\sqrt{\frac{m}{k}}\right)\right)$$

As a consequence, any t_{td} has a touchdown velocity V_0 which would be stable, assuming that the next touchdown also occurred at t_{td}. If t_{td} increases to $t_{tdmax} = t_{extend}$, where f approaches zero, the appropriate touchdown velocity decreases to V_{tdmax}. At the other end of the spectrum, if t_{td} decreases to $t_{tdmin} = t_{extend} - \frac{\pi}{2}\sqrt{\frac{k}{m}}$, so that f approaches 1, the stable touchdown velocity increases.

$$\begin{aligned} V_{tdmax} &= \Delta x\sqrt{\frac{k}{2K}} \\ V_{tdmin} &= \frac{\sqrt{mk}}{2K}\Delta x + \frac{1}{2K}\sqrt{mk\Delta x^2 + 4Kmg\Delta x + 2Kk\Delta x^2} \end{aligned}$$

We need a relationship between touchdown time and the time until the next touchdown. Simplify the problem by assuming that upon touchdown, the velocity immediately changes to the touchdown velocity V_0 which corresponds to the f value associated with t_{td}, with no further energy changes. Under this assumption, the monopod bounces on the ground in a sinusoidal manner until the time $t_{td} + \pi\sqrt{\frac{m}{k}}$, when it takes off. By symmetry, at takeoff it has velocity V_0 upwards; it will remain in the air until gravity has completely reversed the velocity. The next touchdown will take place at time $t_{next}(t_{td}, V_0)$.

$$t_{next}(t_{td}, V_0) = t_{td} + \pi\sqrt{\frac{m}{k}} + \frac{2}{g}V_0$$

If t_{td} is as late as possible, the next touchdown will be at $t_{next}(t_{tdmax}, V_{tdmax})$. Similarly, if t_{td} is as early as possible the next touchdown will be at $t_{next}(t_{tdmin}, V_{tdmin})$. These give a bound on λ within which a fixed point t_{tds} exists.

$$\begin{aligned} t_{next}(t_{tdmax}, V_{tdmax}) &= t_{tdmax} + \pi\sqrt{\frac{m}{k}} + \frac{2}{g}V_{tdmax} \\ t_{next}(t_{tdmin}, V_{tdmin}) &= t_{tdmin} + \pi\sqrt{\frac{m}{k}} + \frac{2}{g}V_{tdmin} \\ \pi\sqrt{\frac{m}{k}} + \frac{2}{g}V_{tdmax} &< \lambda < \pi\sqrt{\frac{m}{k}} + \frac{2}{g}V_{tdmin} \\ \pi\sqrt{\frac{m}{k}} + \frac{2}{g}V_{tds} &= \lambda \end{aligned}$$

If λ is within the given bounds, a fixed point t_{tds} exists. If the monopod touches down at time t_{tds}, the next touchdown will occur at time $t_{tds} + \lambda$. Should the monopod touch down at time $t_{tds} + t_{error}$, the next touchdown will occur at time $t_{tds} + t_{error} + \pi\sqrt{\frac{m}{k}} + \frac{2}{g}V_{tds+error}$. However, velocity is a decreasing function of of touchdown time. Rewriting in terms of V_{tds} and V_{error}, the velocity at touchdown and the velocity change due to the touchdown time error, the next touchdown time is $t_{tds} + \lambda + t_{error} - \frac{2}{g}V_{error}$. Therefore, if the touchdown time is wrong it will be forced back towards the fixed point. This does not necessarily indicate that the fixed point is stable. A perturbation about t_{tds} could be overcorrected if $\frac{1}{g}V_{error} > t_{error}$. I have not witnessed this behavior.

3.1.3 Actuator Period Variation

The hopping motion is stable, regardless of actuator period λ, as long as the period is within the upper and lower bounds dictated by the phase stability derivation. Within those bounds, an increase in λ results in an increase in vertical velocity to give a longer flight time. If λ is too large, there may be no stable touchdown velocity which has enough air time. On the other hand, if λ is too small, the touchdown velocity may need to be zero.

As λ increases, the stable touchdown time t_{tds} decreases, and t_{extend} approaches t_{max}. When $t_{extend} = t_{max}$, the phase loses stability. If the monopod touches down early, the leg will extend after maximum compression. The energy introduced will decrease, and the stable velocity will be smaller, resulting in an even earlier touchdown next step. On the other hand, if λ decreases the takeoff velocity decreases until it no longer takes off.

3.1.4 Validating Approximations

~~In showing that the monopod generates a stable hopping height and phase, I make four~~ major assumptions. This section is an attempt to show that these approximations are reasonable.

- I neglect the motion changes due to the actuator when I calculate the damping.

- I neglect damping when calculating the monopod's motion on the ground, except to calculate the energy loss.
- I assume that the motion on the ground is well approximated by a spring, without damping or leg extension, to calculate the takeoff time.
- I assume that the energy stabilization happens in a single step.

I neglect the motion changes due to the actuator when I calculate the total damping because I assume that the damping is small compared to the total energy. As a result, the change of motion caused by the actuator extension is small. I have already showed that $\mathcal{K} < m$ results in guaranteed convergence, provided the $V_e > -2V_0$, and as $\mathcal{K}$ increases the range of allowable error begins to decrease. Increasing that to $\mathcal{K} \ll m$ gives $\mathcal{K}V_{td}^2 \ll \frac{1}{2}mV_{td}^2$; that is, the amount of energy taken out is much less than the total energy. The actual approximation I make when calculating the damping is $\mathcal{K} = b\int_{t_{td}}^{t_{to}} \tilde{V}^2(t)$, with $\tilde{V}(t) = V(t)/V_{td}$. On the ground, the motion is approximately sinusoidal (especially for low damping), with the result that $x(t) = V_{td}\sqrt{\frac{m}{k}}\sin(\sqrt{\frac{k}{m}}t)$. Increasing the amplitude of the oscillation as much as possible by extending the leg gives an oscillation bounded by $x(t) = (V_{td}\sqrt{\frac{m}{k}} + \Delta x)\sin(\sqrt{\frac{k}{m}}t)$. The corresponding velocity is $v(t) = (V_{td} + \Delta x\sqrt{\frac{k}{m}})\cos(\sqrt{\frac{k}{m}}t)$, which means that, effectively, V_{td} increases by at most $\Delta x\sqrt{\frac{k}{m}}$. Using the fact that damping is small, and in the area we are interested in the energy loss due to damping is approximately the same as the addition due to the leg actuator, $\Delta E_a \ll \frac{1}{2}mV_{td}^2$. From the equation for ΔE_a as a function of V_{td}, it follows that $V_{td} \gg \Delta x\sqrt{\frac{k}{m}}$. As a result, the leg extension will not even double the effective V_{td}. Any errors resulting from the approximation that the leg extension is not relevant to the damping can be counteracted by decreasing damping, which will increase the total stable energy. The increased total energy makes the change due to the leg extension less important.

The second major assumption I make is that for calculating the energy added by the leg extension you can neglect the damping. The energy added by leg extension is a function of the amount of energy currently in the spring, and damping can reduce the amount of energy by taking some away before the leg extends. However, this reduction is equivalent to a smaller value of f. Effectively, because of the damping $f \in [0..1-\epsilon]$ for some value of ϵ which will decrease as damping decreases. The lower maximum f reduces some of the boundaries, but does not greatly alter the behavior.

Third, I neglect the damping and actuator extension when calculating the takeoff time. The damping and actuator extension do not greatly affect the takeoff time, but the proof relies on changes in takeoff time so this approximation requires further consideration. I base my calculation of takeoff time on the motion of an undamped oscillator, whose frequency does not change with oscillation magnitude. A damped oscillator shares this property, although the oscillation frequency is lower. Therefore, adding damping into the computation merely extends the time until takeoff by a constant. The actuator extension, however, does not have a constant effect on the time until takeoff. As long as the actuator extends while the spring is compressing, the extension decreases the time until takeoff, which in turn decreases the time until touchdown. Figure 3-4 shows visually what happens when the leg actuator extends. The greatest takeoff time alteration occurs if the leg extends almost immediately after touchdown, while the smallest extension occurs if the leg extends when fully compressed. The effects of these extensions are scaled by $\frac{E_a}{E}$, and this fraction can be decreased as far as necessary be decreasing the damping (which increases the total energy at equilibrium). However, this is not necessary. As f decreases, the takeoff time decreases. The phase stability proof uses the fact that as f decreases, the time in the air decreases. Decreasing the time until takeoff brings the next touchdown time further forward, helping the phase stabilize.

Finally, when showing that the hopping phase stabilizes I assume that the energy stabilizes in a single step. This approximation allows solution for the phase. Simulations have shown extremely fast height recovery. Additionally, if the energy is slightly off, the time on the ground is unaffected. Only the flight time changes. Should the energy be too high, the flight time will extend, resulting in a later touchdown which will add less energy. The touchdown time change resulting from hopping height error aids the hopping height's

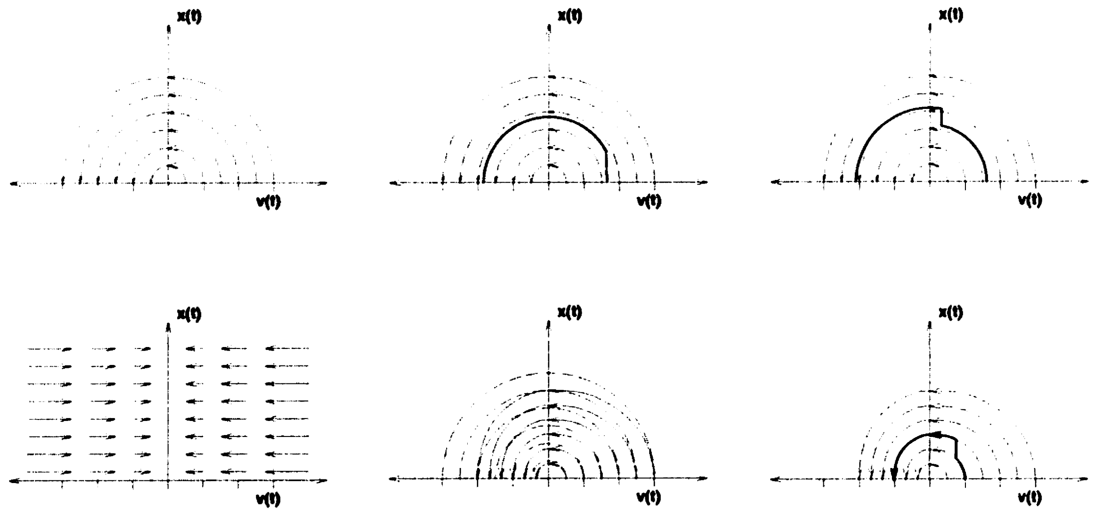

Figure 3-4: Graphing the spring compression on the vertical axis and the rate of compression on the horizontal axis gives a simplified explanation of what happens when the leg extends. If you scale the axes properly, the physics of the spring-damper oscillation rotates the system state about the origin at a constant rate (top left). Leg extension is equivalent to a sudden jump in the spring compression. Early leg extension results in a phase change (top middle), while late leg extension results in a magnitude change (top right). Takeoff occurs when the spring compression is zero. Damping results in a gradual push toward the vertical axis (bottom left), resulting in an inward curve (bottom middle). Damping and leg extension, together, can result in zero net energy change (bottom right).

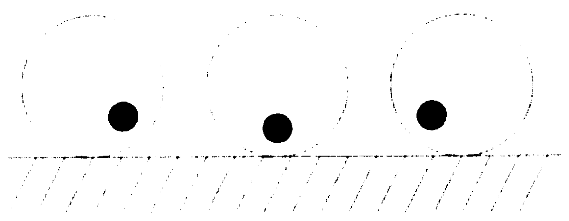

Figure 3-5: Wheel with an off–center mass. If the wheel rolls to one side, the mass is no longer above the contact point and applies a restoring torque. Another way of looking at it is to observe the total energy. If the wheel rolls to one side, the center of mass raises, increasing the total potential energy of the wheel. The wheel will then roll back to decrease the potential energy.

stabilization.

3.1.5 Pitch Stabilization

Given a monopod which hops stably vertically, with a constant phase, the remaining unstable degree of freedom is pitch. I base the pitch control on the curvature of the foot. It works much the same way as a wheel with an off–center center of mass: the wheel will roll side to side, until the damping of its rolling motion places the center of mass at its lowest point (Figure 3-5). For the monopod, though, the "wheel" is only on the ground part of the time. Moving a wheel's center of mass further off–center makes it more stable, but if the wheel is periodically off the ground moving the center of mass too far could have the opposite effect. If the pitch is strongly corrected while the wheel is on the ground, it may take off with a high angular momentum. While the wheel is in the air, it will continue to rotate, and may land with a much greater pitch offset than when it took off (Figure 3-6). Therefore, there is a limited set of foot radii which will stabilize hopping motion, but if the parameters of the monopod are chosen properly the set is nonempty. Figure 3-7 graphs the instability of the monopod at various foot radii. Whenever any of the graphed lines are above 1, the monopod will not hop stably.

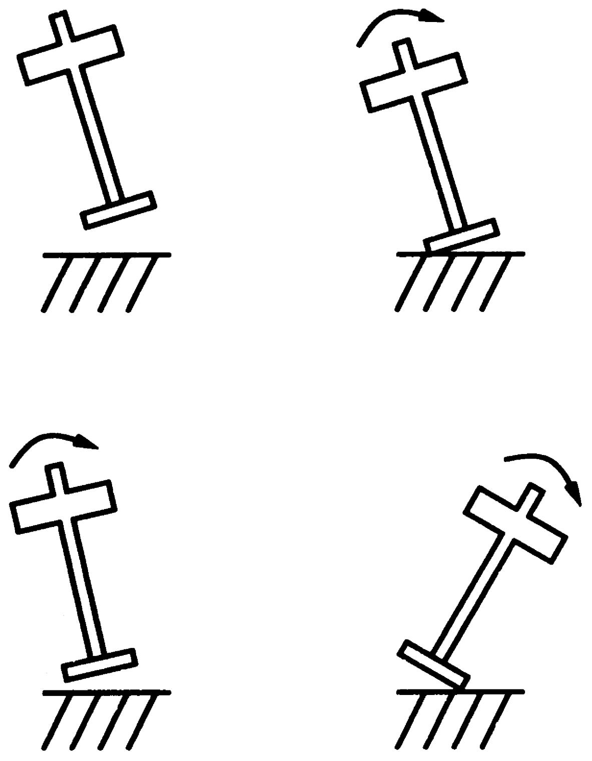

Figure 3-6: Monopod hopping, overcorrecting. If the foot is too flat, when the monopod comes down with a slight pitch (upper left) it will receive a large torque pitching it towards vertical (upper right). However, if it takes off too soon it may continue rotating in the air (lower left) and land with an even greater pitch error than it had on the previous step (lower right).

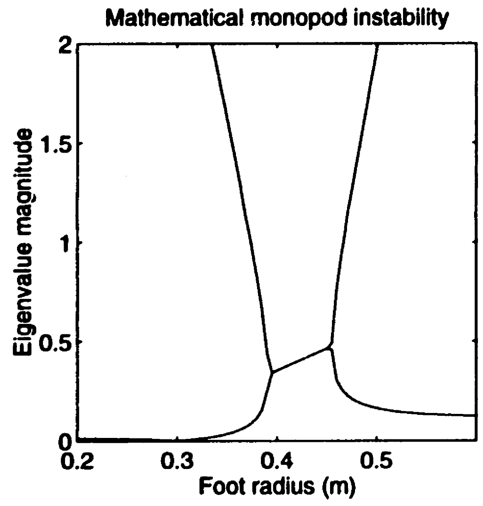

Figure 3-7: Instability of the monopod as the foot radius changes. This is a plot of all the eigenvalue magnitudes about the reentrant hopping trajectory. If any eigenvalue has magnitude greater than one, the monopod is unstable for that foot radius and will tip over. If the foot radius is too small, there will not be enough restoring force to keep the monopod upright. If the foot radius is too large, it will overcorrect and tip over. In between, there are a range of foot radii which cause the monopod to hop stably.

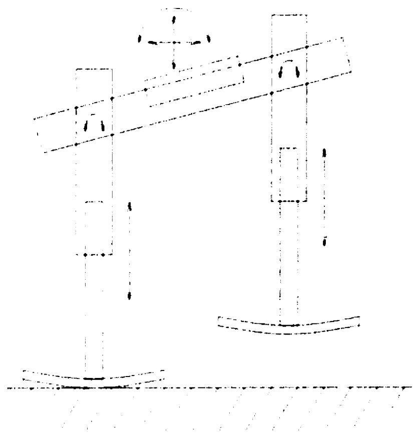

Figure 3-8: Biped with curved feet. The model is planar. There are pin joints at the hips, and linear joints along the legs. The stability of the monopods helps stabilize the biped locomotion.

3.2 Planar Biped

The biped is essentially a pair of self-stabilizing monopods hopping next to each other, connected by a pelvis (Figure 3-8). Intuitively, if I can correct for the perturbations caused by the pelvis then the monopods should self-stabilize. All the joints except the hip joints are in use, so it would be best to correct with only the hip joints. Although the simulation will allow the biped to tip towards either foot, it cannot leave the plane and therefore does not need stabilization out of the plane of the legs.

If the monopods are far apart from each other, there should be little interference except for the rotation of the body as the monopods hop out of phase. In this case, the expected monopod heights give expected pelvic rotations, which, negated and used as hip actuator paths, compensate for rotational disturbances. Another way to think of this is that the monopods self-stabilize into a vertical hopping position. Therefore, the hips should keep the biped's legs vertical as it hops.

As the length of the pelvis decreases, the pelvic rotation increases. Eventually, it begins interfering with the monopod hopping, and finally the difference between the expected

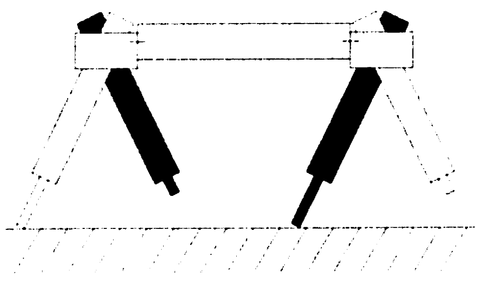

Figure 3-9: Quadruped with a twisting back. Note the two pins connecting the back to the hips; these pins are rotational joints. The front and back bipeds are stabilized by each other and the spine.

monopod heights could become greater than the pelvis length. In this case, the hips begin driving the pelvis while a foot is on the ground, rather than moving the legs to a vertical position for landing. There are specific times at which the pelvis position is important, such as leg touchdown and takeoff. In between, though, I use the hips to drive the pelvis to the next position. Because some time is spent in the air, rotational velocity upon takeoff is also important so that the amount of rotation upon landing is correct. Fortunately, the individual monopods are quite stable, and manage to reject the disturbances caused by the hips.

3.3 Quadruped

Finally, a quadruped is a pair of self-stabilizing bipeds (one for the front legs, one for the rear legs) connected by a back (Figure 3-9). These bipeds, as previously indicated, are stable within a plane perpendicular to the direction the quadruped faces. The back provides stability in the forward direction. All that remains is to see that the bipeds interfere with each other as little as possible.

The front and back bipeds do not have to hop in phase. Therefore, they may be rotated by different amounts at different times. The back isolates the two quadrupeds by twisting by the expected rotational difference. Additionally, the legs of the quadruped can swing forward. Forward leg motion allows forward movement and allows the legs to be used much like the hips of a biped, to help stabilize the quadruped if the back is short.

In total, because each of the various elements (monopod and biped) are quite stable the resulting quadruped is also stable. It can trot, pace, bound, and gallop.

Chapter 4

Approach

I use computer simulations and mathematical analysis to study self-stabilizing legged systems. For simulations with few degrees of freedom, such as the monopod, I mathematically calculate the motion of the running machine and verify the calculations against a computer-generated physically realistic simulation. As the degrees of freedom increase, I use physically realistic simulations. In both cases, I convert the results into discrete time to analyze the stability.

4.1 Analytical Solutions

Although mathematical notation can express the behavior of legged systems in a reasonably compact form, analysis of these mathematical models can be difficult. As the models get more complex, it becomes increasingly difficult to find closed form solutions. One can solve some systems by making approximations, such as using a polynomial approximation of the solution. More complex systems require techniques such as linearizing about simpler, previously found stable solutions. Linearization has the disadvantage that it assumes no, or little, interaction between disturbances in the portion being solved and disturbances in the previously stable solution.

The vertical height of the monopod over time has a closed form. However, the takeoff time has only a numerical solution, not an analytical one. Finding an initial height and height velocity which recurs as the monopod hops requires an approximation of the takeoff time. Additionally, the equations which determine both the height and the pitch of the monopod over time are intractable. Assuming that any disturbances to height will be damped out without causing pitch disturbances, however, allows you to linearize the system about the stable hopping motion of the monopod and solve for pitch. Such assumptions must be verified, either by simulating the mechanism with all its complexity or by building one (Appendix A).

4.2 Simulations

As a verification of the mathematics and to analyze the behavior of more complicated robots, I employ physically realistic simulations. A commercial dynamic modeling program (Rosenthal & Sherman 1986) generates the dynamics of the simulations. The Creature Library, an aid to simulation generation (Ringrose 1992a) automatically generates the rest of the simulation. I then add a control system which implements the actuator paths for a self-stabilizing robot. The simulations obey the laws of Newtonian physics as applied to trees of rigid bodies connected at joints.

The limbs and body have specified shapes and sizes. These shapes determine the mass properties, assuming the creature's density is that of water (most natural creatures are about the density of water). I assume that the links are rigid, and the simulation does not check for interpenetration. Normal operation of a running robot does not have the various parts hitting each other, so the computationally intensive task of checking for interpenetrating links is not necessary. Visual inspection of the results suffices.

Revolute joints, such as hips, have a torque source at the joint, and sliding joints, such as telescoping legs, have a force source along the sliding joint. These force and torque sources emulate a spring in series with a perfect actuator, in parallel with a velocity damper. This actuator configuration is equivalent to a spring with a settable rest length. I used this mechanism in previous simulations (Ringrose 1992b). Generally, the spring and damping values are fixed during a run and the control system only controls the actuator. For a self-stabilizing running machine, the actuator goes through a repeating, predetermined cycle while the actual joint may vary from the actuator position by compressing or extending the spring.

Arbitrary contact is a difficult problem for rigid-body simulations (Baraff 1989)(Moore & Wilhelms 1988)(Hahn 1988). For two rigid bodies to avoid interpenetrating, they must stop when they hit. Because the simulator applies forces, and forces result in accelerations, the only way to completely stop the bodies is to apply an impulse which cancels the bodies' relative velocity. I get around this problem by ignoring it: portions of the simulation can interpenetrate. The control system, not the simulation, sees to it that the various portions of the robot do not hit each other. Visual inspection of animated results allows a human operator to check for interpenetration problems.

I allow the foot to slightly interpenetrate the ground and use the interpenetration distance as a measure of how much force the ground should apply. This method approximates a flexible foot pad which deforms slightly under pressure. I model ground contact by creating a spring-damper system connecting predetermined contact points to the ground when they reach zero height. This contact disappears as soon as it would pull the foot into the ground. Although this method of handling ground contacts is simple, it has the drawback that it has to be tuned for different objects (Figure 4-1). When the foot has a region, all of which can touch the ground, I spread multiple ground contacts across the region and treat them individually. This method allows me to emulate a curved foot on arbitrary ground with a reasonable degree of accuracy. It is simpler and more compatible with the simulation than other contact models proposed for curved surfaces (Baraff 1990).

The simulation does not include minor effects such as air resistance. Nor does it include the possibility of a limited amount of power being available to move the actuators, or external disturbances. However, if a well-made simulation indicates stable motion, a physical robot similar to the simulation should be able to handle the differences between the simulation and reality as disturbances to the motion. As a sanity check on my simulations, I

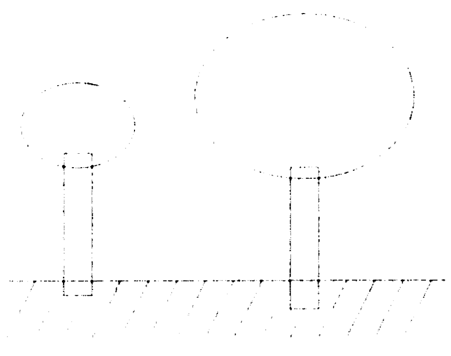

Figure 4-1: When springs implement ground contact, they are not precisely accurate. Two objects with different weights, or different momentums, will penetrate the ground differently. This behavior does approximate the flexibility of a foot pad.

have a physical self-stabilizing monopod robot (Appendix A).

4.3 Continuous and Discrete Time

Self-stabilizing running machines exist in continuous time. Their running and hopping motions are a continuous function of time $\vec{X}(t)$ (this state vector uniquely determines the position, configuration, and velocities of a hopping machine). However, the self-stabilizing control is a periodic input with a fixed cycle time λ. It is possible to construct a discrete stride function $\vec{S}(\vec{X})$ from $\vec{X}(t)$ such that $\vec{S}(\vec{X}(k\lambda)) = \vec{X}((k+1)\lambda)$ for integer k. Because the state vector $\vec{X}$ is complete, stability analysis on $\vec{S}$ and $\vec{S}(t)$ are equivalent unless the simulation is pathologically ill-behaved (Figure 4-2). To date, I have not observed any simulations where stability analysis of $\vec{S}$ differs significantly from the true simulation stability.

Most running machines cannot usefully translate their continuous time state vector into a useful discrete time stride function because their control is not cyclic. However, there are a number of advantages to studying discrete-time stability. The foremost advantage is that a running cycle is a single fixed point of $\vec{S}$, rather than a repeating cycle of positions $\vec{X}(t)$. It is much easier to analyze the stability of a single fixed point, rather than a repeating cycle.

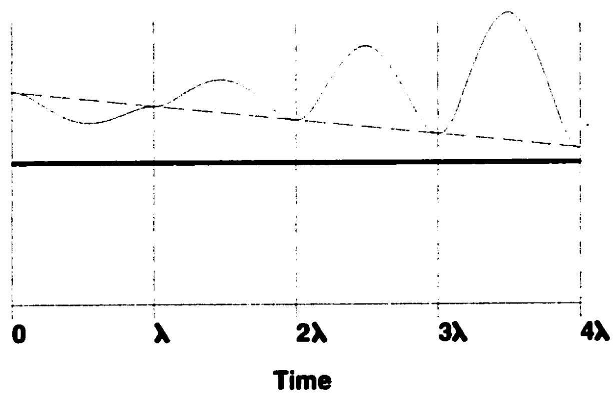

Figure 4-2: An example of a pathologically ill-behaved simulation in which a stability analysis of the discrete stride function $\vec{S}$ concludes stability, while the motion is unstable. The heavy solid like is the fixed point of the stride function, the dashed line is the approximation given by $\vec{S}$, and the lighter solid line is the actual path of $\vec{X}(t)$. Note that the closer $\vec{X}(kt)$ gets to the fixed point of the stride function, the greater the excursions. Although it may be possible, I have not observed this behavior in any running simulation.

4.4 Stability

I commonly use stability to mean "If you push it, it will eventually come back to where it was." However, this is too vague to prove; it can only be disproved. A more precise, mathematical definition is necessary. I have adapted this definition of stability from (Siljak 1969), pages 308–309.

Let $\vec{X}(t)$ be the time-dependent position, configuration, and velocities of the machine. Self-stabilizing running robots have a periodic control with a cycle time λ, so define $\vec{S}(\vec{X})$ as the appropriate stride function. As mention in section 4.3, stability analysis of the stride function is sufficient.

Let $\vec{X}_s$ be a fixed point of $\vec{S}(\vec{X})$; therefore $\vec{S}(\vec{X}_s) = \vec{X}_s$. $\vec{S}(\vec{X})$ is stable about $\vec{X}_s$ if for every real number $\epsilon > 0$ there exist a real number $\delta(\epsilon) > 0$ so that $\left|\vec{X}_0 - \vec{X}_s\right| \leq \delta$ implies $\left|\vec{S}^k(\vec{X}_0) - \vec{X}_s\right| \leq \epsilon$ for all $k \geq 0$ ($\vec{S}^k(\vec{X})$ represents k applications of $\vec{S}$ to $\vec{X}$).

This definition of stability allows a system to oscillate around the fixed point without actually approaching it, so one more condition is necessary: There is some real constant $\nu > 0$ and to every real number $\rho > 0$ there corresponds a real number $K(\rho, \vec{X}_0)$ so that $\left|\vec{X}_0 - \vec{X}_s\right| \leq \nu$ implies $\left|\vec{S}^k(\vec{X}_0) - \vec{X}_s\right| \leq \rho$ for all $k \geq K(\rho, \vec{X}_0)$.

All of this means, essentially, that any motion which starts sufficiently close to $\vec{X}_s$ will converge toward $\vec{X}_s$ over consecutive running cycles. This definition of stability specifically excludes all running cycles with a period other than 1 (such as a monopod which alternates between high and low hops).

4.4.1 Global Stability

Global stability (also referred to as stability in the large, or completely stable) means that any initial state will converge to a given fixed, stable hopping oscillation. In its most general form, it is highly unlikely since it includes such absurd starting conditions as upside down or underground. For simple models, such as a linear monopod which cannot pitch or move horizontally, it is possible to prove global stability (Koditschek & Buhler 1991).

It is sometimes possible to determine a smaller set of initial conditions which are guaranteed to converge. This is no longer *global* stability, it is local. The largest set of initial conditions which converge to a given fixed, stable hopping oscillation is known as the basin of attraction. If there are multiple stable fixed points, each will have its own independent basin of attraction.

The stride function $\vec{S}$, which determines the state of the machine on the next stride given the current state, is highly nonlinear. When $\vec{S}$ can be mathematically determined, it may be possible to solve to determine global stability, or find the basins of attraction for the fixed points. For the more complex machines one cannot calculate $\vec{S}$, so proving global stability is impractical as well as extremely unlikely. Calculating the basin of attraction is also impractical, as without a mathematical definition of $\vec{S}$ there is no good way to determine the size of the basin or define its edges.

4.4.2 Local Stability

We can linearly approximate the nonlinear stride function. Near the stride function's fixed points, this linear approximation gives an accurate estimation of the effects of small perturbations. A linear approximation of $\vec{S}$ can be computed numerically from repeated simulation runs.

One can approach a fixed point by using the Taylor series expansion of the stride function and the formula for a fixed point. Other methods for finding a fixed point of an implicit function exist, but this one is fairly simple and works well enough for my purposes. For a $\vec{X}_0$ sufficiently close to a fixed point $\vec{X}_s$ of $\vec{S}$, you have $\vec{S}(\vec{X}_0) \approx \vec{S}(\vec{X}_s) + (\nabla\vec{S})(\vec{X}_0 - \vec{X}_s)$ (there are higher order terms in this Taylor series expansion, but if $\vec{X}_0$ is close enough to $\vec{X}_s$ they can usually be safely ignored for well behaved $\vec{S}$). Note that $\nabla\vec{S}$ is the gradient of $\vec{S}$ taken at $\vec{X}_0$. In the cases where $\nabla\vec{S}$ or $\vec{X}_s$ cannot be evaluated mathematically, they can be approximated numerically. Given:

$$\begin{aligned}
\vec{S}(\vec{X}_s) &= \vec{X}_s \\
\vec{X}_s &= \vec{X}_0 + \Delta\vec{X} \\
\vec{S}(\vec{X}_0 + \Delta\vec{X}) &\approx \vec{S}(\vec{X}_0) + (\nabla\vec{S})(\Delta\vec{X})
\end{aligned}$$

one can calculate an approximation $\vec{X}_s$:

$$\begin{aligned}
\vec{X}_s &\approx \vec{S}(\vec{X}_0) + (\nabla\vec{S})(\Delta\vec{X}) \\
\vec{X}_0 + \Delta\vec{X} &\approx \vec{S}(\vec{X}_0) + (\nabla\vec{S})(\Delta\vec{X}) \\
(\nabla\vec{S} - I)(\Delta\vec{X}) &\approx \vec{X}_0 - \vec{S}(\vec{X}_0) \\
\Delta\vec{X} &\approx \left(\nabla\vec{S} - I\right)^{-1}\left(\vec{X}_0 - \vec{S}(\vec{X}_0)\right) \\
\vec{X}_s &\approx \vec{X}_0 + \left(\nabla\vec{S} - I\right)^{-1}\left(\vec{X}_0 - \vec{S}(\vec{X}_0)\right)
\end{aligned}$$

This approximate relation between $\vec{X}_0$ and $\vec{X}_s$ allows one to determine $\vec{X}_s$ numerically if they are reasonably close together. I have determined experimentally that if you start with a reasonable[1] value of $\vec{X}_0$, iterating this equation returns steadily improving estimations of $\vec{X}_s$, although convergence is not guaranteed.

Repeated evaluations of the stride function near $\vec{X}_0$ will give you a numeric approximation of $\nabla\vec{S}$. The approximation requires one evaluation of $\vec{S}$ near $\vec{X}_0$ for each degree of freedom in $\vec{X}$, plus one evaluation at $\vec{X}_0$. For greater accuracy, I sometimes use more evaluations near $\vec{X}_0$ to help cancel higher-order terms of the Taylor series.

The linear approximation of $\vec{S}$ simplifies the process of determining local stability. The deviation $\vec{X}_e$ from a fixed point $\vec{X}_s$ propagates from stride to stride with the simple matrix equation $\vec{X}_e^{k+1} = A\vec{X}_e^k$, where $A = \nabla\vec{S}$ at $\vec{X}_s$. This equation is a discrete time matrix equation; its stability can be measured by the eigenvalues of A.

The matrix A has as many eigenvalues e_i and associated eigenvectors $\vec{V}_i$ as it has columns. The eigenvector determines the mode which is damped or increased by the eigenvalue. Imaginary eigenvalues come in pairs, as do their eigenvectors, and indicate oscillating modes. If the magnitudes of the eigenvalues of A are all less than one, all modes in which $\vec{X}$ could be excited are dissipative and $\vec{S}$ is locally stable at $\vec{X}_s$. The smaller the eigenvalue magnitude, the more quickly the corresponding eigenvector's motion is dissipated. If any eigenvalue of A has magnitude greater than one, $\vec{X}_s$ is unstable. Any disturbances along the corresponding eigenvector will increase exponentially. A magnitude of one indicates that the corresponding eigenvector is undamped. However, if the eigenvalue magnitude is close to unitary, the original mechanism's nonlinearities, neglected here, will probably determine its stability.

An alternate, less reliable way of verifying the stability of a configuration is to create a simulation, let it run in the presence of disturbances, and examine the data for instabilities. This method helps verify that linear approximations reflect the relevant dynamics.

[1]A reasonable value of $\vec{X}_0$ is one which does not involve impossible situations such as upside down, underground.

4.5 Summary

In order to determine the stability of a system, I determine the motion over time, either mathematically or through simulation. I then implicitly create a stride function which given the system state returns the state one stride later. If the robot is stable, the stride function will have a stable fixed point. I find a fixed point of the stride function, linearly approximate it, and take the gradient. The eigenvalues of the gradient determine the self-stabilizing properties of the system about that fixed point.

I never explicitly solve for the stride function. Instead, I use techniques which only require evaluating the step function at specific points, and run the simulation for a single actuator cycle to find out the result. These techniques scale reasonably well. I was able to apply them to simulations with up to 20 degrees of freedom. The limiting factor, in terms of scaling, is calculating the reentrant trajectory.

On the other hand, because the stability calculation is based on the gradient of a linearized step function the result only indicates local stability. There is no reliable indication of how far you can go from the stable trajectory and still return to it.

Finally, it is worth noting that there may be several reentrant trajectories. This is basically different ways that the robot could run without changing the actuator paths. For example, a bounding quadruped normally runs on its front and back leg pairs. If you tilt it on its side and balance it properly, it might be able to bound on the two legs on that side. This motion would be another reentrant trajectory, probably unstable. None of the simulations I have used showed multiple stable reentrant trajectories, unless you count trajectories where the robot has already crashed and is thrashing on the floor.

Chapter 5

Control

One of the goals of this work is to demonstrate that self-stabilizing running quadrupeds can use a variety of gaits. Work to date has included trotting, pacing, and bounding, but has excluded galloping. Although self-stabilizing running involves no sensing, there is a "control system" which contains the motion patterns for each actuator. All computations for these patterns are static, and do not change during the course of a run. I break the quadruped design down into separate parts, building on the previous self-stabilizing controllers. I treat the quadruped as a pair of bipeds connected by a back, and each biped as a pair of connected monopods.

I select a stable vertical oscillation for the monopod. Based on that oscillation, I calculate a foot radius which will stabilize the pitch. I then determine the necessary torque at the hips for the biped to keep the biped stable. Finally, I coordinate the quadruped's twisting back with the expected biped motion to generate the control for a self-stabilizing quadruped.

5.1 Monopod

The monopod consists of a single telescoping leg attached to the body (Figure 3-1). The linear action of the leg controls the hopping height, while the curvature of the foot controls the body pitch. In simulation, the leg and body both have mass, but I ignore the mass of the leg when doing analysis. The monopod is able to move vertically, move horizontally, and pitch, but it is constrained to move in a plane.

5.1.1 Hopping Height

Many leg length patterns will result in a stable vertical oscillation. When working on the vertical hopping motion, I constrain the simulation to move vertically without pitching or horizontal motion.

Many leg actuator patterns, including square waves, sine waves, and triangle waves can result in stable hopping. I use a pieced-together polynomial which happens to give a stable, visually appealing hop. The leg actuator pattern, given an oscillation period λ and an oscillation magnitude C, is:

$$l(t) = \begin{cases} C((2 * t/\lambda)^2 - 1) & \text{if } t < \lambda/2 \\ -C^{\frac{0.5^2 - t/\lambda + (t/\lambda)^2}{.3^2}} & \text{otherwise} \end{cases}$$

This pattern, and the actual leg length over time, are in figure 5-1. It has period λ, and ranges in value from 0 to C.

5.1.2 Pitch

The shape of the foot controls the pitch of the hopping monopod. If the monopod tilts to one side, the curve of the foot moves the contact point in the same direction. If the foot is curved properly, the contact point will move further than the center of mass, in much the same way as a wheel with an off-center mass (Figure 3-5). The location of the contact point results in a restoring torque on the body. Additionally, the pitch motion is damped by the foot rolling on the ground during ground contact.

In order to determine the proper foot radius, I begin with a stable vertical hop. I then calculate the flight time t_f, the stance time t_s, the height of the center of mass $h(t)$, and the force applied to the ground $F(t)$. From these I can calculate the torque the ground applies to the body while they touch, and the body's angular momentum while in the air.

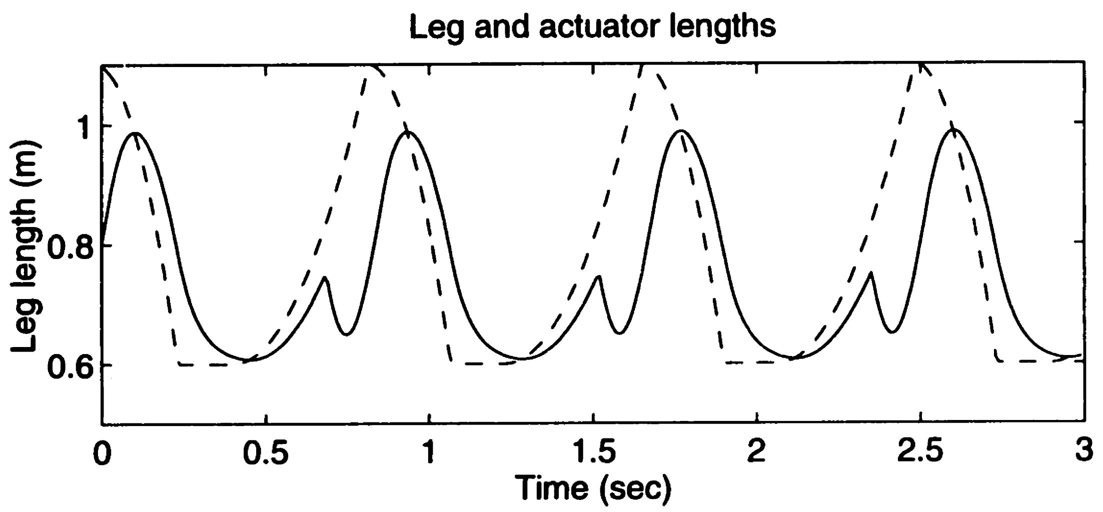

Figure 5-1: Leg length (solid) and actuator length (dashed) over time for stable hopping. The leg has position-damping servomechanism, which does not apply force unless the leg is offset from the actuator position. The difference between actual and desired leg length is caused by the force when the foot contacts the ground.

These will determine the pitch and pitch velocity upon the next landing. I assume that small alterations in pitch will minimally affect the vertical hopping motion, an assumption which I validate with a simulation including both pitch and vertical hopping motion.

The pitch and pitch velocity upon the next landing, as a function of the current pitch and pitch velocity, gives a formula for the stride function of pitch and pitch velocity. As outlined in section 4.4.2, the eigenvalues of the gradient of the stride function will indicate how stable the monopod's pitch is. Altering the radius of the foot changes the gradient, and thus the eigenvalues. I graph the instability for various foot radii and choose one which works with as large a margin for error as possible. The result is a stabilized, damped pitch motion.

5.2 Biped

Structurally, the biped has a pair of telescoping legs, each identical to the monopod's leg, attached by rotating hips to different ends of a pelvis. There is also a mass attached to the middle of the pelvis, in anticipation of a pair of bipeds being joined together (Figure 3-8). The model is planar: it can rotate, move vertically, and move horizontally, but it cannot move out of the plane of the hips. The hip joints are not present in the monopod, and must be controlled.

Given two monopods hopping stably a long distance apart, one can calculate the pelvic rotation of the equivalent biped from the monopod heights over time. The rotation over time determines the desired hip angles to keep the biped legs vertical. If the biped rotates further than expected, the individual monopods will tilt as well. Because the monopods are stable, they will apply restoring forces to regain their vertical positions. Note that the monopods are only locally stable, and too large a disturbance will overcome the monopod's stability and tip the biped over.

As the monopods are placed closer together, the fixed connection between them begins to interfere with their stable hopping. When the distance between the hips is close to the maximum difference in monopod height, the connection between the hips will tend to pull the monopods over. However, to some extent it is possible to force the body pitch to pass through a specific path. Modifying the hip path applies torques at the hips, which will alter the body pitch while the foot is on the ground.

When the hips are close enough together that the hopping motions interfere with each other, the important times for the orientation and velocity of the body to be correct are when legs take off or touch down. Therefore, I determine the desired hip positions from the expected hopping heights of the individual monopods at those important times and smoothly connect those positions and velocities.

The expected monopod heights determine the body orientation at touchdown and take-off. While the biped is in the air, the body's angular velocity is approximately constant, as the feet have small mass and therefore little effect. Therefore, the angular velocity at both takeoff and touchdown is fixed by the orientation at touchdown and takeoff. Because the monopods try to keep the legs vertical, the expected hip positions are determined by the expected body positions. However, to some extent when one foot is on the ground the hips determine the body orientation. I fill out the hip path between takeoff and touchdown with a linear interpolation, and then between touchdown and takeoff with a smooth curve which has zero acceleration at each end. The fact that the curve has zero acceleration at each end means that there will be fewer problems if a foot touches down or takes off at the wrong time.

5.3 Quadruped

The quadruped is two bipeds, joined by a back (Figure 3-9). The back can twist along the axis connecting the two bipeds, and has mass. The back is long enough that there is little interference between the front and rear bipeds. The quadruped, unlike the biped and monopod simulations, is capable of translation in any direction, as well as yaw, pitch, and roll motion. The two bipeds support each other along the connecting back. The legs are capable of swinging side to side, just as in the biped, and front to back. The side to side motion is controlled in the same way as the biped's hips, while the forward motion of the legs counters the expected pitching of the robot and moves the quadruped forward. Despite the fact that the quadruped is three-dimensional, the quadruped's feet are circular in the same plane as the biped rather than hemispherical.

The legs coordinate their forward motion so that at the times they are expected to be on the ground the feet are moving backward at the quadruped's desired forward speed. Unless otherwise indicated, this forward speed is zero. The legs also extend so that the vertical distance is unchanged by the forward displacement. This forward motion of the legs is superimposed upon additional hip motion which counteracts the expected pitching motion of the quadruped. The pitching between the front and back bipeds is similar to that of a single biped, except that the extra length of the back increases stability.

The back twists according to the expected motions of the front and back bipeds, enabling them to go through their motions with minimal interference. However, the back also serves as a stabilizing influence. Any instabilities which appear in one biped will be partially transferred to the other, and both bipeds will help damp it out.

5.3.1 Gaits

A quadruped has a large number of possible gaits. Historically, these gaits have been defined by the times at which the various feet touch down. Decide on a single reference foot (usually the front left). Each foot has relative phase ϕ_i, the fraction of the running cycle which occurs between the touchdown of the reference foot and the touchdown of foot i. Additionally, each leg has a duty factor β_i of each leg, the fraction of the running cycle that the leg is on the ground (McGhee 1968). Under normal circumstances β_i is assumed to be the same for each foot. Note that β_i can be fairly large and still have the creature exhibit running behavior (McMahon, Valiant, & Frederick 1987).

Figure 5-2 illustrates the quadrupedal gaits I have attempted to generate using self-stabilizing quadrupeds.

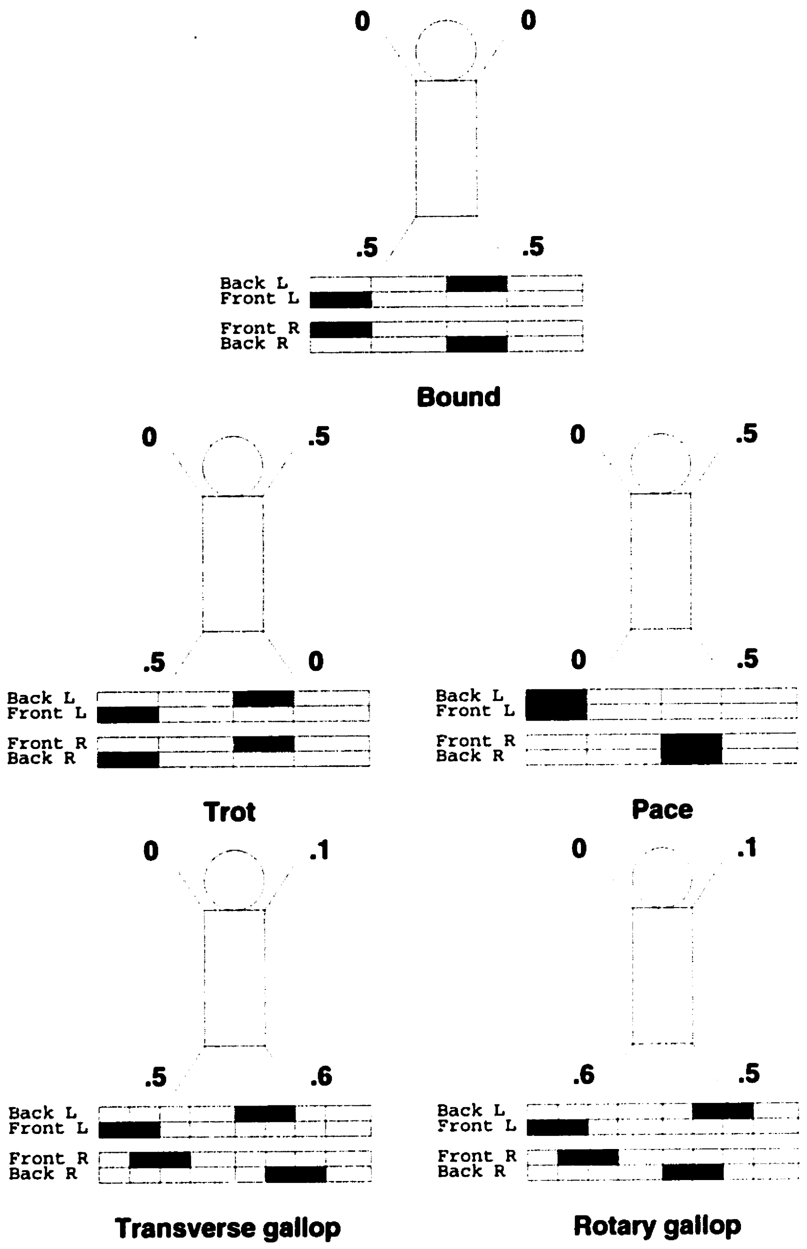

Figure 5-2: Relative phases for quadrupeds in various gaits (Trot, pace, bound, transverse gallop, rotary gallop).

Chapter 6

Monopod Results

The quadruped and biped simulations are based on the stability of the monopod. I have created a simulation and mathematical model of a self-stabilizing monopod. I describe the monopod and its control in section 5.1. There is much leeway in choosing parameters which will achieve a self-stabilizing hopping motion (oscillation frequency, oscillation magnitude, foot radius).

I stabilize the hopping height, first mathematically and then in simulation, and examine how quickly it stabilizes from various initial conditions. Using a stable vertical hopping motion, I then vary the foot radius and map, mathematically and in simulation, the effects the foot radius has on pitch stability. I finally take a stable foot radius, calculate the eigenvalues and eigenvectors, and examine its recovery properties.

6.1 Hopping Height

The hopping height and phase are stable in simulation. I mathematically calculated the same results, as a check on the simulation, and they correspond well. The fact that height and phase are stable indicates that the intuition behind the simplified monopod is probably valid for this more complex version.

The hopping cycle is powered by the leg actuator whose motions are a repeating cycle λ

Start	End	Desired leg length	Equation
0	t_{td}	$l(t) = \mathcal{C}((2 * t/\lambda)^2 - 1)$	$\ddot{z}(t) = -g$
t_{td}	t_{tc}	$l(t) = \mathcal{C}((2 * t/\lambda)^2 - 1)$	$\ddot{z}(t) = -g + \frac{k}{m}(l(t) - z(t)) - \frac{b}{m}\dot{z}(t)$
t_{tc}	t_{to}	$l(t) = -\mathcal{C}\frac{0.5^2-(t/\lambda)+(t/\lambda)^2}{.3^2}$	$\ddot{z}(t) = -g + \frac{k}{m}(l(t) - z(t)) - \frac{b}{m}\dot{z}(t)$
t_{tl}	λ	$l(t) = -\mathcal{C}\frac{0.5^2-(t/\lambda)+(t/\lambda)^2}{.3^2}$	$\ddot{z}(t) = -g$

Table 6.1: The hopping cycle is divided into four sections. The divisions occur at changes in the actuator path and changes in the equations of motion due to touchdown or takeoff.

seconds long. For the mathematical analysis of the hopping gait, I divide the cycle into four sections (Table 6.1). During the first section, from the beginning of the cycle until the time of touchdown ($t = t_{td}$), gravity determines the monopod's motion. From touchdown until halfway through the hopping cycle ($t = t_{tc} = \lambda/2$), the leg spring, damping, and rest length join gravity. This is the second section of the cycle. These forces remain active during the third section as well, except that the path of the spring rest length is different. Finally, from takeoff ($t = t_{to}$) until the end of the cycle gravity is again the only influence on the monopod.

In these equations, $\mathcal{C}$ is the amplitude of the leg length oscillation, k and b are the spring constant and damping coefficient of the leg spring, m is the mass of the body of the monopod (the leg is massless for simplicity), g is the acceleration due to gravity, and λ is the controller cycle time.

Touchdown occurs when $z(t) = l(t)$. Note that z is not the height of the center of mass above the ground; for vertical hopping, the actual height of the center of mass is irrelevant. This will not be the case for pitch analysis. Solving for touchdown in the presence of arbitrary $z(0)$ and $\dot{z}(0)$ gives:

$$t_{td} = \frac{\dot{z}(0) \pm \sqrt{\dot{z}(0) + 2(z(0) + \mathcal{C})(\mathcal{C}8/\lambda^2 + g)}}{8\mathcal{C}/\lambda^2 + g}$$

More math gives a set of equations for $z(t)$ and $\dot{z}(t)$.

$$\begin{array}{rclrcl}
0 & \leq t \leq & t_{td} & z(t) & = & z(t) + \dot{z}(0) * t - \frac{1}{2}gt^2 \\
 & & & \dot{z}(t) & = & \dot{z}(0) - g * t \\
t_{td} & \leq t \leq & t_{tc} & z(t) & = & \mathcal{K}_1 e^{\mathcal{K}_a t} + \mathcal{K}_2 e^{\mathcal{K}_b t} + \mathcal{C}(8(\frac{b}{k\lambda})^2 - 8\frac{bt}{k\lambda^2} - 8\frac{m}{k\lambda^2} + 4(\frac{t}{\lambda})^2 - 1) - g\frac{m}{k} \\
 & & & \dot{z}(t) & = & \mathcal{K}_1\mathcal{K}_a e^{\mathcal{K}_a t} + \mathcal{K}_2\mathcal{K}_b e^{\mathcal{K}_b t} + 8\mathcal{C}\frac{1}{\lambda^2}(t - \frac{b}{k}) \\
t_{tc} & \leq t \leq & t_{to} & z(t) & = & \mathcal{K}_3 e^{\mathcal{K}_a t} + \mathcal{K}_4 e^{\mathcal{K}_b t} - \frac{gm}{k} + \\
 & & & & & \mathcal{C}\frac{1}{.3^2}(\frac{t}{\lambda} - \frac{1}{4} - (\frac{t}{\lambda})^2 + 2\frac{m}{k\lambda^2} + 2\frac{bt}{k\lambda^2} - \frac{b}{k\lambda} - 2(\frac{b}{k\lambda})^2) \\
 & & & \dot{z}(t) & = & \mathcal{K}_3\mathcal{K}_a e^{\mathcal{K}_a t} + \mathcal{K}_4\mathcal{K}_b e^{\mathcal{K}_b t} + \mathcal{C}\frac{1}{.3^2}(\frac{1}{\lambda} - 2\frac{t}{\lambda^2} + 2\frac{b}{k\lambda^2}) \\
t_{to} & \leq t \leq & \lambda & z(t) & = & z(t_{tu}) + \dot{z}(t_{tu})(t - t_{tu}) - \frac{1}{2}g(t - t_{tu})^2 \\
 & & & \dot{z}(t) & = & \dot{z}(t_{tu}) - g(t - t_{tu})
\end{array}$$

The new constants in those expressions must make z and $\dot{z}$ continuous across the changes in equations.

$$\begin{array}{rcl}
\mathcal{K}_a & = & -\frac{b}{2m} + \sqrt{(\frac{b}{2m})^2 - \frac{k}{m}} \\
\mathcal{K}_b & = & -\frac{b}{2m} - \sqrt{(\frac{b}{2m})^2 - \frac{k}{m}} \\
\mathcal{K}_1 & = & \frac{\mathcal{K}_a r_1 - r_2}{(\mathcal{K}_a - \mathcal{K}_b)e^{\mathcal{K}_a t_{td}}} - \frac{r_1}{e^{\mathcal{K}_a t_{td}}} \\
\mathcal{K}_2 & = & \frac{r_2 - \mathcal{K}_a r_1}{(\mathcal{K}_a - \mathcal{K}_b)e^{\mathcal{K}_b t_{td}}} \\
\mathcal{K}_3 & = & \frac{\mathcal{K}_a r_3 - r_4}{(\mathcal{K}_a - \mathcal{K}_b)e^{\mathcal{K}_a t_{tc}}} - \frac{r_3}{e^{\mathcal{K}_a t_{tc}}} \\
\mathcal{K}_4 & = & \frac{r_4 - \mathcal{K}_a r_3}{(\mathcal{K}_a - \mathcal{K}_b)e^{\mathcal{K}_b t_{tc}}} \\
r_1 & = & 8\mathcal{C}(\frac{b}{k\lambda})^2 - z(t_{td}) - 8\mathcal{C}\frac{bt_{td}}{k\lambda^2} - 8\mathcal{C}\frac{m}{k\lambda^2} - \mathcal{C} - \frac{gm}{k} + 4\mathcal{C}(\frac{t_{td}}{\lambda})^2 \\
r_2 & = & -\dot{z}(t_{td}) - 8\mathcal{C}\frac{b}{k\lambda^2} + 8\mathcal{C}\frac{t_{td}}{\lambda^2} \\
r_3 & = & -\frac{gm}{k} + \frac{\mathcal{C}}{.3^2}(2\frac{m}{k\lambda^2} + -2(\frac{b}{k\lambda})^2) - z(t_{tc}) \\
r_4 & = & 2\mathcal{C}\frac{b}{k(.3\lambda)^2} - \dot{z}(t_{tc})
\end{array}$$

The one thing still indeterminate is the time of takeoff, t_{to}. Takeoff occurs when the force between the foot and the ground reaches zero. Since the only other force on the monopod is gravity, takeoff occurs when $\ddot{z}(t_{to}) = -g$. Finding t_{to} requires solving the equation

$$0 = \mathcal{K}_3\mathcal{K}_a^2 e^{\mathcal{K}_a t_{to}} + \mathcal{K}_4\mathcal{K}_b^2 e^{\mathcal{K}_b t_{to}} - 2\mathcal{C}(\frac{\lambda}{.3})^2 + g$$

for t_{to}. I have been unable to solve this equation analytically[1]. I have solved it numerically, allowing a numeric stability analysis.

I have posed the problem with $\mathcal{C}$, k, b, and m as free parameters. However, the solution allows one to eliminate mass from the equation, rewriting it in terms of $\mathcal{C}$, $\frac{k}{m}$, and $\frac{b}{k}$. As a result, one can accommodate additional mass by stiffening the leg and increasing damping. Given $\mathcal{C}$, $\frac{k}{m}$, and $\frac{b}{k}$ I can find a fixed point using Newton's method (see section 4.4.2) and numerically determine the stability of that fixed point.

The physics-based simulation of the monopod corresponds fairly closely to the mathematical model (Figure 6-1). The simulation allows the same type of local stability analysis and calculation of basins of attraction as the mathematical model. The major difference between the mathematical model and the simulated monopod lies in the fact that the simulated foot has mass. Because the foot has mass, the leg actuator does not precisely control the leg length at touchdown (Figure 6-2). The discrepancy shows up here, and becomes more important as the degrees of freedom increase.

I found the instability for both the simulated and the mathematical monopod at a variety of damping values. Figure 6-3 has graphs of the eigenvalues about the reentrant trajectory. If any eigenvalue is greater than one, the monopod is unstable along the associated eigenvector. This graph indicates that as the damping decreases, the vertical hopping becomes less stable. This corresponds with my intuitive explanation (section 3.1.1) of why the simplified monopod's height is stable.

I have experimentally found the monopod's basin of attraction (the set of initial conditions which will eventually reach the stable fixed point). Again, the mathematical model and the simulation correspond well (Figure 6-4). I generated the basin of attraction by

[1]The equation can be solved if there is no damping in the leg motion.

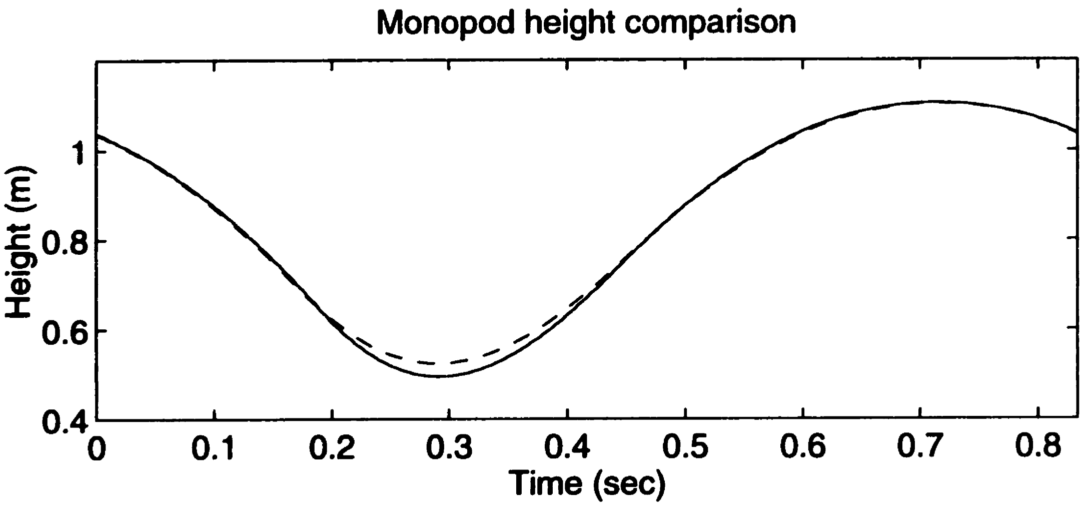

Figure 6-1: Monopod height over time, mathematical model (dashed) and simulated (solid).

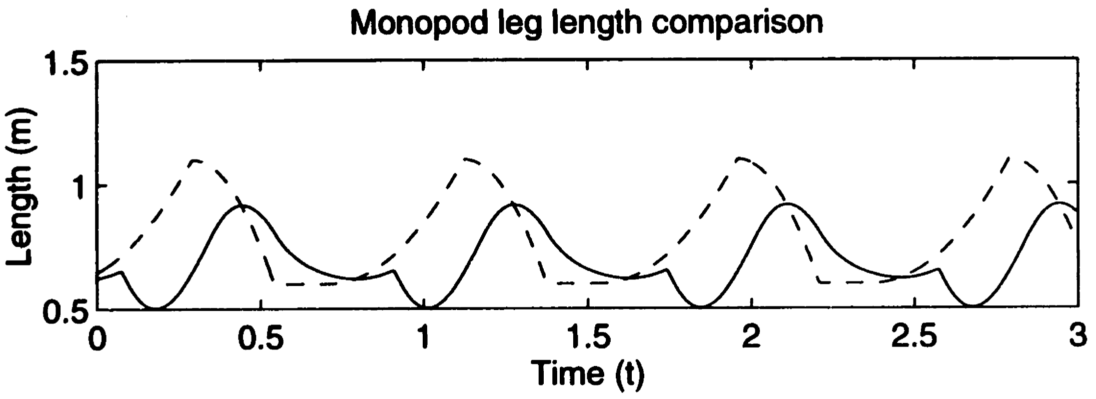

Figure 6-2: Monopod leg length over time, actual (solid) and desired (dashed). This graph uses data from the monopod simulation. The monopod is on the ground from the point at which the actual leg length has a cusp until the point at which the actual and desired leg lengths cross. The difference between actual and desired leg lengths when the monopod is on the ground is expected. Note, however, that when the monopod is in the air there are still significant differences between actual and desired leg lengths. These differences are caused by the fact that the foot has mass and inertia, and are not reflected in the results from the mathematical model of the monopod. The foot mass increases the monopod stability.

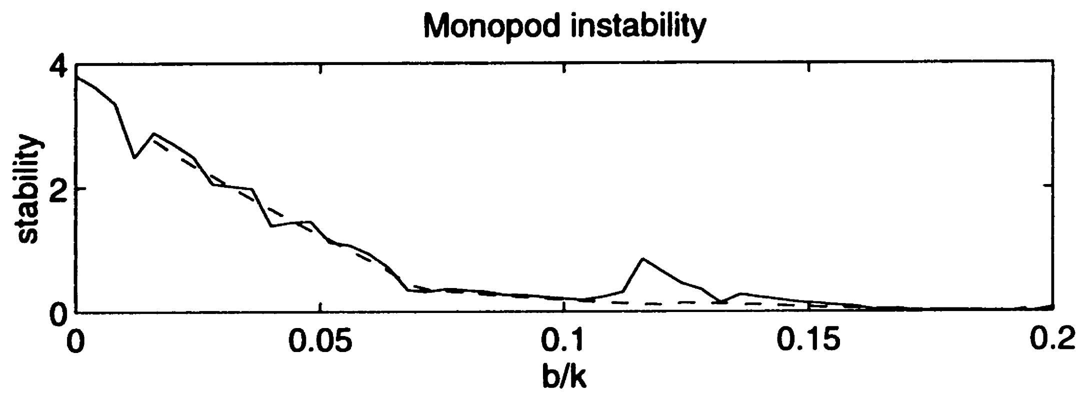

Figure 6-3: Maximum eigenvalue magnitude of the stride delta function for different leg damping values. An eigenvalue with magnitude greater than one indicates that there is an unstable mode (section 4.4.2). This dashed line is data from the mathematical model of the monopod, the solid line is simulated results. At low and high b/k values the mathematical model ceases to have validity; therefore those areas are not graphed. $C = 0.4932$, $b/k = 0.0986$, $\lambda = 0.833$, and $k/m = 100$.

starting simulations, or mathematical models, at various initial conditions and counting how many hops it took before coming close to the stable fixed point. The mathematical monopod has a clear boundary beyond which the mathematical model "fails". This boundary is caused by certain occurrences which the mathematical model cannot handle, such as touchdown occurring more than halfway through the running cycle. These failure modes appear as a clear outer boundary to the basin of attraction. The simulated monopod has similar boundaries, but they are different conditions and happen not to show up as much on the graphs. The sizes of these basins of attraction indicate that the monopod can withstand substantial deviations from the stable vertical hop and still return to it.

As the damping changes, the stable position changes and the basin of attraction changes. I decided to map those basins of attraction, to see if any damping levels had a considerably larger basin of attraction than others. At around $b/k = 1$, the basin of attraction stops expanding as rapidly (Figures 6-5 and 6-6). Increasing the damping further will increase the basin of attraction a small amount. Higher damping means more work done by the

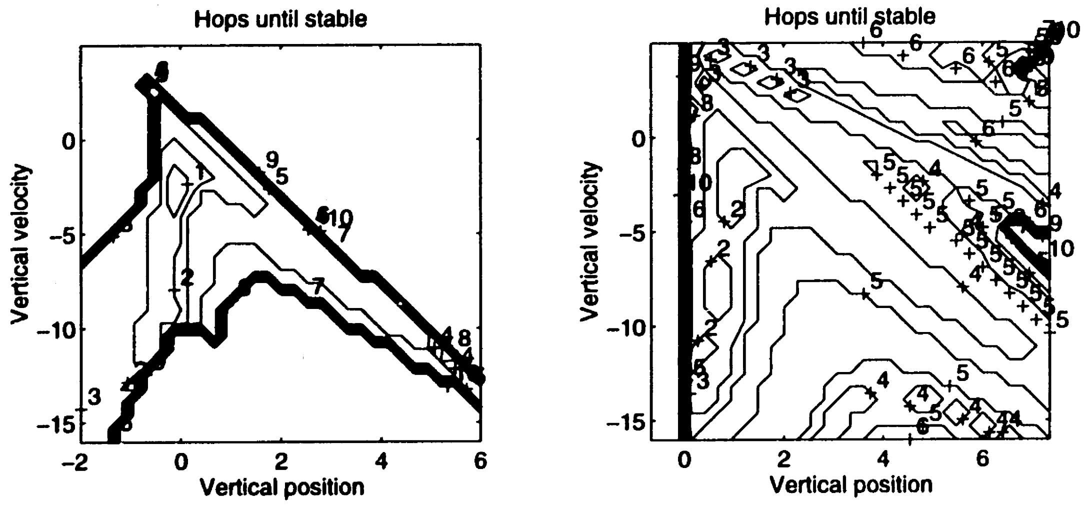

Figure 6-4: Monopod basin of attraction. The basin of attraction is the set of possible starting conditions which will result in stable vertical hopping. These are contour plots, where the graphed value is the number of cycles it will take the monopod to reach a stable vertical hops. The left plot uses data from the mathematical model of the monopod, the right plot uses data from the simulated monopod. These plots are generated by starting simulations at a grid of initial points, running them, and measuring how many hops it takes for them to come close to the stable vertical hop. $C = 0.4932$, $b/k = 0.0986$, $\lambda = 0.833$, and $k/m = 100$.

actuator and less done by the springs.

6.2 Monopod Pitch

The next aspect of the monopod which must be stable is the pitch. A monopod with a point foot cannot hop stably without active control; under some circumstances, neither can one with a flat foot. However, a properly curved foot will enable the monopod to recover form pitch disturbances. Again, I have both a mathematical and a simulated monopod, and a physical monopod which recovers in a manner very similar to the simulation.

The curvature of the foot controls the monopod's pitch. It works much the same way as a wheel with an off-center center of mass: the wheel will roll side to side, until the damping of its rolling motion places the center of mass at its lowest point (Figure 3-5). For the monopod, though, the "wheel" is only on the ground intermittently. If the pitch is strongly corrected while the foot is on the ground, the monopod may take off with a high angular momentum. While the monopod is in the air, it will continue to rotate, and may land with a much greater pitch offset than when it took off. Therefore, if the foot is too flat (the wheel's center of mass is too far from the wheel center) the pitch will not be stable. This intuition is supported by both the mathematical and the simulated solutions (Figure 6-7).

The interaction between the ground and the foot is also important, since the pitch damping is mostly due to the resistance of the foot to rotating on the ground. The interaction between the simulation and the ground is described in section 4.2, and is modeled in the mathematical analysis of the pitch.

The mathematical analysis of the pitch is based on the assumption that the monopod's vertical motion is already stabilized and this motion is effectively undisturbed by pitch errors. The only free parameter for the pitch control is $r(\theta)$, the effective foot radius at a given pitch angle (Figure 6-8). The mathematics indicate that for pitch action similar to a spring–damper system, $r(\theta)$ is effectively constant.

If you assume small pitch angles ($\sin\theta \approx \theta$), the torque applied to the body depends on the vertical force $f(t)$, the height of the center of mass $h(t)$, the current pitch $\theta(t)$, and the

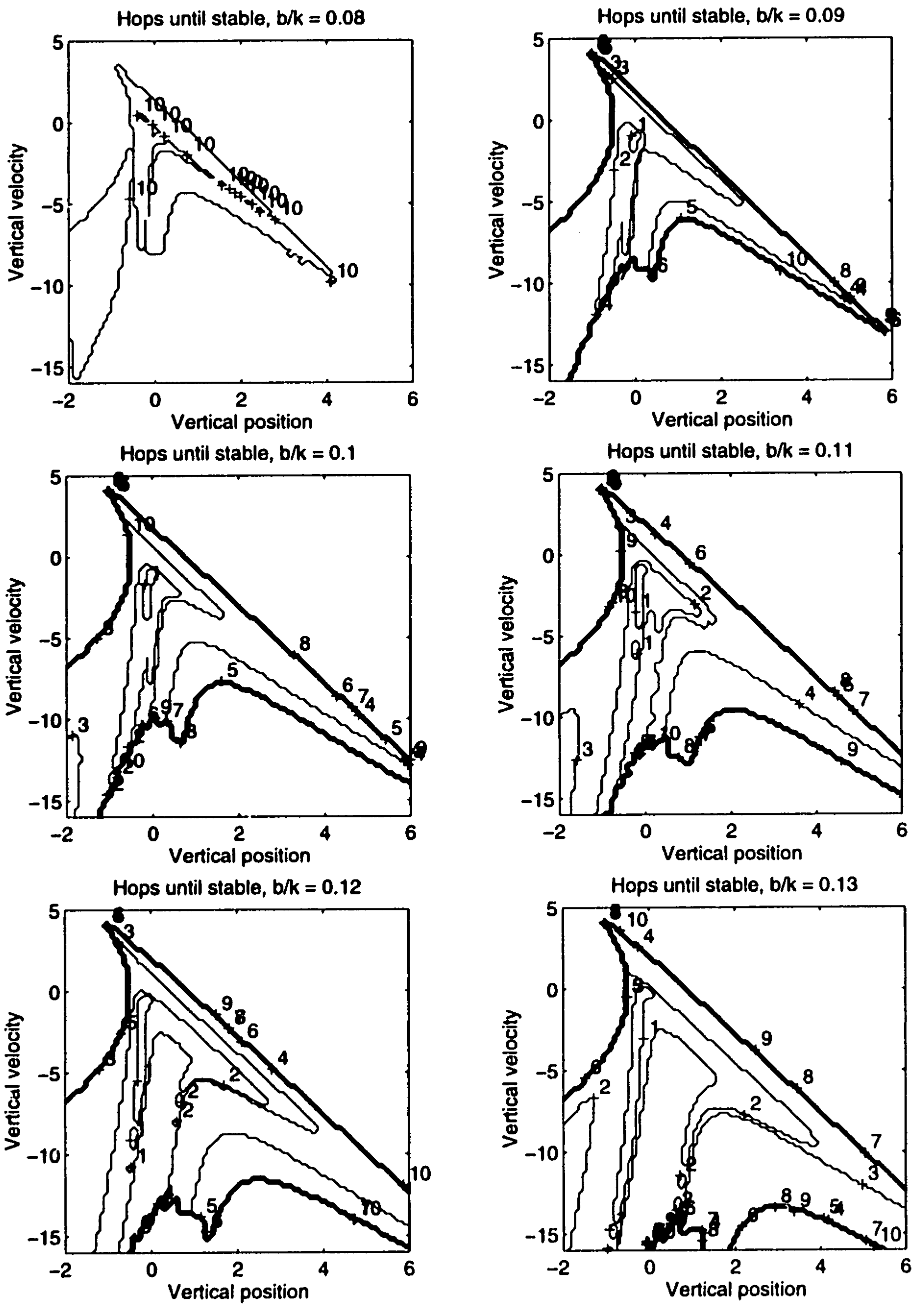

Figure 6-5: Basins of attraction as the damping coefficient increases. These are contour plots, where the graphed value is the number of cycles it will take the monopod to reach a stable vertical hops. These graphs use data from the mathematical model of the monopod, $C = 0.4932$, $b/k = 0.0986$, $\lambda = 0.833$, and $k/m = 100$.

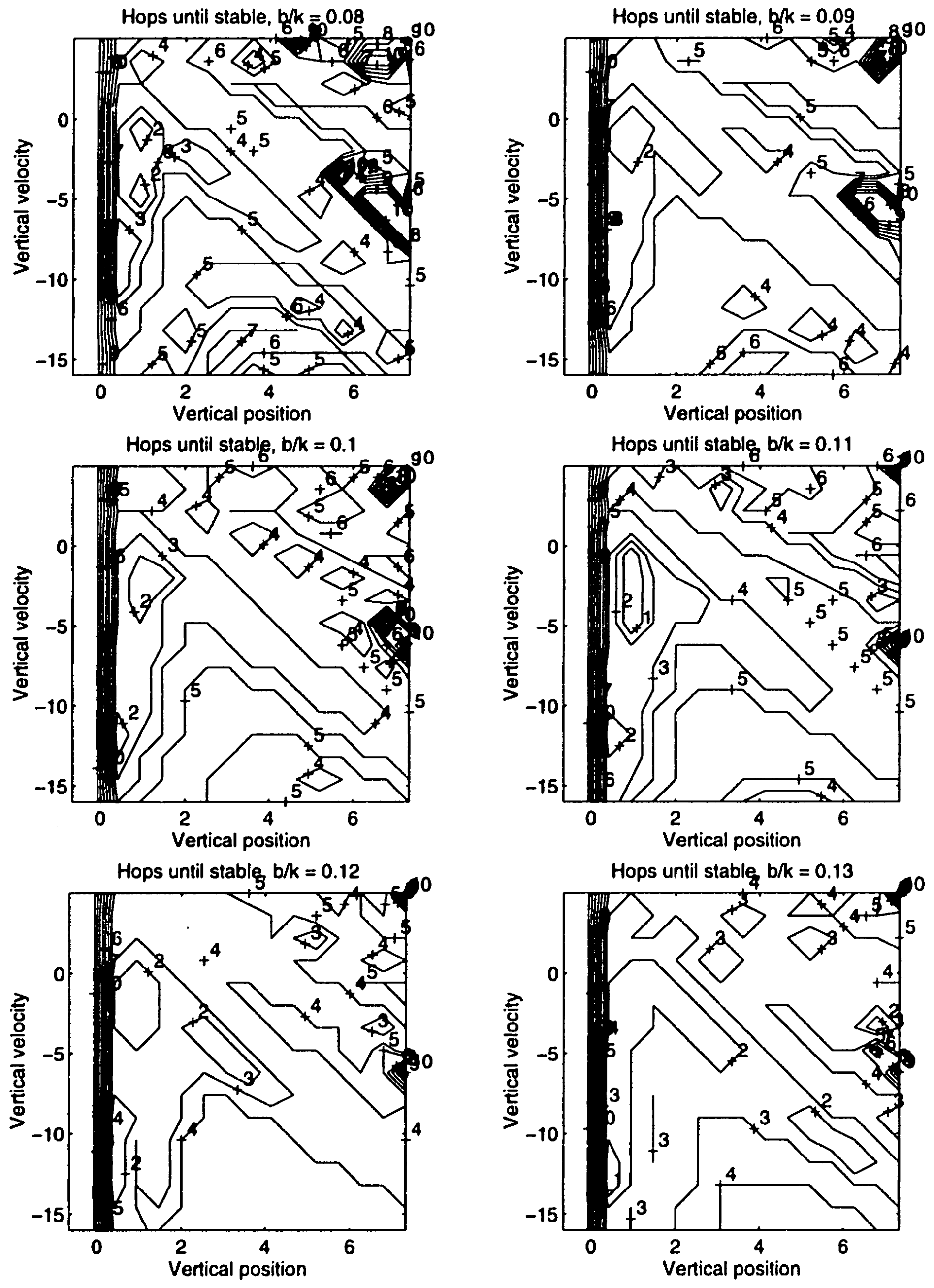

Figure 6-6: Basins of attraction as the damping coefficient increases. These are contour plots, where the graphed value is the number of cycles it will take the monopod to reach a stable vertical hops. These graphs use data from the simulated monopod, $C = 0.4932$, $b/k = 0.0986$, $\lambda = 0.833$, and $k/m = 100$.

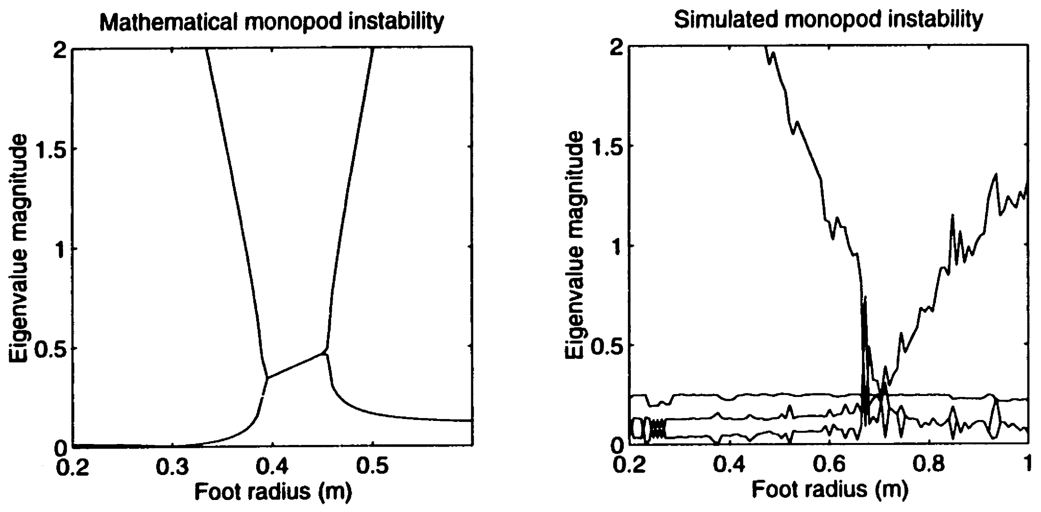

Figure 6-7: Instability of the monopod as the foot radius changes. The mathematical model is on the left, the simulation results are on the right. The bottom axes have different scales to show the similar shape. This is a plot of the eigenvalue magnitudes for the gradient of the stride function about its fixed point, as a function of the foot's radius of curvature. If any eigenvalue has magnitude greater than one, the monopod is unstable for that foot radius and will tip over. The simulation has a larger stable range than the mathematical model.

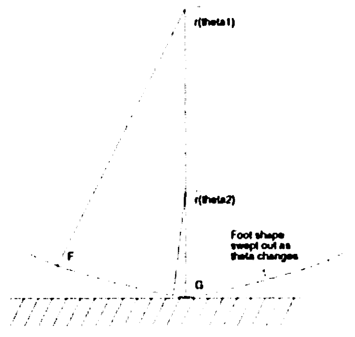

Figure 6-8: The foot shape is determined by a starting angle, an ending angle, and a function $r(\theta)$. Let G be the location of the ground contact if the monopod is vertical. Given θ, find the point F by moving up a distance $r(\theta)$, and then moving down at the angle θ by the same distance. As θ goes from the starting angle to the ending angle, the point F sweeps out the bottom of the foot. This foot formulation simplifies calculating the torque applied when the foot is on the ground.

foot radius for that pitch $r(\theta)$. The height of the body can be taken from the self-stabilizing hopping motion.

As before, the motion must be divided into sections. Since we know information about times of touchdown and takeoff, height, and vertical velocity over time, we can divide it up into four sections: 0 to t_{td}, t_{td} to t_{tc}, t_{tc} to t_{to}, and t_{to} to λ.

For the first and last sections the monopod is in the air. As a result, no external torques can be applied and the only change to momentum is from leg motion. Since we are assuming that the leg is massless, the resulting equations of motion in the pitch direction are

$$\begin{array}{lrcl}
\text{If } t < t_{td} & \theta(t) & = & \theta(0) + t * \dot{\theta}(0) \\
 & \dot{\theta}(t) & = & \dot{\theta}(0) \\
\text{If } t > t_{to} & \theta(t) & = & \theta(t_{to}) + t * \dot{\theta}(t_{to}) \\
 & \dot{\theta}(t) & = & \dot{\theta}(t_{to})
\end{array}$$

The vertical self-stabilizing pattern determines $f(t)$ and $h(t)$, and I choose $r(\theta)$ for high stability. While the foot is on the ground, the ground supplies an external torque $\tau(t)$, and the angular acceleration of the monopod is $\ddot{\theta}(t)$.

$$\begin{aligned}
\tau(t) &= (f(t) + mg)\sin(\theta(t))(h(t) - r(\theta(t))) \\
\ddot{\theta}(t)I(t) &= \tau(t) + b_{th}\dot{\theta}(t) - \dot{\theta}(t)\dot{I}(t) \\
I(t) &= I + mh(t)^2
\end{aligned}$$

$I(t)$ is the inertia about the point on the ground directly below the center of mass; the increase in inertia beyond the monopod's true inertia represents the additional difficulty of accelerating the mass horizontally when rotating a body attached to the ground.

The next step in determining pitch stability is to combine the equations for $h(t)$, $f(t)$, $I(t)$, and $\tau(t)$ to calculate $\theta(t)$. In order to solve for $\theta(t)$ I had to make several approximations. I took Taylor series expansions of $h(t)$ about $t = t_{td}$ and $t = \lambda/2$, approximated $\sin(\theta) = \theta$, set $r(\theta)$ to a constant, and took a series approximation of the solution for the differential equation for $\theta(t)$.

The calculated stride function $\vec{S}$ reasonably approximates the results from simulation. $\vec{S}$ is linear with respect to $\theta(0)$ and $\dot{\theta}(0)$. A low order approximation of the stride function is

$$\vec{S}\begin{pmatrix} \theta(0) \\ \dot{\theta}(0) \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} \theta(0) \\ \dot{\theta}(0) \end{pmatrix}$$

$$\begin{aligned} A_{11} &= 6.4602 - 11.4963r - 5.8742r^2 - 1.1413r^3 + 0.2999r^4 \\ A_{12} &= 1.8294 - 3.1922r - 1.4827r^2 - 0.2716r^3 + 0.0643r^4 \\ A_{21} &= 8.1154 - 19.4769r - 11.4253r^2 - 1.7514r^3 + 0.6966r^4 \\ A_{22} &= 2.3042 - 5.5933r - 2.8563r^2 - 0.4156r^3 + 0.1519r^4 \end{aligned}$$

where r is the foot radius. The stability of this stride function can be determined by looking at the eigenvalues of the matrix. If they have magnitude greater than one the monopod is not stable. The closer to zero the eigenvectors are, the faster the monopod approaches its fixed point (section 4.4.2).

The approximation $\sin(\theta) = \theta$ can be refined a little by rewriting $\tau(t) = (f(t) + mg)(\sin(\theta(t))h(t) - \sin(\theta(t))r(\theta))$ and then setting $r(\theta) = \frac{r\theta}{\sin(\theta)}$ (r is a constant). For angles appropriate for a hopping robot, this refinement is not critical.

The simulation's stability analysis and stable point correspond approximately to the mathematically calculated results, as does the behavior as the radius varies. There are significant differences between the mathematical model and the simulated results. For example, the mathematical model assumes that the actual and desired leg lengths are the same while the monopod is in the air. Figure 6-2 shows that this is not true for the simulation, due to the spring and damping constants in the leg. This changes the touchdown time and the pitch over time (Figure 6-9), which in turn alters the stability for large foot radii.

The simulation's eigenvalues (Table 6.2) indicate that pitch and height have very little interaction. The last eigenvalue pair indicates that hopping height rapidly centers on the stable value, without affecting pitch (the error decreases by over 3/4 every hop). Eigenvalues

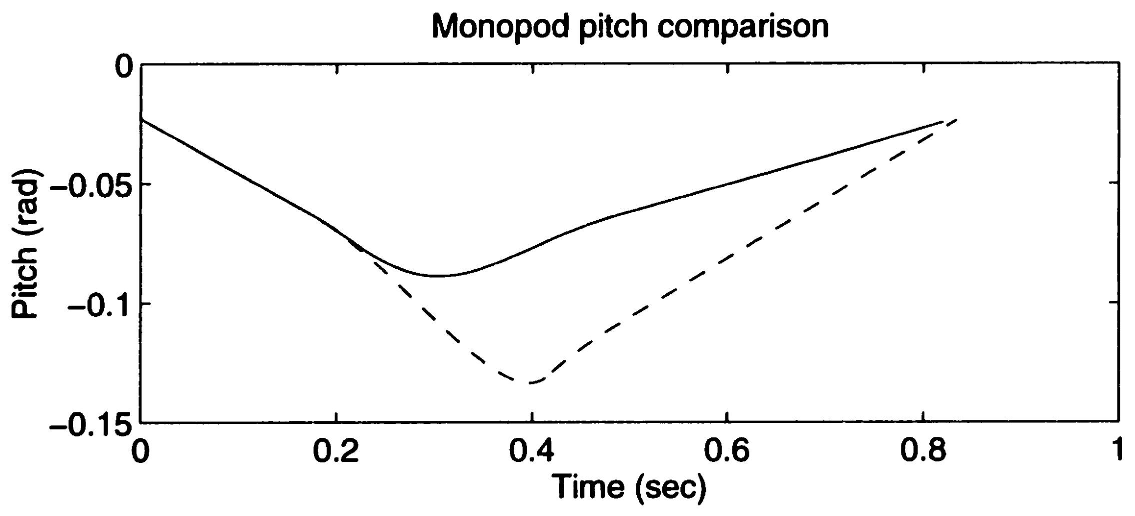

Figure 6-9: Pitch, over time, for a calculated (dashed) and a simulated (solid) monopod. The initial pitch is -0.0228745 radians, and the initial pitch velocity is -0.230141 radians per second. Note the substantial differences in trajectory. I believe they occur because the simulated monopod does not compress the leg as far as the calculated monopod. The different leg lengths result in different inertia with respect to rotation about the point of contact. The simulation's foot radius is 0.7m, while the calculated hopper has a foot radius of 0.445m.

2, 3, and 4 indicate that the errors in pitch and horizontal velocity decrease as well, although not as quickly (decreases by about 1/4 every hop). These eigenvalues also have very small vertical components, so they will not affect the vertical stabilization.

These results indicate that the monopod is stable, in much the same way as the simplified monopod is stable. The eigenvalue results also indicate that hopping height and pitch separate well, as expected. The monopod's stability is important because it is the basic building block for all my other self-stabilizing robots.

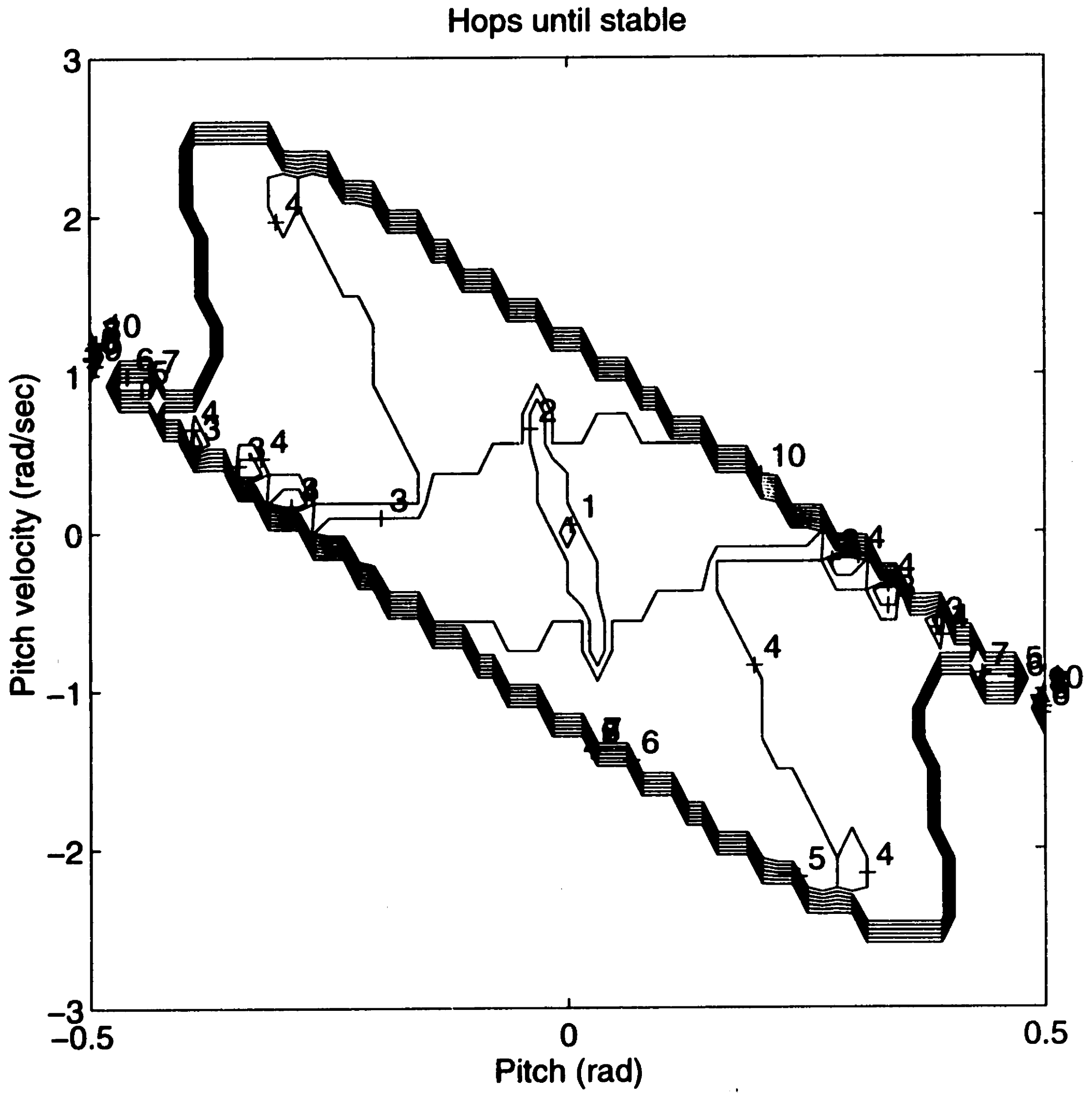

Figure 6-10: Monopod basin of attraction for various pitches and pitch velocities, given stable initial height and height velocity values. The basin of attraction is the set of possible starting conditions which will result in stable vertical hopping. The graphed value is the number of hops before the monopod reaches a stable pitch and pitch velocity. This plot uses data from the simulated monopod.

Eigenvector	1	2	3	4	5-6
Eigenvalue	1.0	−0.6992	0.5310	−0.0318	$0.0650 \pm 0.1990i$
z	0.0	0.0140	0.0048	−0.0097	$-0.0765 \pm 0.1656i$
pitch	0.0	−0.1244	−0.2945	−0.0694	0.0
x	1.0	−0.1225	−0.5627	−0.8682	0.0
z velocity	0.0	0.0153	0.0218	0.0659	$0.6652 \pm 0.7241i$
pitch velocity	0.0	−0.9133	0.5153	0.4607	0.0
x velocity	0.0	−0.3674	−0.5750	0.1571	0.0

Table 6.2: Eigenvectors for a stable simulated hopper. Those eigenvectors with high pitch or pitch velocity values do not have high z or z velocity values, indicating that disturbances in pitch do not readily change into height disturbances. This property is necessary in order to linearize the mathematical model about a stable hopping motion.

Chapter 7

Biped Results

Using the self-stabilizing monopod results, I have created a simulation of a self-stabilizing running biped (Figure 3-8). The biped is a pair of monopods connected by rotating hips to the ends of a pelvis. Again, the biped runs stably in a plane without feedback. I present results from bipeds running in phase and out of phase, including how stability changes with foot radius, disturbance recovery, and typical stable eigenvectors. I also discuss the difficulties with getting the biped to run with other phase differences between the legs and graph the stability at various phases.

The quadruped is formed by two bipeds connected by a back. If the bipeds are stable in the plane perpendicular to the back and the back's joints can keep the bipeds from interfering with each other, the quadruped should be able to stabilize the other dimensions. The bipeds need to be able to run in phase, for bounding, out of phase, for trotting and pacing, and slightly out of phase for quadrupedal gallops.

If the two monopods which make up the biped are nearly in phase, each monopod is supporting approximately half the biped weight. On the other hand, if the monopods are close to out of phase, each monopod supports the entire weight of the biped during stance. This difference would normally necessitate scaling the leg spring constants by the mass. In simulation, the motion is robust enough that the entire point is moot: I was able to consistently use the springs and dampers appropriate for an in phase pair of monopods, even when they were hopping out of phase.

A biped in phase has slightly different characteristics from a biped out of phase, especially in terms of stability and foot radius. For the biped hopping in phase, if the radius is small enough the biped is essentially hopping on two point feet. In this case, although the individual monopods are unstable the hips stabilize the biped. If the radius increases enough, the stability of the individual monopods aids the hips in keeping the biped upright, and it becomes even more stable. Eventually, as the foot becomes flatter, the monopods become even more unstable and overcome the stabilizing effects of the hips. Figure 7-1 shows the eigenvalue magnitudes about the fixed point for the biped hopping in phase. For any foot radius, if any of the graphed values are greater than one, the biped in phase is unstable. Out of phase, each leg touches down separately. The whole hopper is less stable, making it difficult to calculate eigenvalues and plot their magnitude. Too small a foot radius will allow the biped to fall over, and too large a radius will still tip it in the same way as a monopod (Figure 7-2). Again, the largest eigenvalue magnitude is the important one. If the largest eigenvalue magnitude is greater than one, the biped is unstable with that foot radius. I created these two figures by calculating the fixed points and eigenvalues for a large number of different foot radii. Roughness in the graph comes from errors in the fixed point calculation and errors in the gradient calculation. These errors are more pronounced when the simulation is unstable.

In phase, at a stable radius like 0.75m, the biped has reasonably good response to disturbances (Figure 7-3 has an example). The disturbance is in initial height and pitch. Note the extremely quick damping of the hopping height caused by the dampers acting in parallel. This damping is reflected in the small eigenvalue associated with the third eigenvector (Table 7.1). The somewhat slower damping of pitch, and its connection to horizontal motion, corresponds to eigenvalues one and two, which represent a well-damped oscillating pitch and horizontal position combination. The final eigenvalue, and all other eigenvalues, is small enough that its eigenvector is damped out before it has visible effects.

Out of phase, the response to disturbances is still pretty good. Note that when recovering

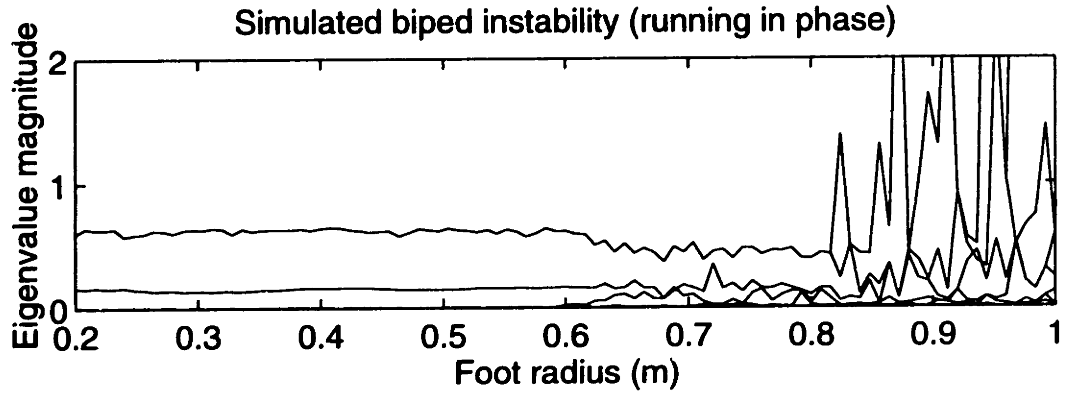

Figure 7-1: Instability of the biped running in phase as its foot radius changes. This is a plot of the eigenvalue magnitudes for the various eigenvectors about the fixed point of the biped's stride function. If any eigenvalue has magnitude greater than one, the biped is unstable for that foot radius and will tip over. When the fixed point is unstable, higher-order effects can result in imprecise eigenvalue measurements, visible in the right quarter of the graph.

from a disturbance (Figure 7-4) the biped hops on each leg separately, so the vertical frequency is twice the actuator frequency. Because the eigenvalue calculations are based on a discretization of the same frequency as the actuator frequency, the period two hopping height appears as a smooth decay in the first and second eigenvalues (Table 7.2).

Because all the eigenvalue magnitudes are less than one, both in phase and out of phase, the biped is stable with a foot radius of 0.75m and can be used in a stable quadruped.

7.1 Phase Problems

I will use the biped as the front or rear half of a quadruped. For a quadruped, complex gaits such as a gallop do not have pairs of legs exactly in or out of phase. Rather, the phase difference is somewhere between 0 and 180 degrees. So far, I have been assuming that hip motions are able to separate the two monopods motions. Unfortunately, when the two monopods are only slightly separated they will destabilize each other by way of the hip connection. Bipedal running in place with arbitrary phase leads to roll stability problems.

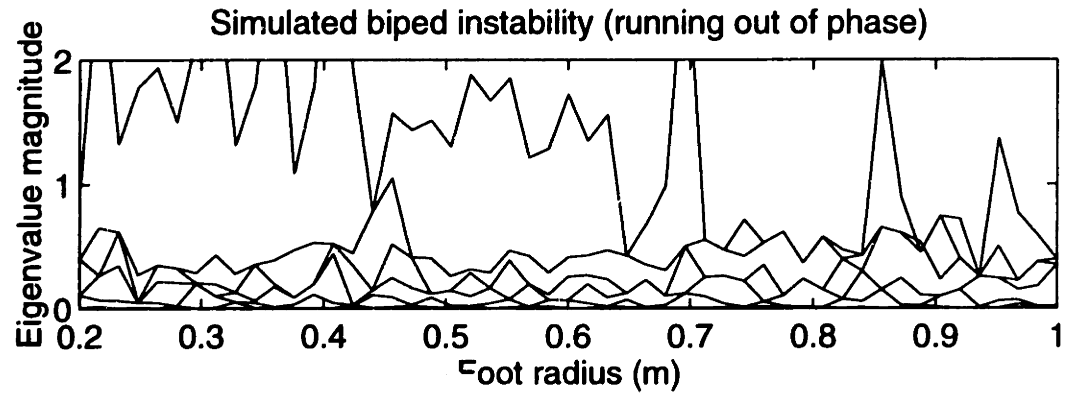

Figure 7-2: Instability of the biped running out of phase as its foot radius changes. This is a plot of the eigenvalue magnitudes for the various eigenvectors about the fixed point of the biped's stride function. If any eigenvalue has magnitude greater than one, the biped is unstable for that foot radius and will tip over. When the fixed point is unstable, higher-order effects can result in imprecise eigenvalue measurements, visible as roughness in the graph. The roughness of this graph is also a little deceiving, as there are not enough data points (they require a fair amount of computation).

Eigenvector	1-2	3	4
Eigenvalue	$-0.1487 \pm 0.3488i$	0.1872	−0.0231
height	$0.0318 \pm 0.0220i$	−0.2775	0.0348
pitch	$-0.2783 \pm 0.0752i$	−0.0000	−0.1253
x position	$-0.4564 \pm 0.1039i$	0.0000	−0.7527
hip 1	0	0	0
hip 2	0	0	0
leg 1	$-0.0013 \pm 0.0021i$	−0.0009	−0.0061
leg 2	$0.0018 \mp 0.0008i$	−0.0009	−0.0003
vertical velocity	$0.1205 \pm 0.1090i$	−0.9606	−0.0411
pitch velocity	$-0.5073 \mp 0.2576i$	−0.0000	0.6378
x velocity	$-0.4457 \mp 0.3827i$	−0.0000	0.0724
hip 1 velocity	$0.0003 \pm 0.0002i$	0.0000	0.0005
hip 2 velocity	$0.0003 \pm 0.0001i$	0.0000	−0.0000
leg 1 velocity	$0.0137 \mp 0.0203i$	0.0098	0.0524
leg 2 velocity	$-0.0185 \pm 0.0100i$	0.0098	−0.0103

Table 7.1: Eigenvectors associated with large eigenvalues, for the biped running with feet in phase. The foot radius is 0.75m.

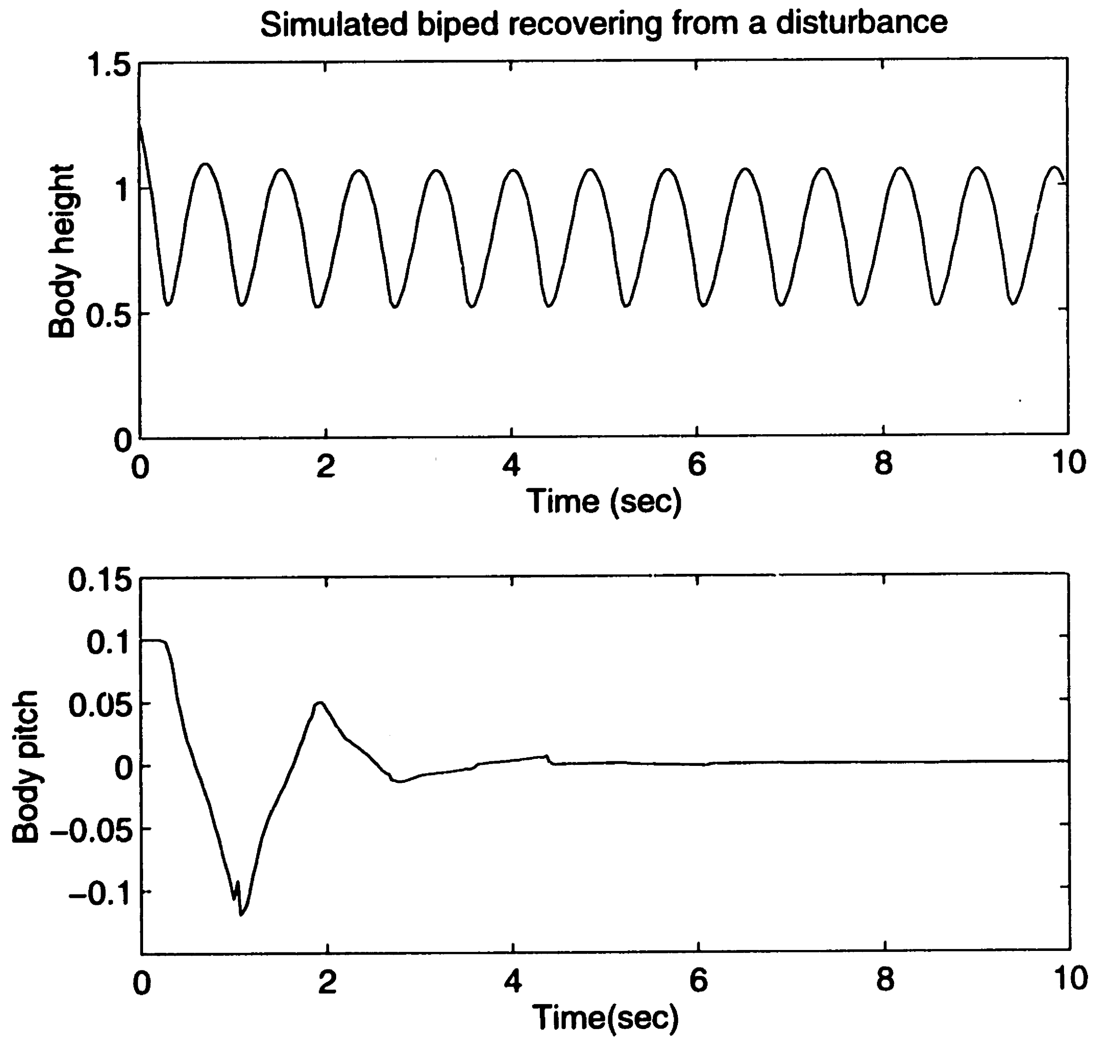

Figure 7-3: Biped recovering from a disturbance in both pitch and vertical position. The disturbance occurs at time $t = 0$. The biped is running *in phase*, with its legs coming down together. Note that, as indicated in the eigenvalues, pitch and height are both damped at approximately the same rate. The foot radius is 0.75m.

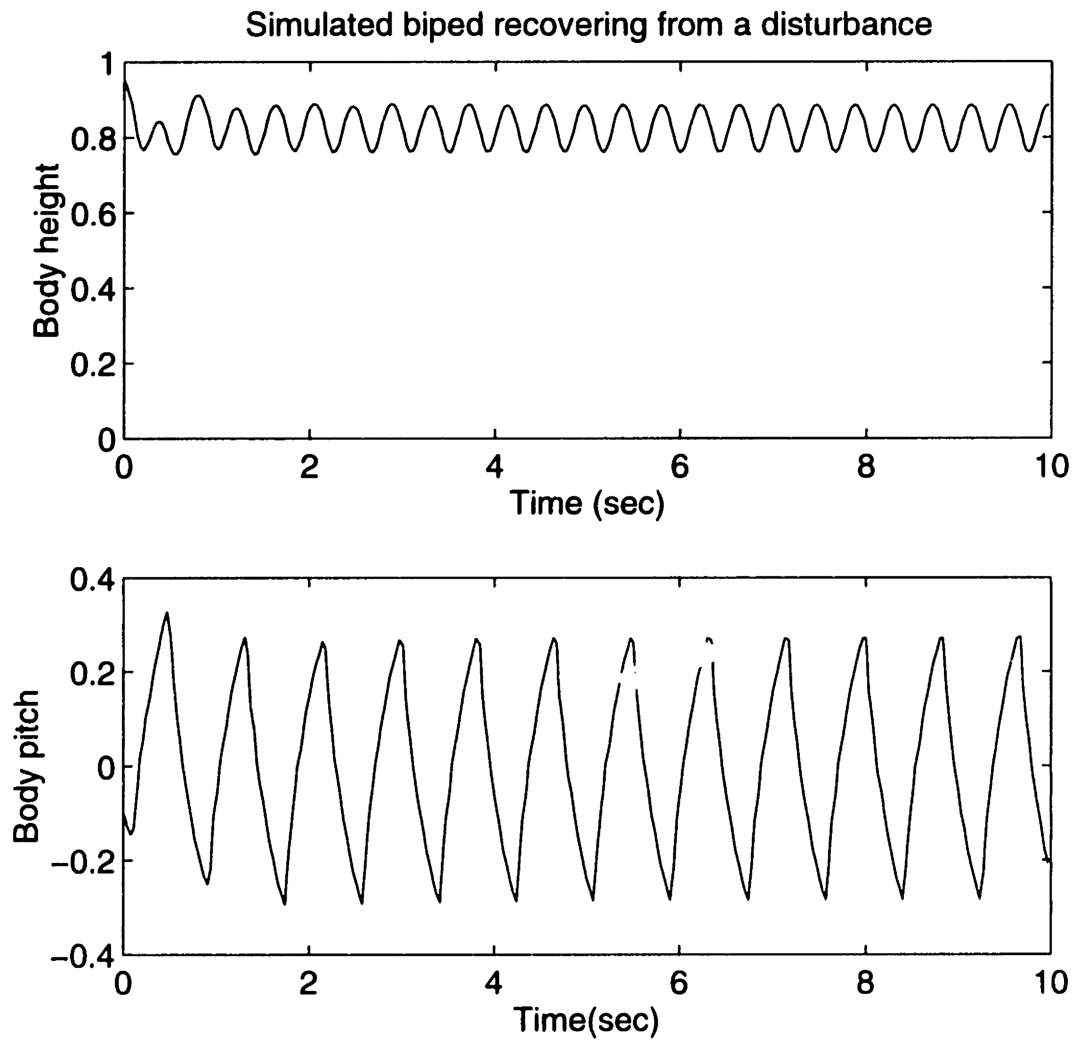

Figure 7-4: Biped recovering from a disturbance in both pitch and vertical position. The disturbance occurs at time $t = 0$. The biped is running *out of phase*, with one leg coming down half a stride later than the other. The foot radius is 0.75m.

Eigenvector	1	2	3-4	5
Eigenvalue	0.5010	0.3308	$0.0687 \pm 0.1445i$	−0.0605
height	−0.1347	0.0872	$0.0100 \mp 0.1501i$	0.0638
pitch	−0.1536	0.0249	$0.2111 \mp 0.1942i$	0.0402
x position	−0.5787	0.6013	$-0.2221 \pm 0.3882i$	−0.0868
hip 1	−0.0005	−0.0003	$-0.0002 \mp 0.0036i$	−0.0017
hip 2	0.0000	0.0003	$0.0000 \pm 0.0014i$	0.0007
leg 1	0.0138	−0.0046	$0.0016 \pm 0.0414i$	−0.0167
leg 2	−0.0007	0.0005	$-0.0009 \mp 0.0017i$	−0.0000
vertical velocity	−0.4831	0.5669	$-0.3181 \pm 0.2559i$	−0.1410
pitch velocity	0.0118	−0.2282	$0.0720 \mp 0.3015i$	−0.9371
x velocity	−0.6068	0.4986	$-0.2090 \mp 0.0160i$	0.0528
hip 1 velocity	0.0299	0.0735	$0.0591 \pm 0.4212i$	0.2339
hip 2 velocity	−0.0058	−0.0235	$-0.0186 \mp 0.1174i$	−0.0745
leg 1 velocity	−0.1432	0.0465	$-0.0152 \mp 0.4365i$	0.1594
leg 2 velocity	0.0076	−0.0029	$0.0102 \pm 0.0262i$	0.0034

Table 7.2: Eigenvectors associated with large eigenvalues, for the biped running with feet out of phase. Other eigenvalues are smaller, or correspond to unimportant positional modes (such as horizontal position). The foot radius is 0.75m.

In the cases where the biped successfully runs with a nonzero phase without tipping over, the stable positions are tilted away from the first foot of the pair to touch down (Figure 7-5). The hips are able to do some compensation for small phase shifts, but for any useful phase shift a more effective solution is to reduce the oscillation magnitude of the first leg to hit the ground.

I characterize the reduced oscillation magnitude with a limping value l for each leg. The limp value is the portion of the leg motion which is removed. A negative value will increase the leg oscillation. The actual path of the leg actuator given limp l is:

$$l(t) = \begin{cases} C((\frac{2*t}{\lambda})^2(1-l)+2l-1) & \text{if } t < \lambda/2 \\ -C(\frac{0.5^2-t/\lambda+(t/\lambda)^2}{.3^2}(1-l)+l) & \text{otherwise} \end{cases}$$

By having the first leg to touch down limp, I can generate a stably running biped with arbitrary phase differences between the legs. Figure 7-6 plots the magnitudes of the eigenvalues for various phases, where 0 is in phase and 0.5 is out of phase. If any of the eigenvalues is greater than one, the biped is unstable. Lower values indicate greater stability. The amount of "limping" is determined automatically by a computerized search.

It is possible that a galloping animal uses the initial foot touchdown to sense the position of the ground for more advanced gait control and hence uses less force; alternately, a galloping animal may use the difference in force requirements to favor one leg or turn more easily. Recently, I also discovered the fact that arbitrarily phased bipedal running can also be self-stabilized if you reduce the base leg length instead of limping, but I have not yes explored that approach.

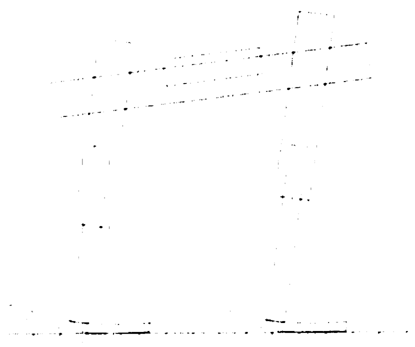

Figure 7-5: When the legs are neither in phase nor out of phase, the stable running position is generally tilted away from the first foot to hit the ground. In this case, the left foot has hit the ground and is in the process of extending, while the right foot has just touched down. I have addressed this problem by decreasing the oscillation in the first leg to touch down, allowing the biped to assume a stable vertical orientation (not shown).

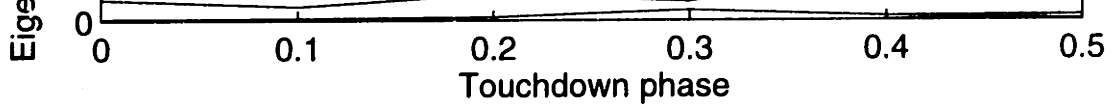

Figure 7-6: Instability (largest eigenvalue magnitude) of the biped running at various leg phases. Phase 0 indicates the legs are hopping together, while phase 0.5 indicates the legs touch down at opposite points in the running cycle. The amount of "limping" required for stable locomotion was determined automatically with a downhill simplex search. If any eigenvalue has magnitude greater than one, the biped is unstable for that phase and will fall over (section 4.4.2).

Chapter 8

Quadruped Results

The self-stabilizing quadruped consists of two self-stabilizing bipeds connected by a back. The back can rotate along the nose-to-tail axis, allowing the two bipeds to rotate independently. The back is long enough that the two bipeds stabilize each other along the back and do not tip over. Reusing the biped control system to control the shift between the front and back pairs of legs improves speed and stability. Because the self-stabilizing biped has some freedom as to the phase between the feet, the self-stabilizing quadruped has the flexibility required to pace, trot, bound, and gallop.

8.1 Quadrupedal Pacing

Pacing is a gait wherein the left side feet touch down and take off, then the right side, with a flight phase in between (Figure 8-1). Some animals, such as camels, pace naturally. Horses can be trained to pace (Gambaryan 1974). A physical robot already has been programmed to pace using Raibert's virtual leg controller (Raibert 1990). The roll stability of a pacing quadruped is fairly difficult to control, since it results in changes in the hopping height, which then change the touchdown and takeoff times. However, increasing the damping in the legs makes the vertical oscillation more stable and increasing the foot radius reduces the roll problems. The ratio of damping to spring constant is approximately twice that used for the other gaits. Table 8.1 has the settings for the leg offsets, radii, and damping which I used to control the pacing quadruped.

The pacing quadruped has little problem correcting errors involving tilting forward or backward, but it has the same side-to-side problems present for the biped hopping out of phase. These problems are visible as it recovers from a disturbance (Figure 8-2), with hopping height varying and roll gradually settling to a steady oscillation. The disturbance, which I artificially introduced into the simulation, is in both roll and hopping height.

The eigenvalues and eigenvectors (Table 8.2) indicate the quadruped's behavior if it deviates from the stable configuration. The first eigenvalue pair indicates that if you disturb the roll, it will take a long time to recover, and oscillate slowly in the process. In figure 8-2 the roll and the height errors both damp out at about the same rate because they are both excitements of the first pair of eigenvectors. The eigenvalue pair 4-5 represents an oscillation about the yaw velocity, which is reasonably damped. Therefore, the yaw velocity will reduce to zero, although the yaw position is not controlled. The final eigenvector is a quickly damped connection between height deviations and roll.

The third eigenvalue is a special case. It exists because there is no yaw control. I have included it because, unlike the horizontal position, the eigenvector does not come out as a clean 1 for yaw angle and 0 everywhere else. This difference comes about because the quadruped has a horizontal velocity when it is in the air. If you yaw the quadruped without also rotating its velocity, the horizontal velocity will induce disturbances.

The quadruped's eigenvalues confirm the behavior implied by its recovery from an impulse. Regardless of how you disturb the quadruped's motion, as long as you do not disturb it beyond its basin of attraction, it will gradually return to a stable running motion.

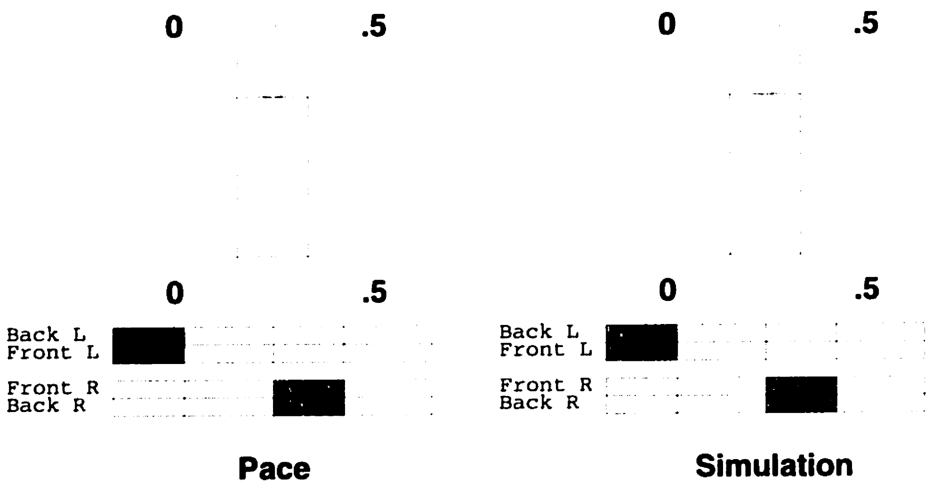

Figure 8-1: Relative phases for the pacing quadruped. Left is ideal, right is the actual result from the simulated quadruped.

Leg 1 limp	0.0
Leg 2 limp	0.0
Leg 3 limp	0.0
Leg 4 limp	0.0
Leg 1 offset	0.0
Leg 2 offset	0.5
Leg 3 offset	0.0
Leg 4 offset	0.5
Foot radius	0.9
Leg b/k	0.2
Leg C	0.4932
Leg k/m	100
λ	0.8333

Table 8.1: Leg limp fractions and timing offsets for the pacing quadruped. The increased foot radius and higher b/k ratio improve stability.

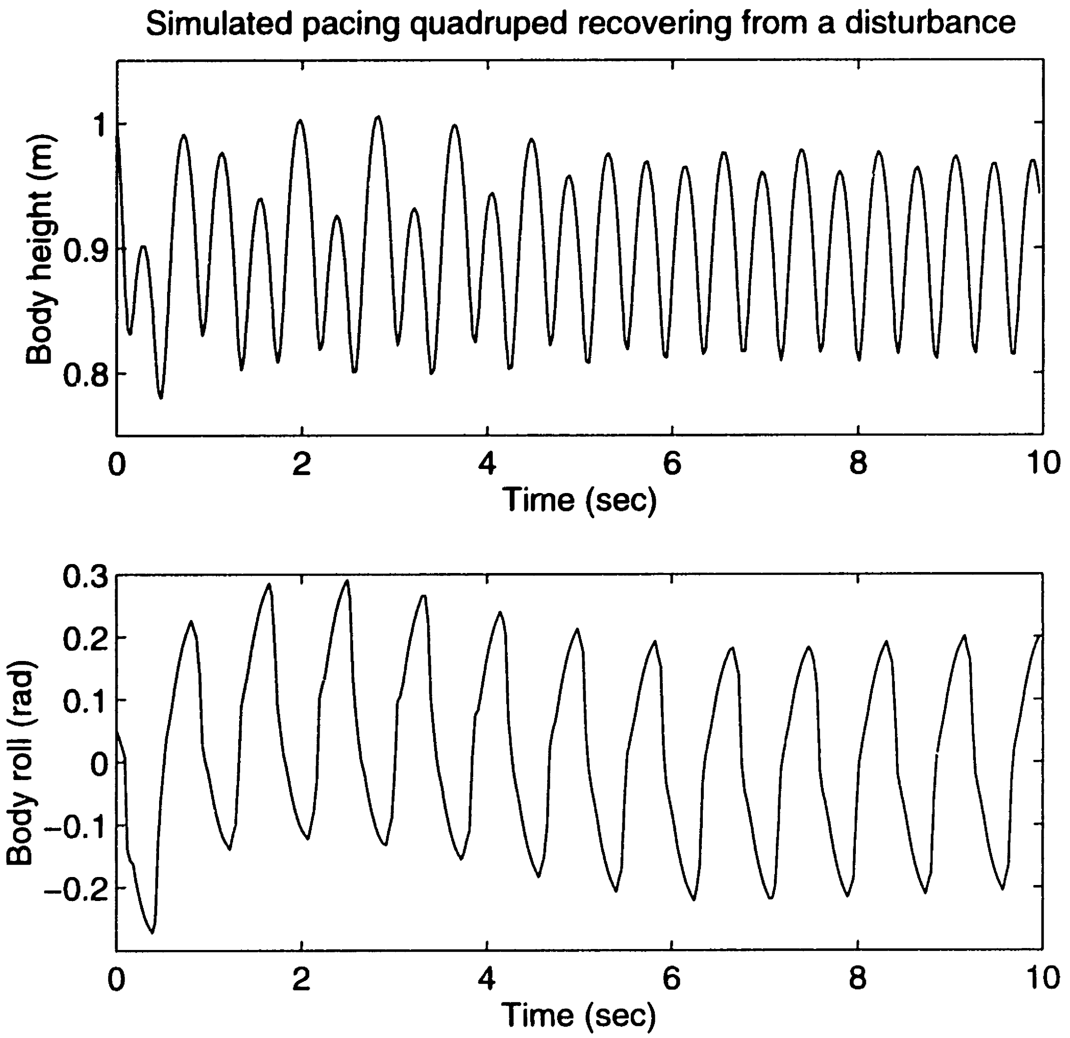

Figure 8-2: Pacing quadruped recovering from a disturbance in both roll and vertical position. The disturbance occurs at time $t = 0$. The foot radius, increased for greater stability, is 0.9m.

Eigenvector	1-2	3	4-5	6
Eigenvalue	$0.7117 \pm 0.4067i$	0.9729	$-0.4589 \pm 0.3394i$	0.0901
x position	$-0.4211 \pm 0.1212i$	0.0080	$0.0000 \pm 0.0002i$	−0.0814
y position	$-0.0005 \pm 0.0001i$	−0.1491	$-0.0565 \mp 0.1431i$	−0.0001
height	$0.0219 \mp 0.2354i$	0.0002	$0.0001 \mp 0.0001i$	0.0163
yaw	$-0.0002 \mp 0.0001i$	−0.9316	$-0.1128 \mp 0.1002i$	−0.0005
pitch	$-0.0000 \pm 0.0000i$	0.0017	$0.0699 \pm 0.0782i$	0.0001
roll	$0.1070 \pm 0.3509i$	−0.0005	$0.0001 \mp 0.0000i$	−0.2203
shoulder	$0.0000 \mp 0.0000i$	−0.0000	$0.0014 \pm 0.0010i$	0.0001
hip 1 forward	$-0.0000 \pm 0.0000i$	0.0000	$-0.0000 \mp 0.0002i$	0.0000
hip 1 side	$0.0000 \mp 0.0002i$	−0.0000	$0.0015 \pm 0.0012i$	−0.0002
leg 1 length	$0.0000 \mp 0.0002i$	0.0000	$0.0000 \mp 0.0000i$	0.0006
hip 2 forward	$-0.0000 \pm 0.0000i$	0.0000	$0.0001 \mp 0.0002i$	0.0000
hip 2 side	$0.0000 \pm 0.0000i$	−0.0001	$0.0009 \pm 0.0007i$	−0.0002
leg 2 length	$-0.0009 \pm 0.0053i$	−0.0000	$0.0002 \mp 0.0050i$	−0.0095
pelvis	$0.0000 \mp 0.0000i$	0.0000	$-0.0014 \mp 0.0010i$	0.0001
hip 3 forward	$-0.0000 \pm 0.0000i$	0.0000	$-0.0000 \mp 0.0002i$	0.0000
hip 3 side	$0.0000 \mp 0.0002i$	0.0000	$-0.0015 \mp 0.0012i$	−0.0002
leg 3 length	$0.0000 \mp 0.0002i$	−0.0000	$-0.0000 \pm 0.0000i$	0.0006
hip 4 forward	$-0.0000 \pm 0.0000i$	0.0000	$0.0001 \mp 0.0002i$	0.0000
hip 4 side	$0.0000 \pm 0.0000i$	0.0001	$-0.0009 \mp 0.0007i$	−0.0002
leg 4 length	$-0.0009 \pm 0.0053i$	−0.0000	$-0.0002 \pm 0.0049i$	−0.0096
x velocity	$0.1391 \pm 0.3713i$	−0.0000	$0.0004 \mp 0.0000i$	0.2952
y velocity	$0.0001 \pm 0.0003i$	−0.3283	$-0.5150 \pm 0.1406i$	0.0000
height velocity	$0.1079 \mp 0.4797i$	0.0004	$0.0002 \mp 0.0001i$	−0.7205
yaw velocity	$-0.0000 \pm 0.0002i$	0.0454	$0.1442 \mp 0.7096i$	0.0022
pitch velocity	$0.0009 \mp 0.0000i$	−0.0006	$0.2938 \pm 0.1167i$	−0.0002
roll velocity	$0.1572 \pm 0.4245i$	−0.0008	$0.0006 \mp 0.0008i$	0.5541
shoulder vel.	$0.0008 \mp 0.0049i$	−0.0015	$-0.0626 \mp 0.0590i$	0.0085
hip 1 forward vel.	$-0.0000 \pm 0.0000i$	0.0003	$0.0001 \mp 0.0057i$	0.0000
hip 1 side vel.	$-0.0039 \pm 0.0169i$	−0.0045	$-0.0087 \mp 0.0332i$	−0.0163
leg 1 length vel.	$-0.0009 \pm 0.0008i$	0.0003	$-0.0053 \mp 0.0024i$	−0.0063
hip 2 forward vel.	$-0.0000 \pm 0.0000i$	0.0001	$-0.0001 \mp 0.0025i$	0.0000
hip 2 side vel.	$0.0108 \mp 0.0573i$	0.0001	$-0.0657 \mp 0.0272i$	0.0795
leg 2 length vel.	$0.0093 \mp 0.0549i$	−0.0004	$0.0033 \pm 0.0541i$	0.0940
pelvis velocity	$0.0009 \mp 0.0050i$	0.0015	$0.0627 \pm 0.0590i$	0.0077
hip 3 forward vel.	$-0.0000 \pm 0.0000i$	0.0003	$0.0001 \mp 0.0057i$	0.0000
hip 3 side vel.	$-0.0039 \pm 0.0168i$	0.0045	$0.0087 \pm 0.0333i$	−0.0166
leg 3 length vel.	$-0.0009 \pm 0.0008i$	−0.0003	$0.0053 \pm 0.0024i$	−0.0063
hip 4 forward vel.	$-0.0000 \pm 0.0000i$	0.0001	$-0.0001 \mp 0.0025i$	0.0000
hip 4 side vel.	$0.0108 \mp 0.0574i$	0.0000	$0.0657 \pm 0.0273i$	0.0788
leg 4 length vel.	$0.0092 \mp 0.0549i$	0.0005	$-0.0032 \mp 0.0541i$	0.0944

Table 8.2: Eigenvectors associated with large eigenvalues, for the pacing quadruped.

8.2 Quadrupedal Bounding

Bounding involves landing with the front feet, taking off, then landing with the back feet (Figure 8-3). Most animals do not bound; they will usually gallop (which does not involve bringing both feet down together). Yaw control is difficult when bounding, because the feet on the ground are located close to each other; this is reflected in the eigenvectors' large yaw components. However, bounding does not have the same damping and radius requirements (Table 8.3) as the pacing quadruped. It can use less damping and a smaller radius and still self-stabilize.

The bounding quadruped can correct roll errors fairly easily, as its feet come together and correct roll in the same way as a biped with feet together can correct the body rotation (chapter 7). This roll stability is evident in the bounding quadruped's rapid recovery from a roll disturbance (Figure 8-4). It is also supported by the fact that eigenvectors 4 and 5, the ones with substantial roll components, converge quickly with eigenvalues which correspond to a 90% reduction in roll error in two steps (Table 8.4).

On the other hand, height errors and their resulting pitch errors are slow to disappear. The corresponding eigenvectors (1 and 2) have an eigenvalue which has magnitude close to one. The height in figure 8-4 shows the slowly damping, oscillating hopping height. Eigenvectors 6 and 7 are also relevant to the pitch velocity; the fact that they damp out quickly indicate that the bounding quadruped reaches a stable hopping phase rapidly. Eigenvector 3 is the yaw eigenvector, with some mixture in the x and y positions and velocities for the same reason as for the pacing quadruped.

Again, the quadruped's eigenvalues confirm the behavior implied by its recovery from an impulse. Regardless of how you disturb the quadruped's motion, as long as you do not disturb it beyond its basin of attraction, it will gradually return to a stable running motion. Some disturbances, such as hopping height, will damp out much slower than others, but any small disturbance will go away over time.

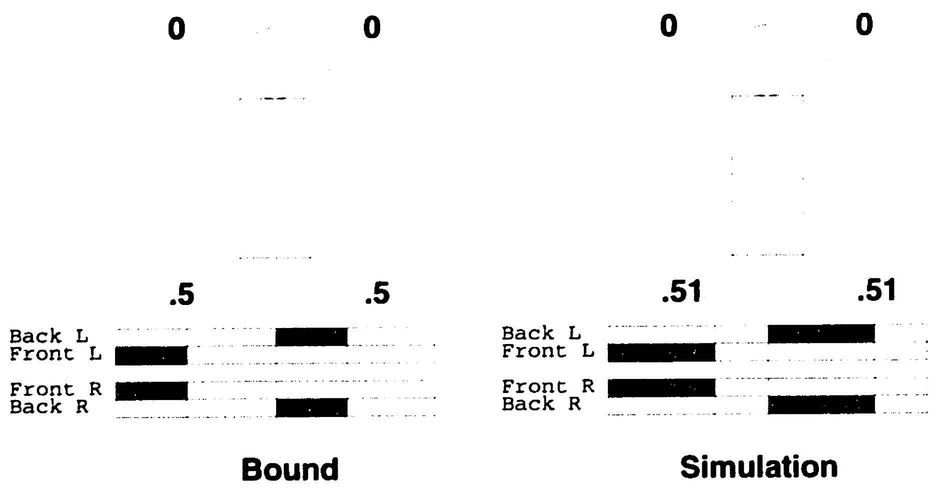

Figure 8-3: Relative phases for the bounding quadruped. Left is ideal, right is the actual result from the simulated quadruped.

Leg 1 limp	0.0
Leg 2 limp	0.0
Leg 3 limp	0.0
Leg 4 limp	0.0
Leg 1 offset	0.0
Leg 2 offset	0.0
Leg 3 offset	0.5
Leg 4 offset	0.5
Foot radius	0.75
Leg b/k	0.0987
Leg C	0.4932
Leg k/m	100
λ	0.8333

Table 8.3: Leg limp fractions and timing offsets for the bounding quadruped.

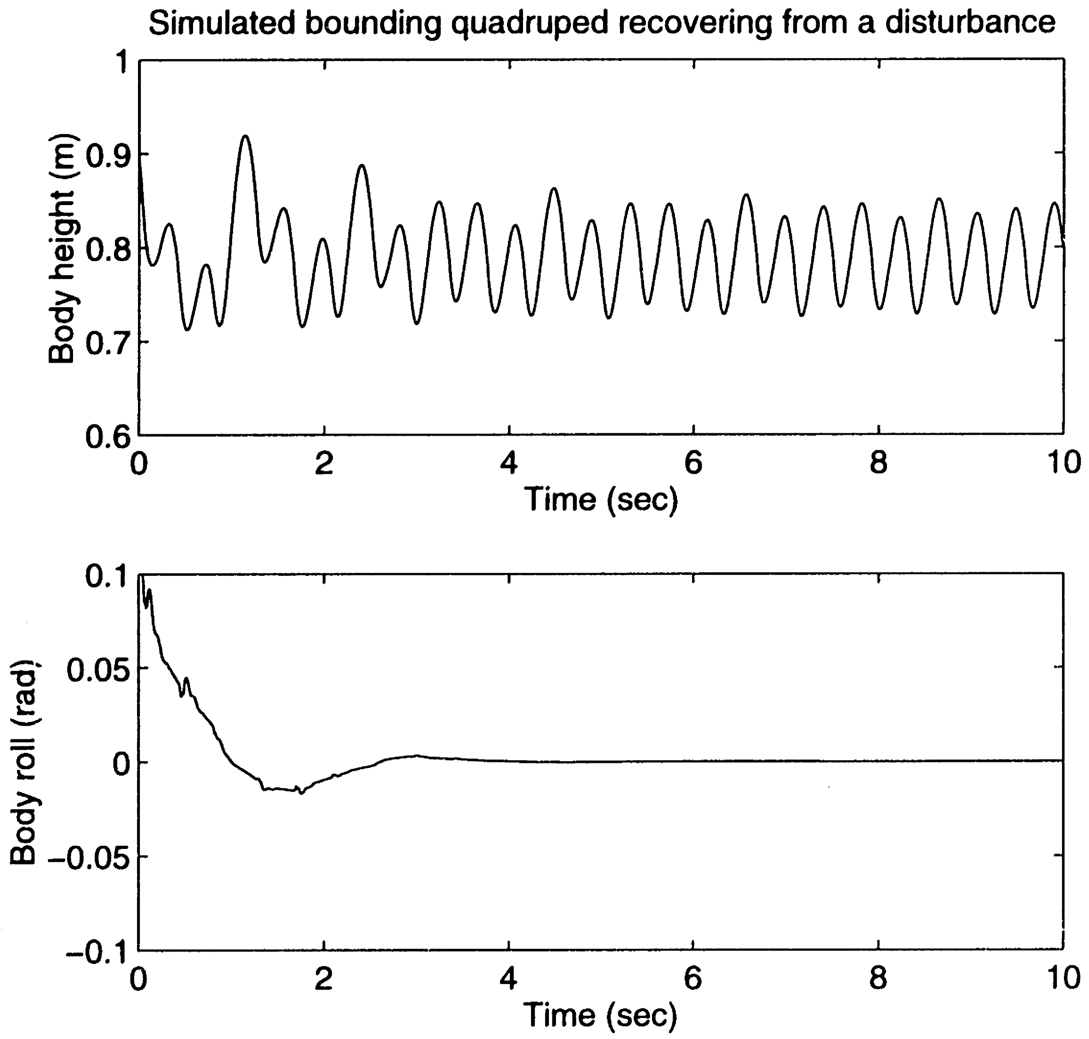

Figure 8-4: Bounding quadruped recovering from a disturbance in both roll and vertical position. The disturbance occurs at time $t = 0$. The foot radius is 0.75m.

Eigenvector	1-2	3	4-5	6-7
Eigenvalue	$0.2233 \pm 0.8949i$	0.9940	$-0.0373 \pm 0.3391i$	$-0.1741 \pm 0.0591i$
x position	$-0.0000 \pm 0.0007i$	0.2551	$0.0490 \mp 0.1584i$	$0.0005 \pm 0.0001i$
y position	$-0.0075 \mp 0.0582i$	-0.0154	$0.0003 \mp 0.0023i$	$-0.0357 \mp 0.0082i$
height	$0.1381 \pm 0.1248i$	-0.0006	$0.0013 \mp 0.0003i$	$-0.0515 \pm 0.0543i$
yaw	$0.0005 \mp 0.0001i$	-0.8259	$-0.0378 \mp 0.0952i$	$-0.0003 \mp 0.0000i$
pitch	$0.1127 \mp 0.0113i$	-0.0007	$0.0006 \mp 0.0004i$	$0.0334 \pm 0.0424i$
roll	$-0.0001 \mp 0.0001i$	-0.0002	$-0.1479 \mp 0.0938i$	$-0.0002 \mp 0.0000i$
shoulder	$0.0000 \mp 0.0000i$	-0.0000	$0.0003 \pm 0.0005i$	$-0.0000 \pm 0.0000i$
hip 1 forward	$0.0001 \pm 0.0001i$	-0.0000	$0.0000 \mp 0.0001i$	$-0.0002 \pm 0.0002i$
hip 1 side	$0.0000 \mp 0.0000i$	0.0000	$0.0001 \pm 0.0003i$	$-0.0000 \pm 0.0000i$
leg 1 length	$0.0003 \pm 0.0010i$	0.0000	$0.0001 \mp 0.0004i$	$-0.0004 \mp 0.0005i$
hip 2 forward	$0.0001 \pm 0.0001i$	0.0000	$0.0000 \pm 0.0001i$	$-0.0002 \pm 0.0002i$
hip 2 side	$0.0000 \mp 0.0000i$	0.0000	$0.0001 \pm 0.0003i$	$-0.0000 \pm 0.0000i$
leg 2 length	$0.0003 \pm 0.0010i$	-0.0000	$-0.0001 \pm 0.0005i$	$-0.0004 \mp 0.0005i$
pelvis	$-0.0000 \pm 0.0000i$	0.0000	$-0.0003 \mp 0.0004i$	$0.0000 \mp 0.0000i$
hip 3 forward	$-0.0011 \mp 0.0029i$	0.0000	$-0.0009 \mp 0.0013i$	$0.0018 \mp 0.0011i$
hip 3 side	$-0.0000 \pm 0.0000i$	0.0000	$-0.0005 \mp 0.0006i$	$0.0000 \mp 0.0000i$
leg 3 length	$-0.0090 \mp 0.0165i$	0.0002	$-0.0082 \mp 0.0118i$	$0.0145 \mp 0.0028i$
hip 4 forward	$-0.0011 \mp 0.0029i$	-0.0000	$0.0009 \pm 0.0012i$	$0.0017 \mp 0.0011i$
hip 4 side	$-0.0000 \pm 0.0000i$	0.0000	$-0.0005 \mp 0.0006i$	$0.0000 \mp 0.0000i$
leg 4 length	$-0.0090 \mp 0.0164i$	-0.0000	$0.0079 \pm 0.0116i$	$0.0146 \mp 0.0028i$
x velocity	$0.0043 \pm 0.0056i$	0.5011	$-0.4032 \pm 0.1332i$	$-0.0033 \mp 0.0002i$
y velocity	$-0.3346 \mp 0.3699i$	0.0080	$-0.0082 \pm 0.0041i$	$0.2182 \pm 0.0156i$
height velocity	$0.1014 \pm 0.1552i$	-0.0008	$0.0016 \pm 0.0006i$	$0.2867 \mp 0.1995i$
yaw velocity	$-0.0002 \pm 0.0006i$	0.0014	$0.2009 \pm 0.5582i$	$0.0007 \pm 0.0001i$
pitch velocity	$-0.2126 \mp 0.7381i$	-0.0029	$-0.0023 \pm 0.0077i$	$0.7325 \mp 0.4806i$
roll velocity	$-0.0004 \pm 0.0006i$	-0.0084	$-0.3365 \pm 0.4657i$	$-0.0000 \pm 0.0000i$
shoulder vel.	$-0.0001 \pm 0.0001i$	0.0012	$-0.0317 \mp 0.0308i$	$0.0004 \mp 0.0001i$
hip 1 forward vel.	$0.0005 \pm 0.0023i$	0.0001	$0.0006 \mp 0.0007i$	$0.0006 \mp 0.0000i$
hip 1 side vel.	$-0.0001 \pm 0.0001i$	0.0005	$-0.0140 \mp 0.0123i$	$0.0002 \mp 0.0001i$
leg 1 length vel.	$-0.0024 \mp 0.0085i$	0.0001	$-0.0048 \mp 0.0010i$	$0.0052 \pm 0.0046i$
hip 2 forward vel.	$0.0005 \pm 0.0023i$	0.0003	$-0.0007 \mp 0.0002i$	$0.0006 \mp 0.0000i$
hip 2 side vel.	$-0.0001 \pm 0.0001i$	0.0005	$-0.0140 \mp 0.0123i$	$0.0002 \mp 0.0001i$
leg 2 length vel.	$-0.0024 \mp 0.0085i$	-0.0000	$0.0046 \pm 0.0008i$	$0.0052 \pm 0.0046i$
pelvis velocity	$0.0001 \mp 0.0002i$	-0.0015	$0.0371 \pm 0.0366i$	$-0.0004 \pm 0.0001i$
hip 3 forward vel.	$-0.0008 \mp 0.0271i$	0.0185	$-0.0676 \mp 0.0109i$	$-0.0483 \pm 0.0193i$
hip 3 side vel.	$0.0002 \mp 0.0002i$	-0.0002	$0.0387 \pm 0.0390i$	$-0.0005 \pm 0.0002i$
leg 3 length vel.	$0.0925 \pm 0.1679i$	-0.0015	$0.0858 \pm 0.1254i$	$-0.1495 \pm 0.0291i$
hip 4 forward vel.	$0.0002 \mp 0.0283i$	-0.0308	$0.0731 \pm 0.0403i$	$-0.0514 \pm 0.0204i$
hip 4 side vel.	$0.0002 \mp 0.0002i$	-0.0007	$0.0389 \pm 0.0401i$	$-0.0006 \pm 0.0002i$
leg 4 length vel.	$0.0925 \pm 0.1678i$	-0.0001	$-0.0828 \mp 0.1232i$	$-0.1496 \pm 0.0288i$

Table 8.4: Eigenvectors associated with large eigenvalues, for the bounding quadruped.

8.3 Quadrupedal Trotting

Trotting is like pacing, except that the animal uses diagonal pairs of legs instead of each side. The front left and back right feet touch down and take off, then the front right and back left feet do the same (Figure 8-5). Trotting seems to be the most stable of the quadrupedal gaits which I have tried. The feet come down together, far enough apart to help control orientation, as with pacing. They also come down positioned to support the center of mass, unlike pacing or bounding. This stability means that no extra damping or foot radius is necessary (Table 8.5).

The trotting quadruped corrects height and roll errors quickly and smoothly (Figure 8-6). The height correction corresponds with the third largest eigenvalue (Table 8.6), which is essentially a decaying hopping height and vertical velocity. The time trace of vertical position looks an oscillating maximum height, but the eigenvectors and eigenvalues correspond to samples every 0.833 seconds. Therefore, only the higher of the pairs of peaks are seen by the eigenvalue and eigenvector computation, so the eigenvalue is not imaginary. The first two eigenvectors correspond to an exchange between roll, pitch velocity, and horizontal motion. Eigenvectors 4 and 5 correspond to a similar exchange between roll velocity, pitch velocity, and horizontal motion. Both of these eigenvectors result from the fact that if the robot moves over a diagonal pair of legs it will both pitch and roll. Finally, the sixth eigenvector shows that the trotting quadruped is fairly good at controlling its yawing motions.

Again, the quadruped's eigenvalues confirm the behavior implied by its recovery from an impulse. The running motion is stable.

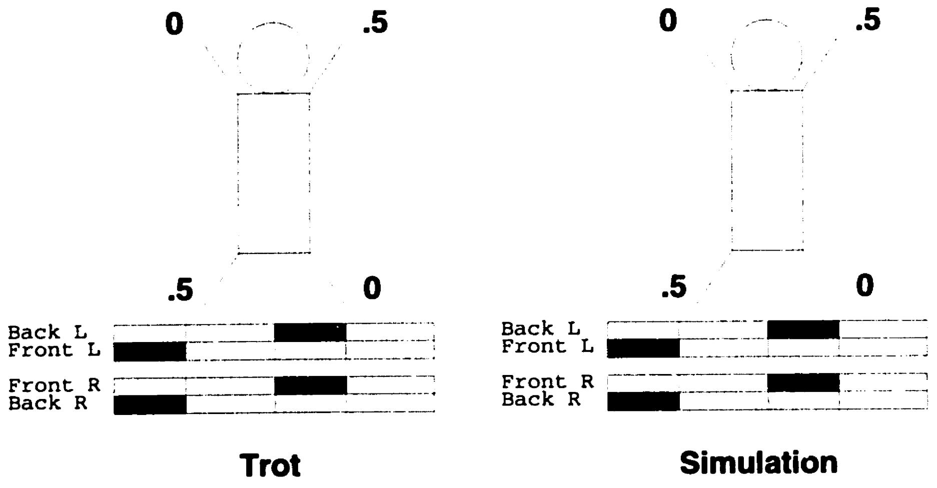

Figure 8-5: Relative phases for the trotting quadruped. Left is ideal, right is the actual result from the simulated quadruped.

Leg 1 limp	0.0
Leg 2 limp	0.0
Leg 3 limp	0.0
Leg 4 limp	0.0
Leg 1 offset	0.5
Leg 2 offset	0.0
Leg 3 offset	0.0
Leg 4 offset	0.5
Foot radius	0.413
Leg b/k	0.0987
Leg C	0.4932
Leg k/m	100
λ	0.8333

Table 8.5: Leg limp fractions, timing offsets, and other parameters for the trotting quadruped.

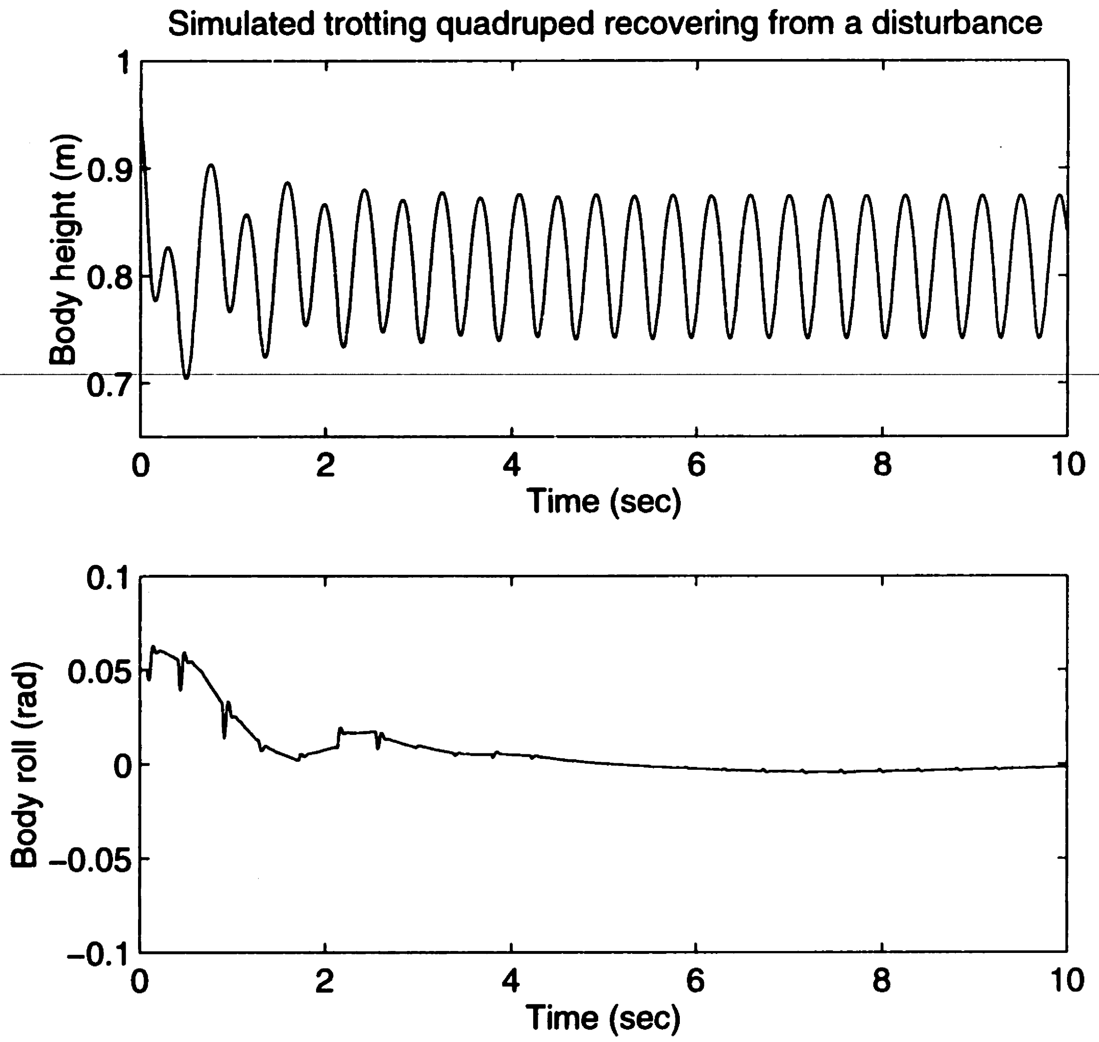

Figure 8-6: Trotting quadruped recovering from a disturbance in both roll and vertical position. The disturbance occurs at time $t = 0$. The foot radius is 0.75m.

Eigenvector	1-2	3	4-5	6
Eigenvalue	$0.0503 \pm 0.9228i$	0.4861	$-0.3374 \pm 0.0494i$	0.2587
x position	$0.2407 \mp 0.0352i$	-0.0018	$-0.0317 \pm 0.0080i$	0.0029
y position	$0.0192 \mp 0.0329i$	0.0004	$0.0406 \mp 0.1148i$	-0.0002
height	$0.0000 \pm 0.0000i$	0.2114	$0.0000 \pm 0.0000i$	0.0010
yaw	$-0.0007 \mp 0.0006i$	0.1178	$0.0007 \mp 0.0005i$	0.0019
pitch	$-0.0541 \pm 0.0384i$	0.0001	$-0.0051 \pm 0.1455i$	0.0006
roll	$0.2715 \mp 0.0732i$	-0.0017	$-0.0506 \pm 0.0223i$	-0.0020
shoulder	$-0.0000 \pm 0.0000i$	-0.0003	$-0.0000 \mp 0.0000i$	0.0007
hip 1 forward	$0.0001 \mp 0.0000i$	-0.0000	$-0.0002 \mp 0.0000i$	-0.0000
hip 1 side	$-0.0000 \pm 0.0001i$	-0.0003	$0.0000 \pm 0.0001i$	0.0002
leg 1 length	$0.0002 \pm 0.0002i$	0.0003	$0.0002 \pm 0.0004i$	-0.0000
hip 2 forward	$-0.0001 \pm 0.0000i$	0.0000	$0.0001 \mp 0.0000i$	-0.0000
hip 2 side	$-0.0000 \mp 0.0001i$	-0.0002	$0.0000 \mp 0.0001i$	0.0007
leg 2 length	$-0.0125 \mp 0.0043i$	-0.0070	$0.0122 \mp 0.0203i$	0.0000
pelvis	$0.0000 \mp 0.0000i$	0.0003	$-0.0000 \pm 0.0001i$	-0.0007
hip 3 forward	$-0.0001 \pm 0.0000i$	0.0000	$0.0002 \pm 0.0000i$	0.0000
hip 3 side	$0.0000 \mp 0.0001i$	0.0002	$-0.0000 \mp 0.0001i$	-0.0007
leg 3 length	$0.0125 \pm 0.0043i$	-0.0069	$-0.0122 \pm 0.0204i$	-0.0005
hip 4 forward	$0.0001 \mp 0.0000i$	-0.0000	$-0.0001 \pm 0.0000i$	0.0000
hip 4 side	$0.0001 \pm 0.0001i$	0.0004	$0.0001 \pm 0.0002i$	-0.0002
leg 4 length	$-0.0001 \mp 0.0002i$	0.0002	$-0.0001 \mp 0.0004i$	0.0000
x velocity	$0.0260 \mp 0.4582i$	0.0016	$-0.0449 \mp 0.3129i$	-0.0064
y velocity	$0.4262 \pm 0.0074i$	-0.0008	$-0.3925 \mp 0.0960i$	-0.0039
height velocity	$0.0001 \pm 0.0002i$	0.9646	$0.0011 \pm 0.0002i$	-0.0098
yaw velocity	$0.0033 \mp 0.0011i$	0.0233	$0.0016 \pm 0.0033i$	-0.9414
pitch velocity	$-0.3718 \pm 0.0426i$	-0.0011	$0.5486 \pm 0.0317i$	0.0025
roll velocity	$0.0022 \mp 0.5345i$	-0.0063	$-0.2442 \mp 0.4611i$	0.0206
shoulder vel.	$0.0021 \mp 0.0005i$	0.0162	$0.0018 \pm 0.0025i$	0.1528
hip 1 forward vel.	$0.0001 \mp 0.0000i$	0.0001	$-0.0002 \mp 0.0000i$	-0.0013
hip 1 side vel.	$-0.0040 \mp 0.0023i$	0.0010	$0.0053 \mp 0.0088i$	0.1618
leg 1 length vel.	$-0.0015 \mp 0.0019i$	-0.0004	$-0.0021 \mp 0.0040i$	-0.0066
hip 2 forward vel.	$0.0009 \mp 0.0001i$	-0.0001	$-0.0013 \pm 0.0001i$	0.0009
hip 2 side vel.	$0.0091 \pm 0.0031i$	0.0145	$-0.0039 \pm 0.0185i$	0.0831
leg 2 length vel.	$0.1287 \pm 0.0445i$	0.0683	$-0.1266 \pm 0.2099i$	0.0068
pelvis velocity	$-0.0018 \pm 0.0007i$	-0.0163	$-0.0019 \mp 0.0013i$	-0.1529
hip 3 forward vel.	$0.0009 \mp 0.0001i$	0.0001	$-0.0013 \pm 0.0001i$	-0.0009
hip 3 side vel.	$0.0054 \pm 0.0038i$	-0.0148	$-0.0067 \pm 0.0140i$	-0.0828
leg 3 length vel.	$-0.1291 \mp 0.0445i$	0.0683	$0.1261 \mp 0.2108i$	0.0106
hip 4 forward vel.	$0.0001 \mp 0.0000i$	-0.0001	$-0.0001 \mp 0.0001i$	0.0013
hip 4 side vel.	$-0.0051 \mp 0.0011i$	-0.0011	$0.0031 \mp 0.0082i$	-0.1621
leg 4 length vel.	$0.0019 \pm 0.0019i$	-0.0003	$0.0024 \pm 0.0048i$	-0.0066

Table 8.6: Eigenvectors associated with large eigenvalues, for the trotting quadruped.

8.4 Quadrupedal Gallop

Galloping is the most complex of the gaits which I have attempted, since the feet neither touch down nor take off at the same times. It has not previously been achieved, neither in reality nor in realistic simulation. Gallops come in two types, rotary gallops and transverse gallops. The front and back feet come down and take off approximately in pairs. In a rotary gallop the first back foot to touch down is diagonally across the body from the first front to touch down (Figure 8-7), while in a transverse gallop the first back foot to touch down is on the same side as the first front foot to touch down (Figure 8-8).

I have simulated self-stabilizing gallops, both rotary and transverse. Neither gait requires exceptional damping or foot radii (Tables 8.7 and 8.8). Both gallops have a short flight time, which makes it difficult to accurately calculate the eigenvalues and eigenvectors.

The rotary gallop turns is the less stable of the two, in my experiments. It has a tendency to gallop in circles (Figure 8-9). This tendency makes sense if you consider the fact that a self-stabilizing biped running slightly out of phase has a slight tendency to creep sideways. In the rotary gallop, the front biped creeps one way, the rear creeps the other, and the net result is a spin. Numerical errors caused by the tendency to spin and the fact that there an extremely short flight time[1] alter the eigenvectors and eigenvalues in table 8.9, but some information remains.

For the rotary gallop, eigenvalues 1 and 2 (Table 8.9) are very close to singular. That is, their eigenvalues are close to one, and therefore their eigenvectors will not readily damp out. I believe their eigenvectors correspond to the gradually decreasing yaw, and its effects on the other state variables. Eigenvectors 3 and 4 correspond to the slight oscillation about yaw visible in the recovery graph. Eigenvectors 5 and 6 are height and pitch interaction; I believe that with slightly different parameters their eigenvalues may merge into a single pair of complex conjugates.

[1]A short flight time causes numerical errors in two ways. First, a short flight time means that the feet are close to the ground. Altering parameters, such as roll or height, can result in accidental ground contact which can significantly alter the simulation results. Second, a short flight time means that at any time when the quadruped is in the air one leg will have transients from its recent departure from the ground.

The transverse gallop has less flight time than the rotary gallop, and when running stably (Figure 8-11) its eigenvectors have enough numerical error to make their results difficult to use. Most of the numerical error is because the time I designate the beginning of the cycle for finding the A matrix occurs when leg 2 has just lifted off the ground. Each of the eigenvectors in table 8.10 has a large hip 2 forward velocity component, even eigenvector 1 (one of the quadruped's horizontal position vectors, which should have no components except x and/or y position). Experience with the other quadrupeds indicates that the second hip's forward velocity is not very important to the total stability, so that portion of the eigenvectors is incorrect.

The transverse gallop's second and third eigenvalues (Table 8.10) may indicate that the yaw velocity is rapidly damped, and the fourth and fifth may mean the same for pitch velocity. The sixth eigenvector can be neglected due to numerical error, as it is mostly in the hip 2 forward velocity. The first eigenvalue is one of the quadruped's horizontal position vectors, which should have no components except x and/or y position. Other values in that eigenvector are indicative of numerical errors. Even though numerical evaluation of the transverse gallop does not show stability, this gait recovers well from a variety of disturbances (Figure 8-12).

The gallop's eigenvalues are too noisy to absolutely confirm the local stability of the galloping gaits, although they are indicative of the major modes of oscillation. I have disturbed the galloping quadruped along the least stable eigenvectors and had them damp out. Both the rotary and transverse gallop spin about yaw, although the rotary gallop's spin is much more pronounced. This spin occurs because the out-of-phase bipeds making the quadruped creep to the side slightly. For the rotary gallop, the bipeds creep in opposite directions, resulting in a fairly quick spin. For the transverse gallop, they creep the same direction, although with different magnitudes, resulting in a somewhat slower circling motion.

Regardless of how you disturb the quadruped's motion, as long as you do not disturb it beyond its basin of attraction, it will gradually return to a stable running motion. Some disturbances, such as hopping height, will damp out much slower than others, but any small

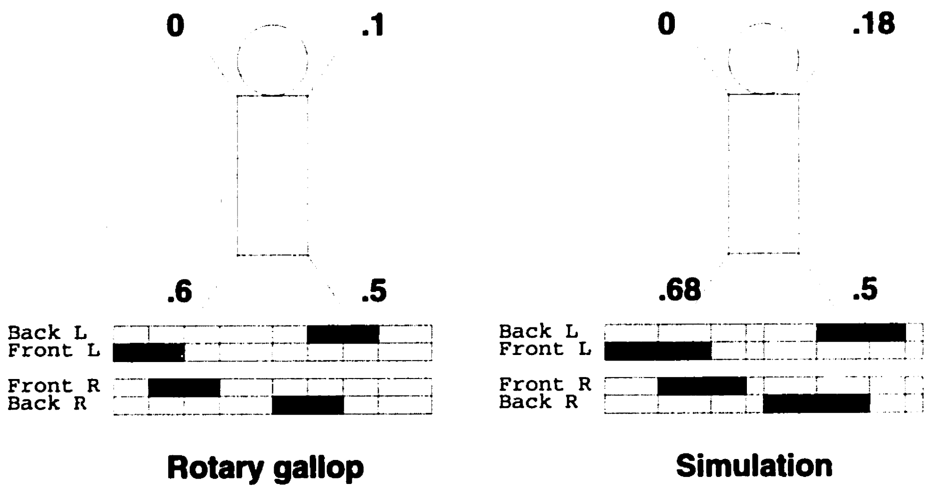

Figure 8-7: Relative phases for the galloping quadruped (rotary). Left is ideal, right is the actual result from the simulated quadruped.

disturbance will go away over time.

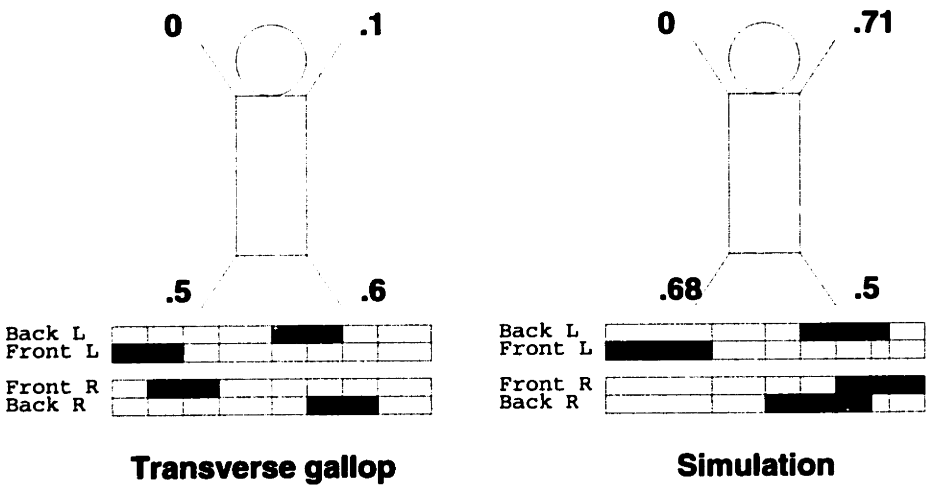

Figure 8-8: Relative phases for the galloping quadruped (transverse). Left is ideal, right is the actual result from the simulated quadruped. The simulated quadruped puts down the front left, back left, back right, and then the front right feet. This ordering is equivalent to a left-right reflected and slightly phase shifted version of the ideal gallop.

Leg 1 limp	−0.291
Leg 2 limp	0.291
Leg 3 limp	0.291
Leg 4 limp	−0.291
Leg 1 offset	0.15
Leg 2 offset	0.35
Leg 3 offset	0.85
Leg 4 offset	0.65
Foot radius	0.75
Leg b/k	0.0987
Leg C	0.4932
Leg k/m	100
λ	0.8333

Table 8.7: Leg limp fractions, timing offsets, and other parameters for the galloping quadruped (rotary).

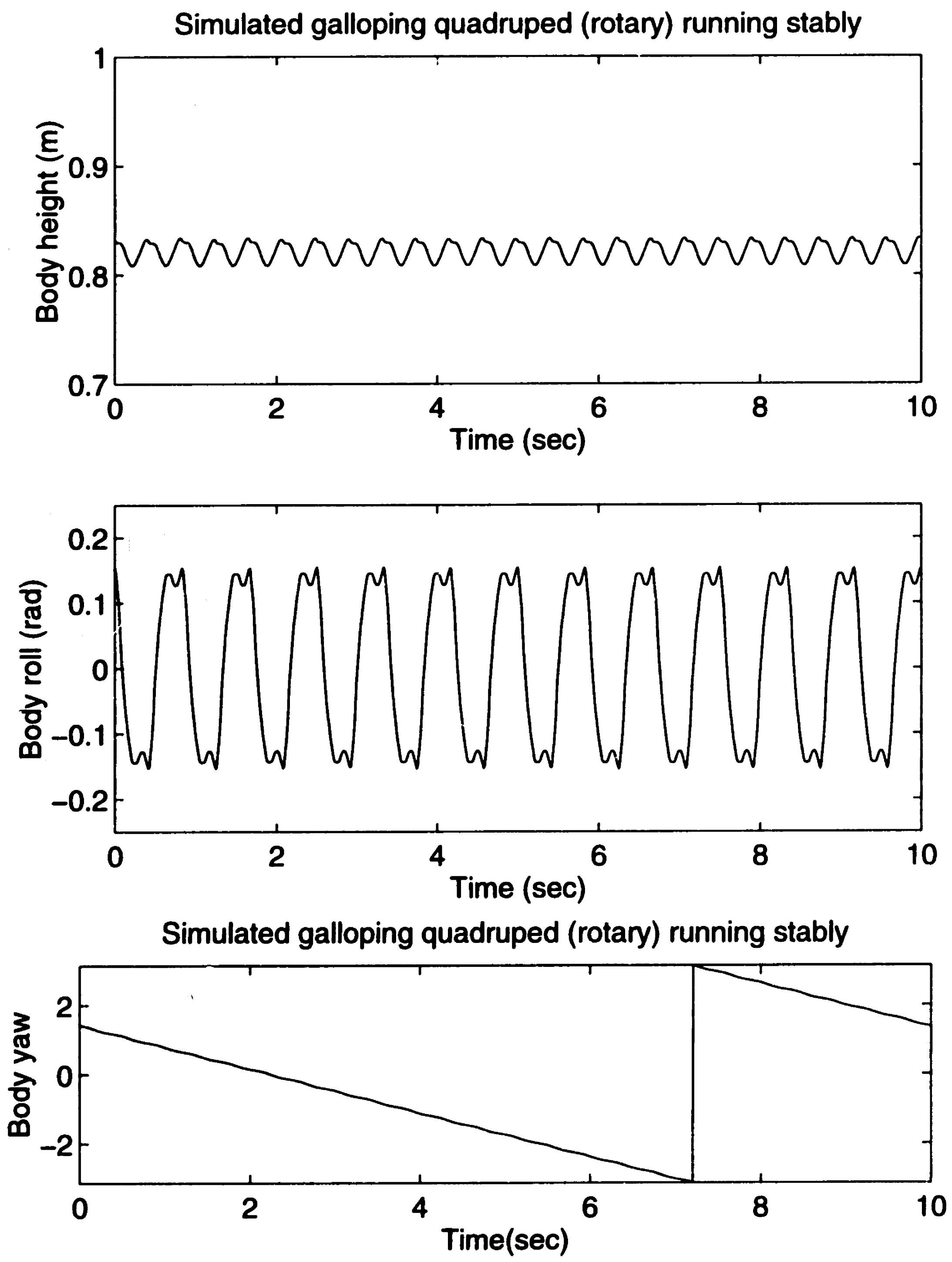

Figure 8-9: Galloping quadruped (rotary) running stably. The yaw position is included to show that the rotary galloping quadruped, unlike the other gaits, tends to spin in circles. The scale is the same as for the quadruped recovering from a disturbance.

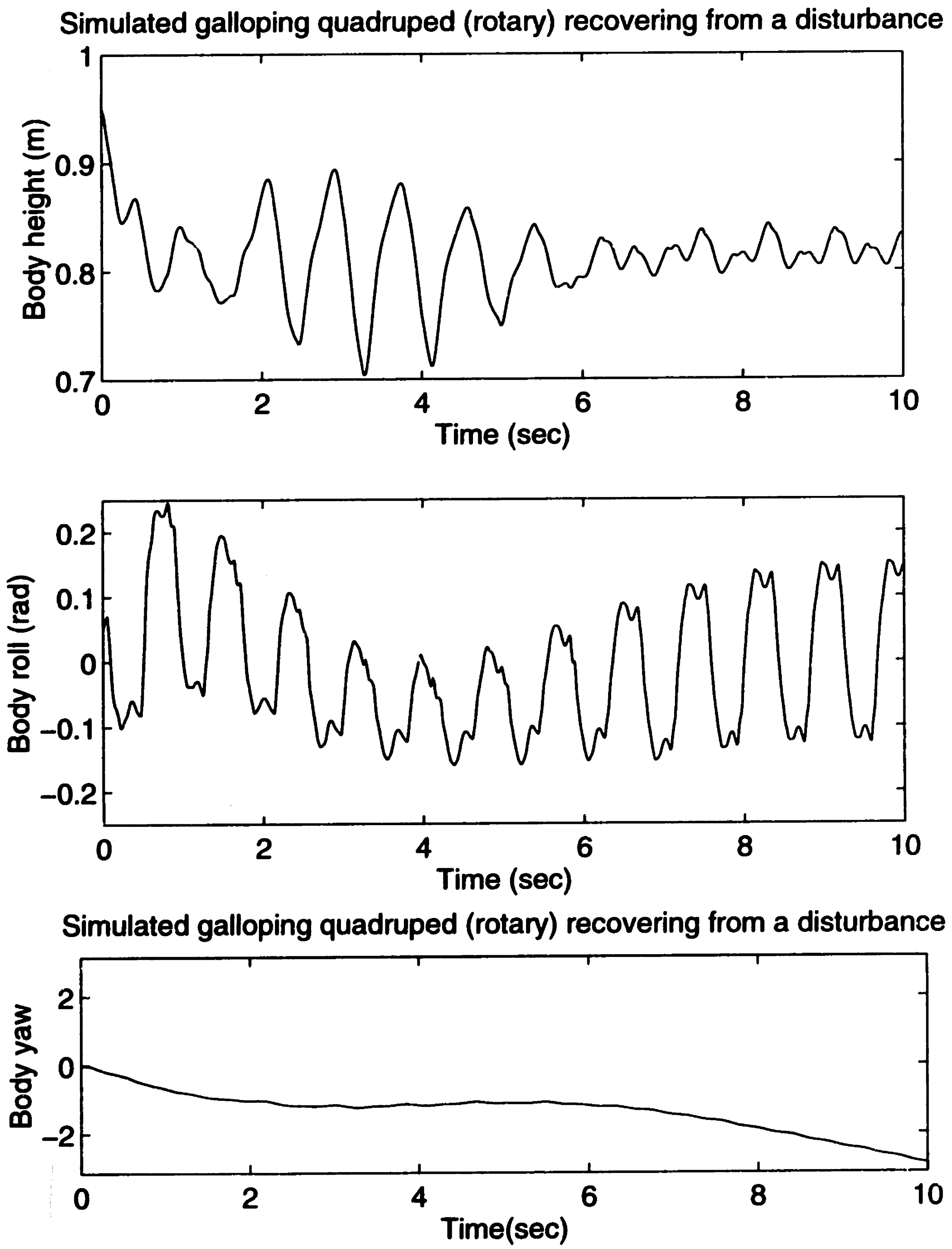

Figure 8-10: Galloping quadruped (rotary) recovering from a disturbance in both roll and vertical position. The disturbance occurs at time $t = 0$. The foot radius is 0.75m. The yaw position is included to show that the rotary galloping quadruped, unlike the other gaits, tends to spin in circles.

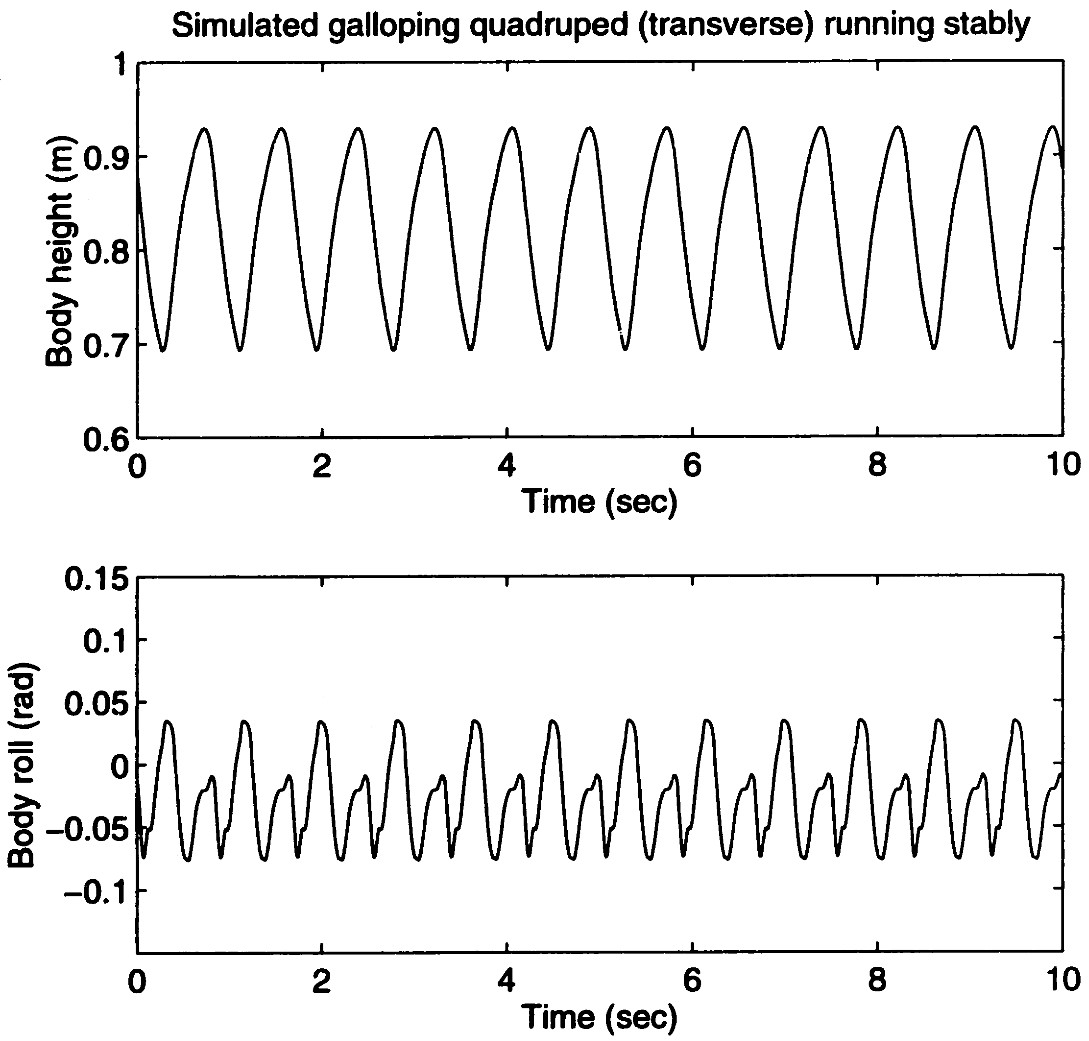

Figure 8-11: Galloping quadruped (transverse) running stably. The scale is the same as for the quadruped recovering from a disturbance.

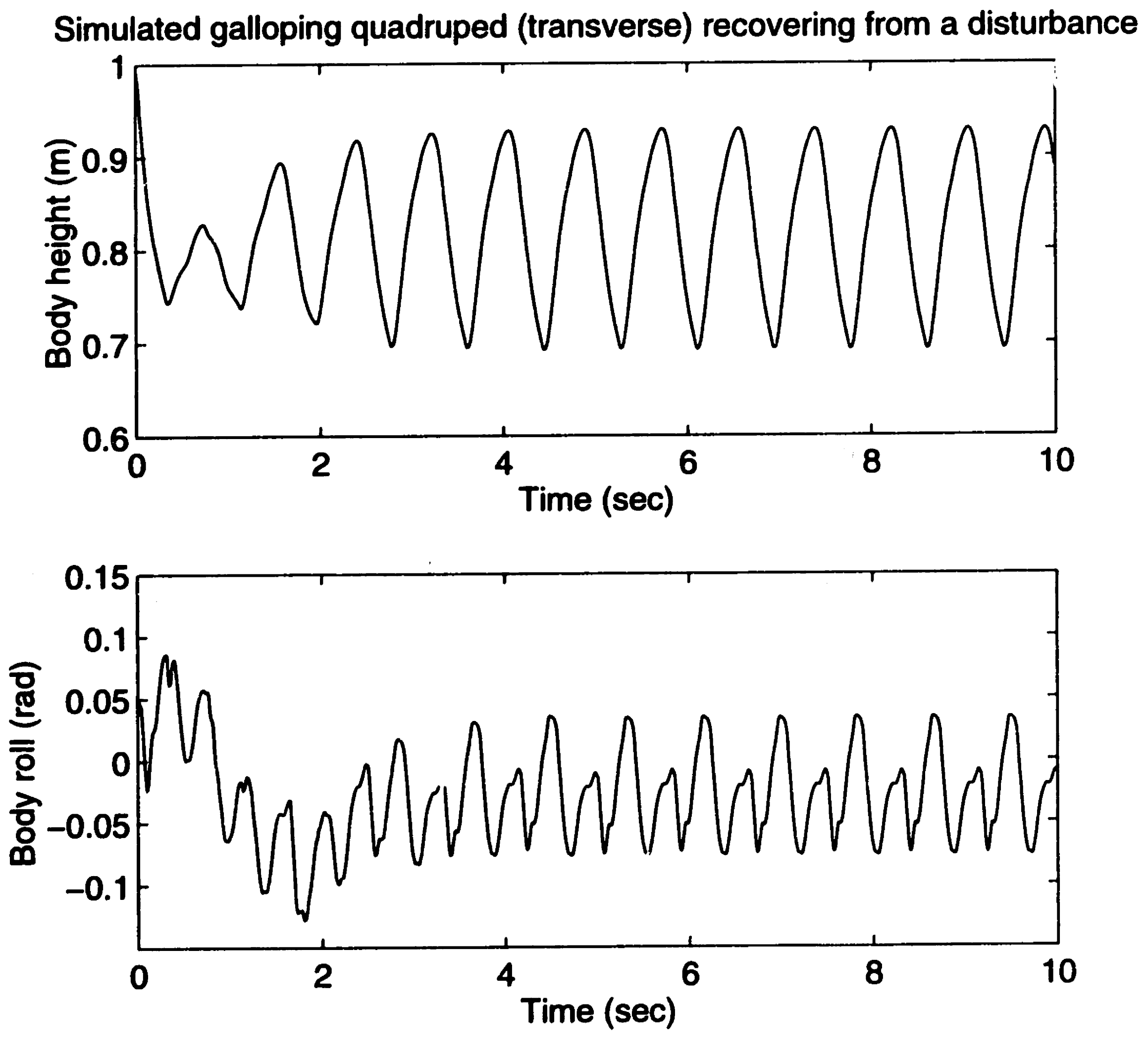

Figure 8-12: Galloping quadruped (transverse) recovering from a disturbance in both roll and vertical position. The disturbance occurs at time $t = 0$. The foot radius is 0.75m.

Leg 1 limp	0.291
Leg 2 limp	−0.291
Leg 3 limp	0.291
Leg 4 limp	−0.291
Leg 1 offset	0.35
Leg 2 offset	0.15
Leg 3 offset	0.85
Leg 4 offset	0.65
Foot radius	0.75
Leg b/k	0.0987
Leg C	0.4932
Leg k/m	100
λ	0.8333

Table 8.8: Leg limp fractions, timing offsets, and other parameters for the galloping g quadruped (transverse).

Eigenvector	1-2	3-4	5	6
Eigenvalue	$0.3678 \pm 0.9093i$	$0.7160 \pm 0.5223i$	−0.1872	−0.1306
x position	$-0.1046 \mp 0.0493i$	$0.0594 \mp 0.0009i$	−0.0446	−0.0150
y position	$-0.0235 \pm 0.1240i$	$0.0203 \mp 0.1008i$	0.0044	−0.0354
height	$0.0927 \mp 0.0052i$	$-0.0200 \pm 0.0027i$	0.1440	0.2459
yaw	$-0.1925 \pm 0.2048i$	$-0.3142 \pm 0.7420i$	0.0574	−0.1588
pitch	$-0.1220 \pm 0.0399i$	$0.0171 \mp 0.0225i$	−0.3046	−0.1304
roll	$-0.0104 \pm 0.0830i$	$-0.0645 \mp 0.0586i$	−0.0767	−0.0854
shoulder	$-0.0001 \pm 0.0005i$	$-0.0003 \mp 0.0003i$	−0.0004	−0.0002
hip 1 forward	$0.0065 \pm 0.0066i$	$-0.0003 \mp 0.0036i$	0.0118	0.0558
hip 1 side	$-0.0013 \pm 0.0021i$	$-0.0017 \mp 0.0007i$	−0.0047	−0.0239
leg 1 length	$-0.0194 \mp 0.0019i$	$-0.0029 \mp 0.0010i$	0.0267	−0.1761
hip 2 forward	$-0.0001 \mp 0.0001i$	$-0.0001 \mp 0.0001i$	−0.0003	0.0003
hip 2 side	$-0.0006 \mp 0.0015i$	$0.0005 \pm 0.0008i$	0.0020	−0.0011
leg 2 length	$-0.0009 \mp 0.0037i$	$0.0028 \pm 0.0017i$	0.0113	−0.0037
pelvis	$0.0001 \mp 0.0004i$	$0.0003 \pm 0.0004i$	0.0008	0.0008
hip 3 forward	$0.0001 \pm 0.0001i$	$0.0000 \mp 0.0000i$	−0.0002	−0.0001
hip 3 side	$0.0001 \mp 0.0001i$	$0.0001 \pm 0.0000i$	−0.0000	0.0006
leg 3 length	$0.0002 \mp 0.0000i$	$0.0002 \pm 0.0002i$	0.0002	0.0005
hip 4 forward	$0.0000 \pm 0.0001i$	$0.0000 \mp 0.0000i$	−0.0002	0.0000
hip 4 side	$0.0001 \mp 0.0002i$	$0.0001 \pm 0.0001i$	0.0000	0.0006
leg 4 length	$0.0003 \pm 0.0001i$	$0.0000 \mp 0.0000i$	−0.0037	0.0002
x velocity	$-0.1824 \pm 0.2791i$	$0.0165 \pm 0.0871i$	−0.2622	−0.0219
y velocity	$0.1931 \pm 0.0761i$	$-0.1012 \pm 0.0996i$	−0.1211	0.0471
height velocity	$0.2163 \pm 0.1189i$	$-0.0029 \pm 0.0061i$	0.6239	−0.5252
yaw velocity	$0.2135 \pm 0.4950i$	$0.3207 \pm 0.0936i$	−0.4370	0.1139
pitch velocity	$-0.4192 \mp 0.2858i$	$-0.1525 \mp 0.1399i$	−0.1955	0.6100
roll velocity	$-0.0566 \mp 0.1300i$	$0.0086 \pm 0.0295i$	0.0456	0.0595
shoulder vel.	$-0.0057 \pm 0.0269i$	$-0.0408 \mp 0.0167i$	0.0416	−0.0485
hip 1 forward vel.	$-0.0326 \mp 0.1596i$	$0.1963 \pm 0.1911i$	0.0795	−0.3068
hip 1 side vel.	$0.0335 \pm 0.0845i$	$-0.0845 \mp 0.0737i$	−0.1432	0.0227
leg 1 length vel.	$0.0725 \mp 0.1097i$	$0.0574 \pm 0.1045i$	−0.1849	0.2230
hip 2 forward vel.	$-0.0003 \pm 0.0093i$	$-0.0086 \mp 0.0084i$	0.0035	−0.0086
hip 2 side vel.	$0.0076 \pm 0.1029i$	$-0.0820 \mp 0.1017i$	−0.1666	−0.1561
leg 2 length vel.	$0.0078 \pm 0.0369i$	$-0.0301 \mp 0.0198i$	−0.1226	0.0513
pelvis velocity	$0.0012 \mp 0.0545i$	$0.0578 \pm 0.0608i$	0.1512	−0.0174
hip 3 forward vel.	$0.0008 \pm 0.0035i$	$-0.0018 \mp 0.0007i$	0.0032	0.0157
hip 3 side vel.	$0.0037 \mp 0.0675i$	$0.0591 \pm 0.0739i$	0.1406	0.0999
leg 3 length vel.	$0.0008 \pm 0.0087i$	$-0.0048 \mp 0.0061i$	−0.0067	−0.0049
hip 4 forward vel.	$-0.0021 \pm 0.0062i$	$-0.0069 \mp 0.0058i$	−0.0025	−0.0060
hip 4 side vel.	$-0.0023 \mp 0.0560i$	$0.0457 \pm 0.0612i$	0.1392	0.0464
leg 4 length vel.	$0.0019 \mp 0.0033i$	$0.0046 \pm 0.0046i$	0.0520	0.0117

Table 8.9: Eigenvectors associated with large eigenvalues, for the galloping quadruped (rotary). The eigenvectors for x and y position do not separate from yaw as well as they have for other simulations, due to numerical error. However, the eigenvectors are still indicative of the major modes of instability. (x, y, and *yaw* eigenvectors not displayed)

Eigenvector	1	2-3	4-5	6
Eigenvalue	1.0134	$-0.4153 \pm 0.1814i$	$0.0187 \pm 0.4505i$	0.2790
x position	0.5439	$-0.0224 \mp 0.0196i$	$-0.0628 \pm 0.0040i$	0.0515
y position	−0.8381	$-0.0322 \pm 0.0039i$	$0.0468 \pm 0.0511i$	0.0229
height	0.0001	$0.0103 \mp 0.0596i$	$-0.0203 \pm 0.0157i$	0.0357
yaw	0.0362	$-0.0235 \mp 0.0248i$	$0.1351 \pm 0.0898i$	−0.2901
pitch	−0.0006	$-0.0377 \pm 0.1200i$	$0.1025 \mp 0.0088i$	−0.1068
roll	−0.0004	$0.1577 \mp 0.1169i$	$0.0432 \mp 0.0067i$	0.0010
shoulder	−0.0000	$-0.0005 \mp 0.0002i$	$0.0000 \mp 0.0000i$	−0.0007
hip 1 forward	−0.0000	$-0.0000 \mp 0.0000i$	$0.0002 \mp 0.0001i$	−0.0002
hip 1 side	0.0000	$-0.0007 \pm 0.0009i$	$-0.0004 \mp 0.0002i$	−0.0002
leg 1 length	−0.0000	$0.0039 \mp 0.0020i$	$0.0010 \pm 0.0010i$	−0.0014
hip 2 forward	0.0001	$-0.0093 \pm 0.0114i$	$-0.0136 \pm 0.0044i$	0.0246
hip 2 side	−0.0000	$0.0001 \mp 0.0022i$	$0.0001 \pm 0.0010i$	0.0001
leg 2 length	0.0000	$-0.0114 \pm 0.0080i$	$-0.0189 \mp 0.0040i$	−0.0018
pelvis	0.0000	$0.0006 \mp 0.0001i$	$0.0000 \pm 0.0001i$	0.0006
hip 3 forward	−0.0000	$0.0003 \mp 0.0003i$	$0.0002 \mp 0.0001i$	−0.0002
hip 3 side	0.0000	$0.0008 \mp 0.0003i$	$0.0001 \pm 0.0001i$	0.0005
leg 3 length	−0.0000	$0.0003 \mp 0.0006i$	$0.0003 \pm 0.0001i$	0.0000
hip 4 forward	−0.0000	$0.0001 \mp 0.0002i$	$0.0001 \mp 0.0000i$	−0.0002
hip 4 side	0.0000	$0.0002 \pm 0.0001i$	$-0.0001 \mp 0.0000i$	0.0003
leg 4 length	−0.0000	$0.0007 \mp 0.0007i$	$0.0001 \pm 0.0000i$	−0.0004
x velocity	0.0001	$-0.1339 \pm 0.1955i$	$-0.0240 \pm 0.1344i$	−0.0186
y velocity	−0.0132	$-0.1743 \pm 0.3049i$	$0.0164 \mp 0.1133i$	−0.0323
height velocity	−0.0007	$0.0847 \mp 0.2206i$	$-0.2670 \pm 0.0839i$	−0.1267
yaw velocity	−0.0007	$-0.3683 \pm 0.3964i$	$0.1547 \mp 0.3512i$	0.4184
pitch velocity	−0.0066	$0.0257 \mp 0.0411i$	$0.7500 \mp 0.0666i$	−0.3300
roll velocity	−0.0002	$0.1966 \mp 0.2747i$	$0.0048 \mp 0.0225i$	−0.0746
shoulder vel.	0.0008	$0.0185 \pm 0.0941i$	$-0.0013 \mp 0.0137i$	0.0749
hip 1 forward vel.	0.0002	$-0.0233 \pm 0.0166i$	$-0.0117 \mp 0.0089i$	0.0044
hip 1 side vel.	0.0006	$0.0084 \pm 0.0062i$	$-0.0048 \mp 0.0162i$	0.0073
leg 1 length vel.	0.0001	$-0.0292 \pm 0.0087i$	$-0.0059 \mp 0.0101i$	0.0108
hip 2 forward vel.	−0.0102	$0.2042 \pm 0.3998i$	$0.1304 \pm 0.2378i$	0.7483
hip 2 side vel.	−0.0005	$0.0817 \pm 0.1708i$	$0.1038 \mp 0.0066i$	0.0811
leg 2 length vel.	0.0001	$0.1111 \mp 0.0831i$	$0.1935 \pm 0.0346i$	−0.0006
pelvis velocity	−0.0008	$-0.0046 \mp 0.1053i$	$0.0230 \pm 0.0116i$	−0.0957
hip 3 forward vel.	0.0001	$0.0076 \mp 0.0315i$	$0.0071 \mp 0.0037i$	−0.0405
hip 3 side vel.	−0.0008	$0.0220 \mp 0.1052i$	$0.0223 \pm 0.0238i$	−0.0520
leg 3 length vel.	0.0001	$-0.0118 \pm 0.0105i$	$-0.0157 \pm 0.0017i$	0.0086
hip 4 forward vel.	0.0002	$-0.0042 \pm 0.0018i$	$-0.0046 \mp 0.0030i$	−0.0083
hip 4 side vel.	−0.0002	$-0.0070 \mp 0.0452i$	$0.0181 \mp 0.0067i$	−0.0782
leg 4 length vel.	0.0001	$-0.0047 \pm 0.0115i$	$-0.0094 \pm 0.0018i$	0.0194

Table 8.10: Eigenvectors associated with large eigenvalues, for the galloping quadruped (transverse). The high value for hip 2 forward velocity in all eigenvectors is evidence of numerical error.

Chapter 9

Conclusion

I set out to show that running is possible without feedback, and to achieve a quadrupedal gallop. Feedback control can increase the stability of a mechanism, but it is possible to design a running robot which self-stabilizes. I generated self-stabilizing monopedal, bipedal, and quadrupedal simulations. For each simulation, I used the fact that the previous is self-stabilizing to reduce it to a solvable problem. I also applied mathematical techniques to demonstrate that they are stable. Although my simulations have telescoping legs, I think that it is possible to create a self-stabilizing monopod with a bending leg.

I found that running can be self-stabilizing. Provided the ground is flat and consistent, or even if the ground has minor bumps, it is possible to trot, pace, bound, and gallop without any sensors. This is important, because you can now chose sensors which are reliable, add them, and layer control based on those sensors over the self-stabilizing running. I believe that the technology for creating running robots exists, but the control and sensing has a long way to go. A self-stabilizing runner should make a useful base upon which one can layer more complex control systems, expanding the range in which the quadruped can robustly run. Should the more complex control systems fail, the self-stabilizing robot has a reasonably competent default behavior.

Self-stabilizing running, as I have implemented it, is only valid on flat surfaces. Depending on the nature of the terrain, the variances in height could be considered either disturbances to the running motion or problems which must be addressed by a higher level control system. If the ground varies little enough that it can be considered a disturbance to the running motion, no modification to the control is necessary. If the ground is irregular enough that further control is necessary, the extra control could be as simple as predicting the touchdown altitude and modifying the leg length and leg angle appropriately. In this case another benefit of self-stabilizing running is that, because the basic running motion is stable, errors resulting from small amounts of sensor noise should damp out.

Normal control systems have a set of required inputs. Sensors for these inputs must be added to the robot in order to use the control system. Self-stabilizing running, on the other hand, requires no sensors. As a result, any superimposed control systems can be chosen based on what sensors are available or reliable. Multiple sensors could be used with different superimposed controllers, although fusing the results is not as simple as just adding them. If there is a rock in the way, one controller may step on one side while the other might want to step on top. Somehow, whatever is fusing the control inputs must decide which controller to use in an intelligent manner. These are decisions the human brain makes frequently, especially when interpreting visual data. However, the decisions occur at a level above stable running and are out of the scope of this work.

Finding a self-stabilizing motion in something like running, which is traditionally considered unstable, makes me wonder: how many more "unstable" actions could actually be self-stabilizing? Is there a way to generalize the technique I have used here to break down self-stabilizing running into manageable pieces, and apply it to other problems?

In the nearer future, I believe that the next generation of active robots will have a self-stabilizing base, with feedback control modifying or overriding the basic stable motion as necessary. Actuators at the joints would have a basic path they follow, and an input signal which would modify that path according to some feedback control system. I believe this combination would combine the best of feedback control and self-stabilizing running. The self-stability would simplify the feedback control, and the feedback control would give the robot the versatility necessary to make it practical.

Appendix A

Reality Check

The central piece of this work is the fact that a monopod can hop self-stabilize height, phase, and pitch. The simulations I use are physically realistic, but reality has a way of adding problems which aren't dealt with in simulation. David Robinson and I have built a physical self-stabilizing monopod robot to verify the simulation results. Our robot hops stably in the presence of pitch, height, and phase disturbances and corresponds quite well with the simulated results.

Figure A-1 diagrams the physical robot. A linkage transfers the electric motor's rotary motion into an approximately sinusoidal vertical motion. The vertical motion (the actuator in the simulation) is applied to a spring which is connected to a curved foot. A set of linear bearings constrains the foot and leg to move vertically with respect to the robot, and a hinge mechanism keeps the foot from twisting (Figure A-2). The bottom of the foot is covered with rubber to prevent slippage. A boom planarizes the robot's motion. The robot can move vertically and horizontally, and it can pitch around the boom. The boom is not actuated.

The total actuator travel is six inches. At 78 motor revolutions per minute, it rapidly settles into a repeating, stable pattern in which it hops in the air 78 times per minute. The maximum foot height is consistent, approximately 12 inches. Simulation predicted, correctly, that a 20 inch foot radius would stabilize this machine's pitching motions. When

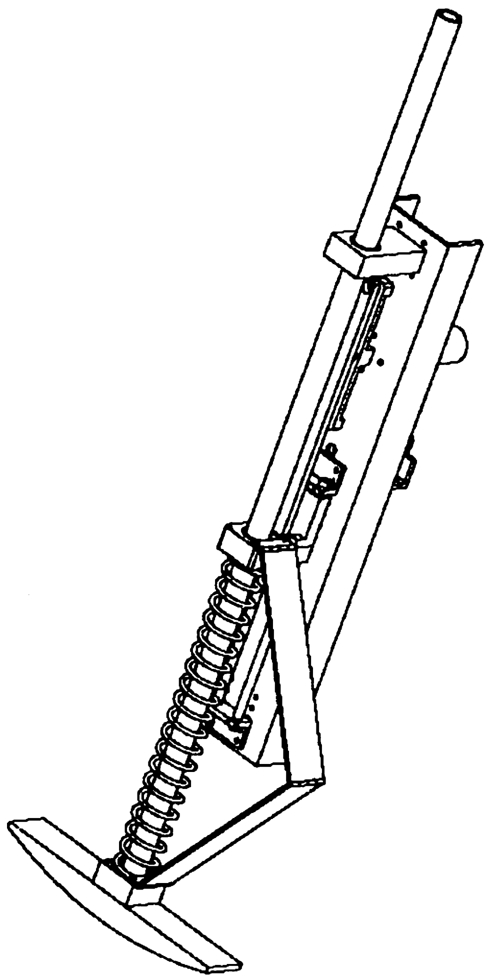

Figure A-1: Diagram of the self-stabilizing monopod hopper. The central actuator rotates, driving an arm. This arm attaches to a piston, converting the rotary motion to approximately sinusoidal linear motion. The moving section is attached to a spring, which connects to the foot and leg. Aside from the spring, the foot and leg are free to move vertically. The hinge mechanism prevents the foot from twisting.

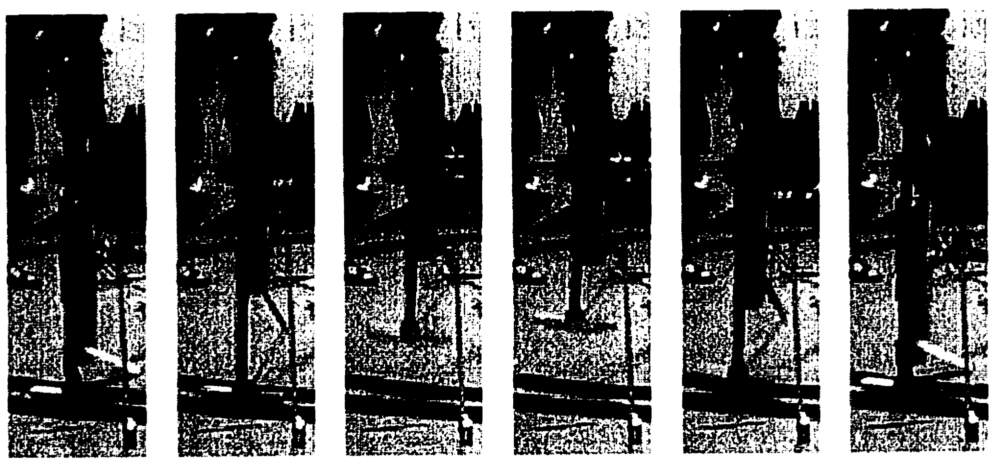

Figure A-2: The self-stabilizing monopod hopper. The boom prevents motion towards the camera, but allows motion side-to-side and pitching motion. These frames are taken left to right in succession from videotape, 0.133 seconds apart (four frames).

given a pitch perturbation, it pitches considerably for 2 hops and then stabilizes vertically. This is the same behavior as the simulation exhibits. The 20 inch foot radius is slightly larger than the height of the center of mass with the actuator fully retracted, so the monopod is stable when not in use.

We have also attached a flat foot to the monpod. In this configuration, it overcorrects for small pitch errors and falls over, as predicted in simulation.

Appendix B

Video

I did much of my work with simulated creatures. There is a videotape which includes computer-generated animations of these creatures. This appendix includes digitized sequences demonstrating the running motions of the various creatures, for use when that videotape is unavailable.

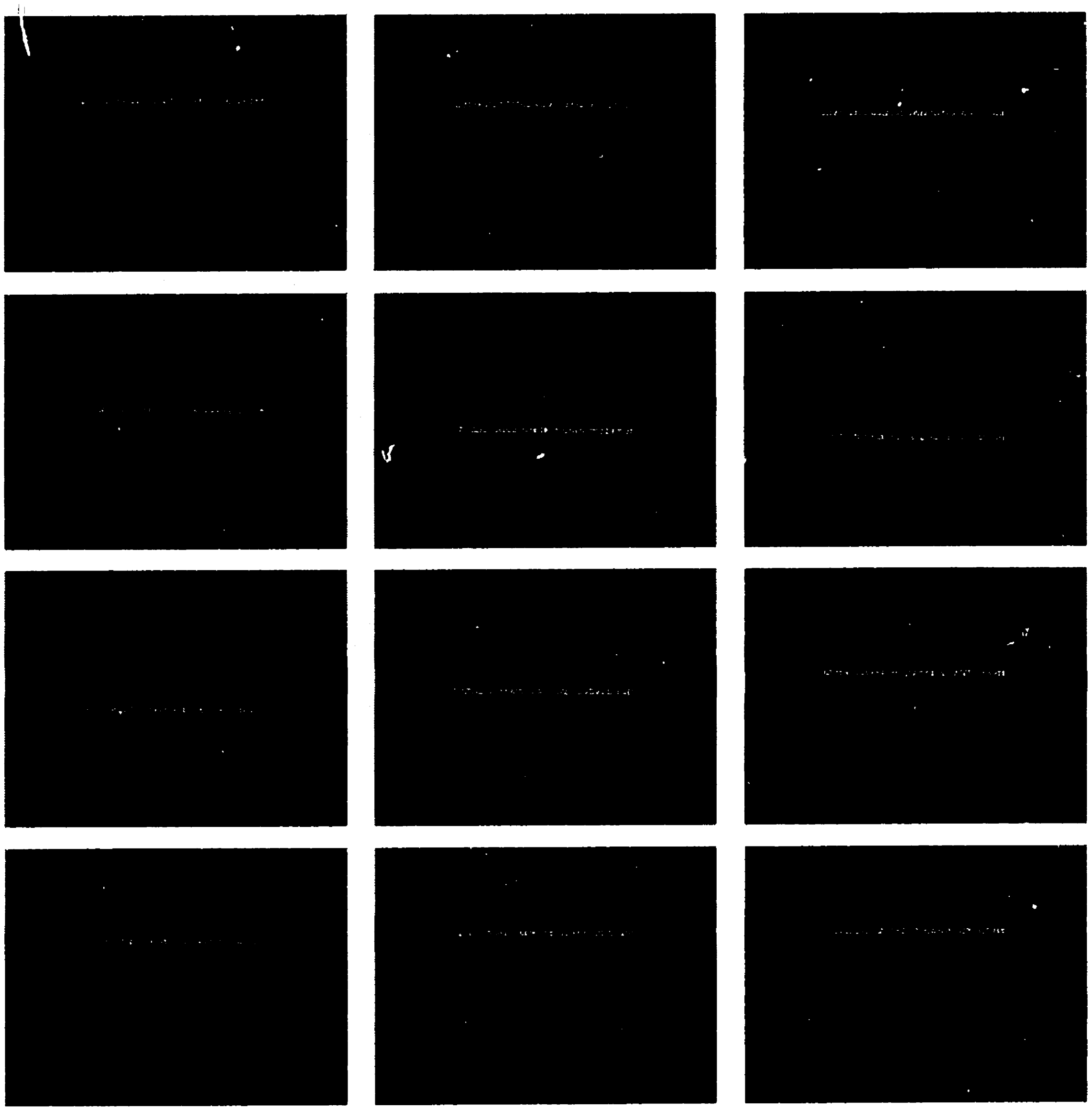

Figure B-1: The simulated monopod, as it goes through a single hop. The hopping cycle is 0.83333 seconds long, and the individual pictures (left to right, top to bottom) are separated by 0.07575 seconds. The twelve pictures make a complete cycle.

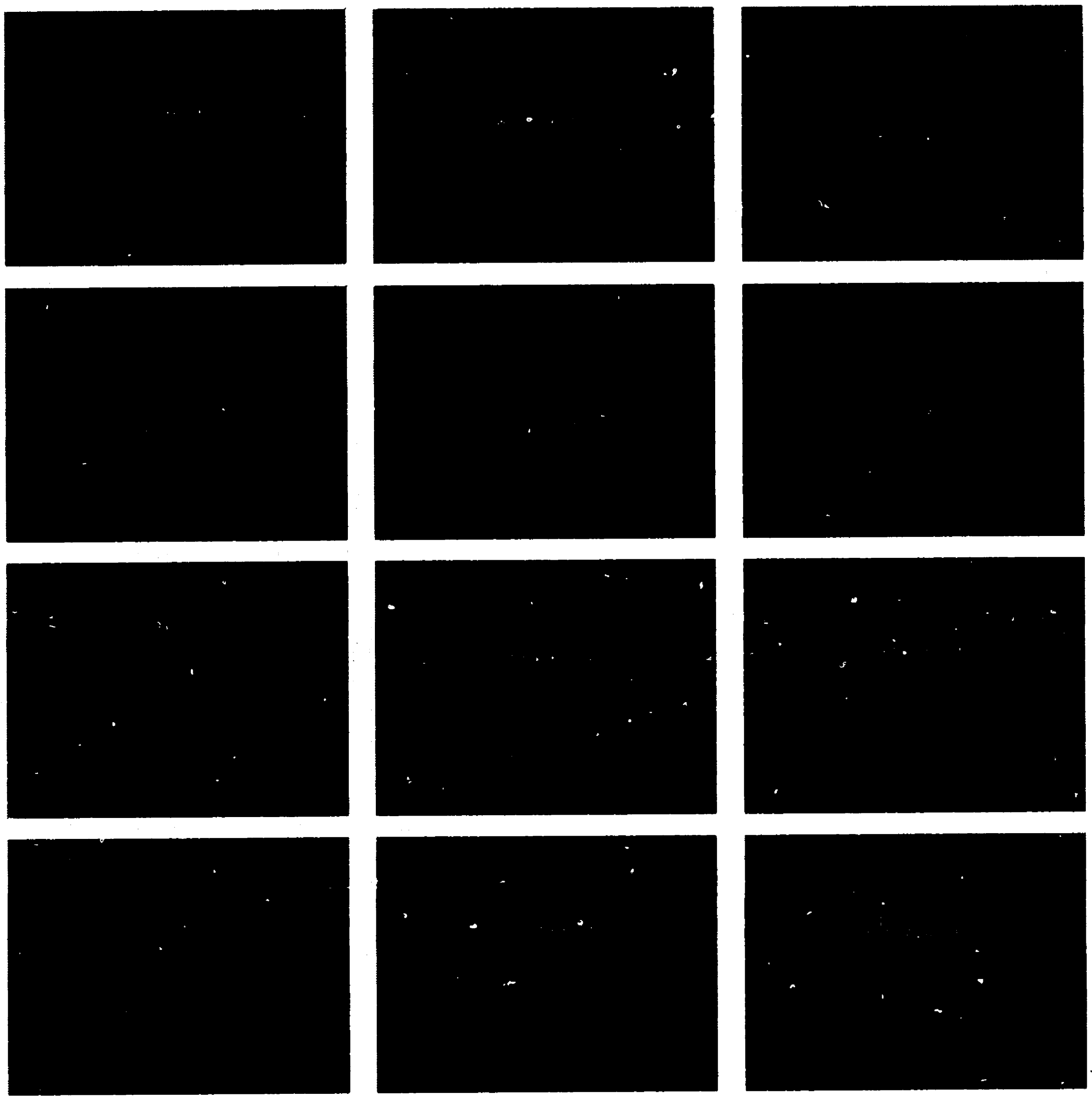

Figure B-2: The simulated biped hopping in phase. The hopping cycle is 0.83333 seconds long, and the individual pictures (left to right, top to bottom) are separated by 0.07575 seconds. The twelve pictures make a complete cycle.

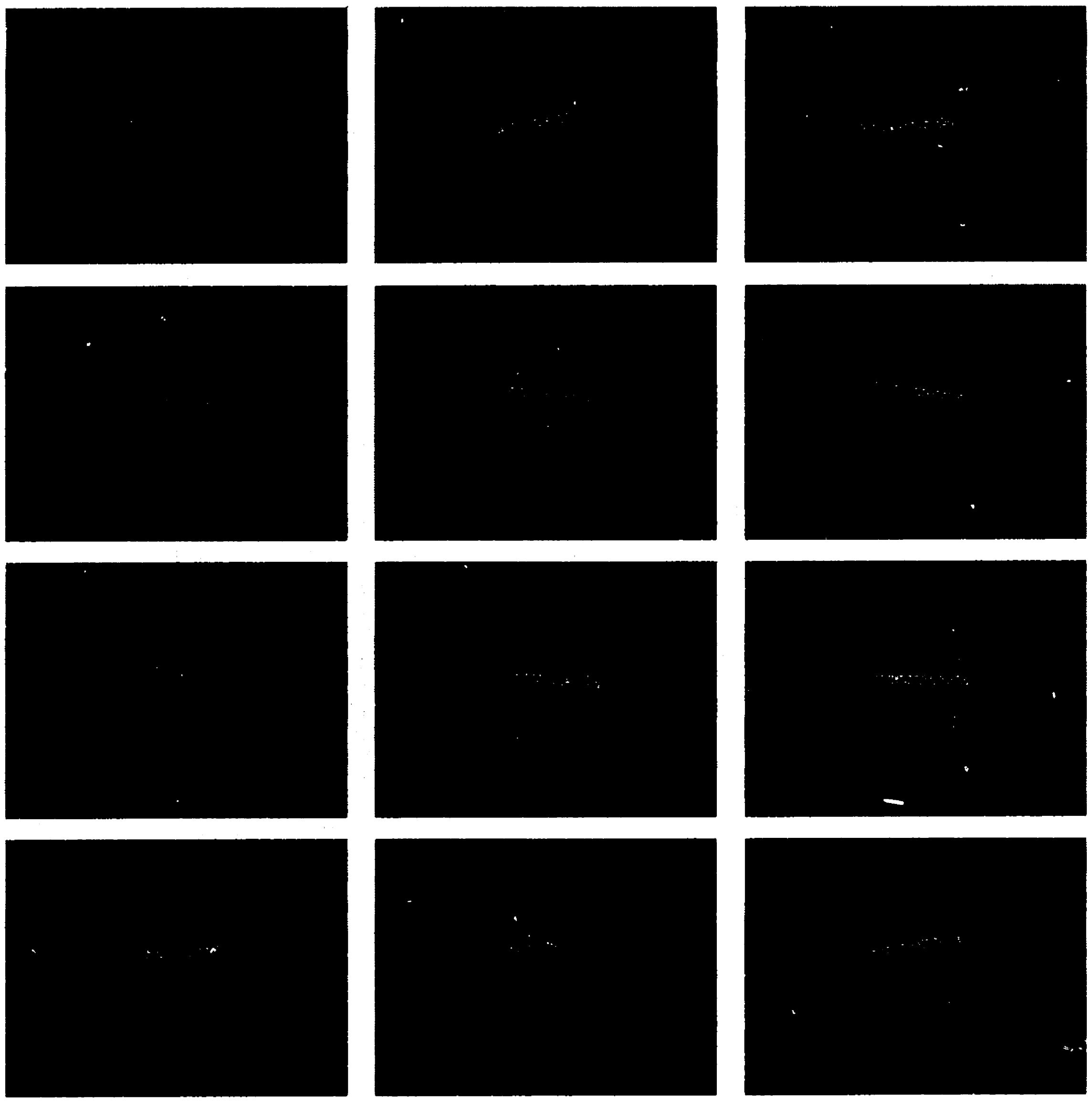

Figure B-3: The simulated biped, hopping out of phase. The hopping cycle is 0.83333 seconds long, and the individual pictures (left to right, top to bottom) are separated by 0.07575 seconds. The twelve pictures make a complete cycle.

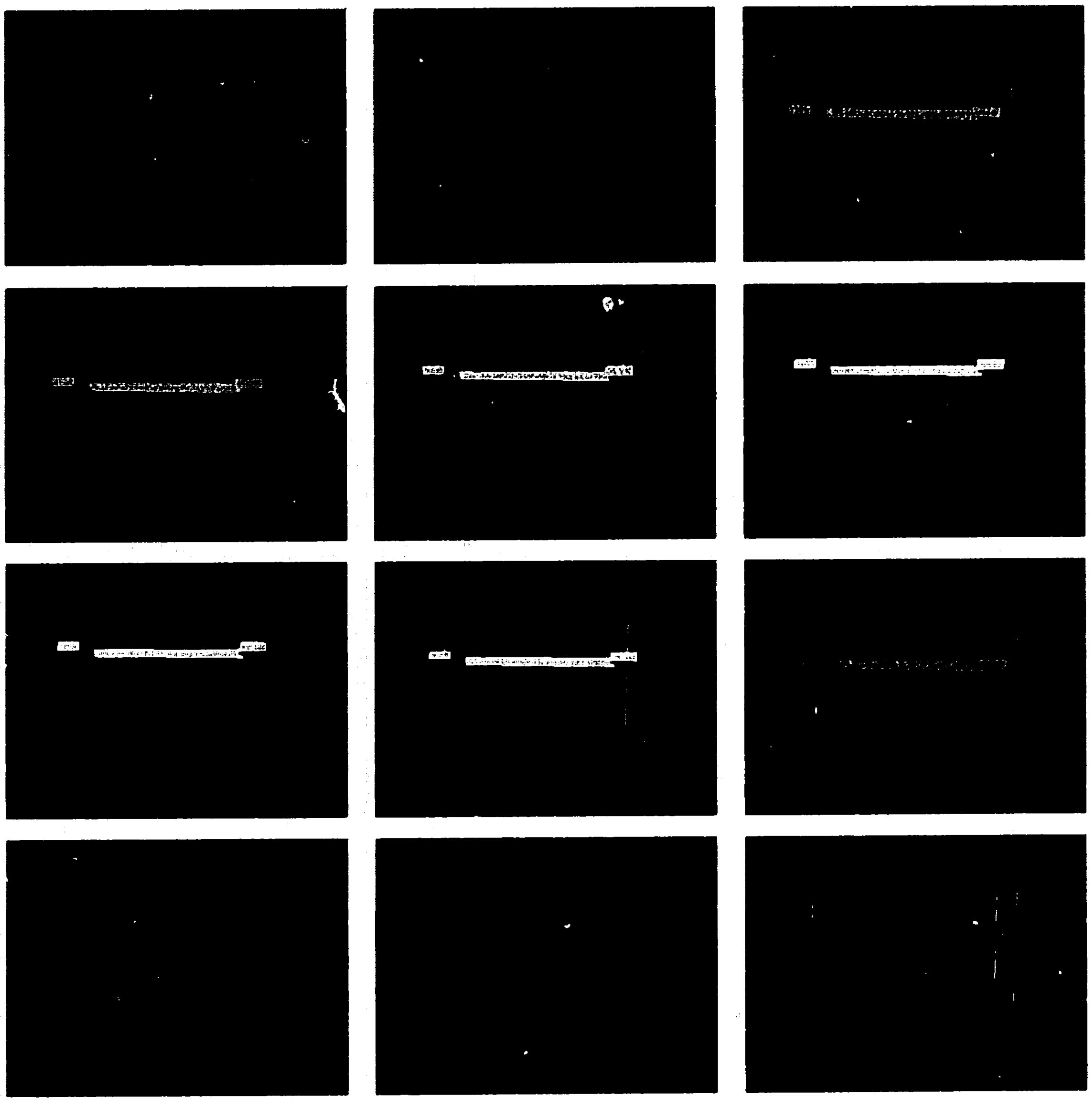

Figure B-4: The simulated quadruped, pacing. The running cycle is 0.83333 seconds long, and the individual pictures (left to right, top to bottom) are separated by 0.07575 seconds. The twelve pictures make a complete cycle.

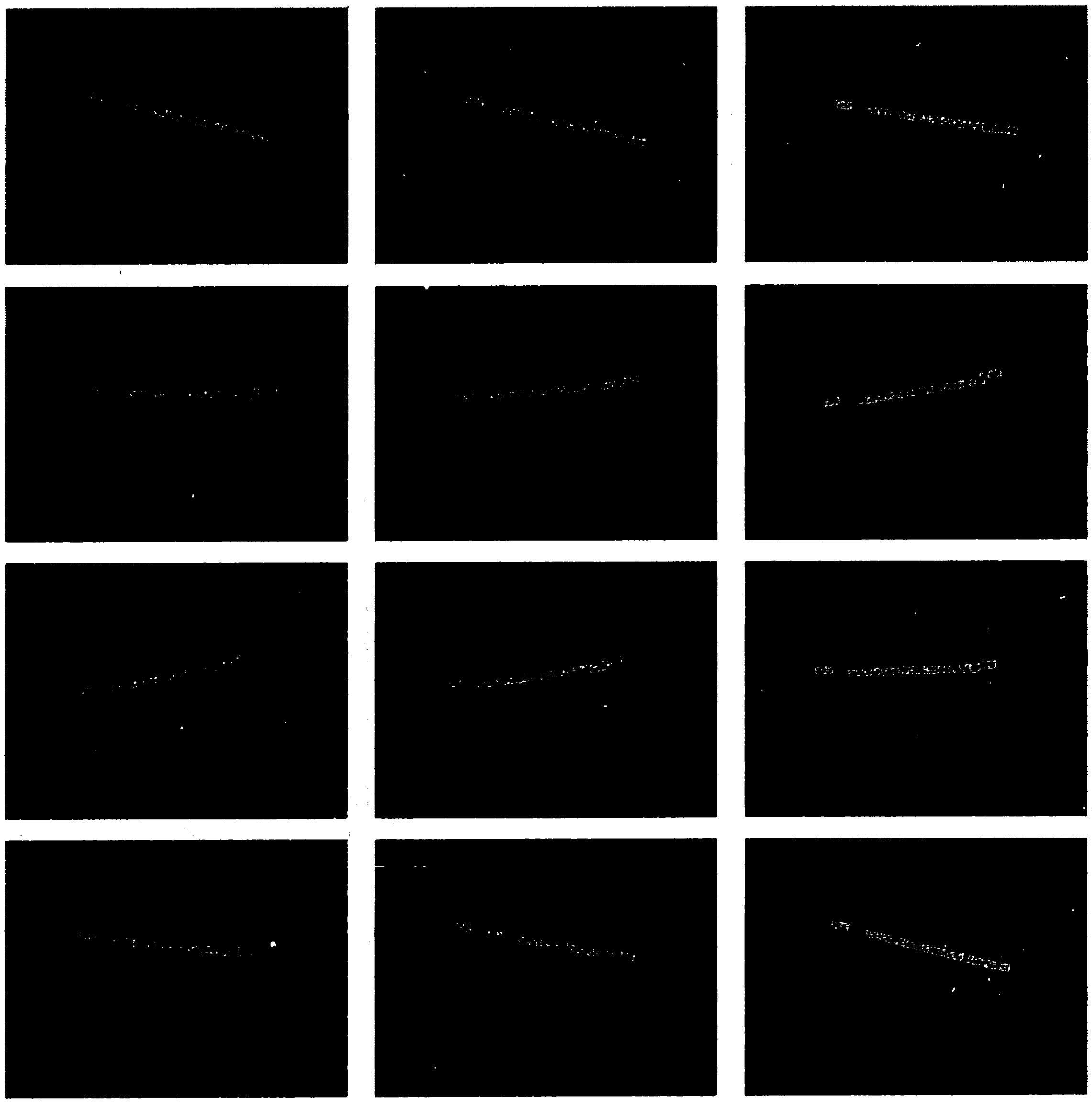

Figure B-5: The simulated quadruped, bounding. The running cycle is 0.83333 seconds long, and the individual pictures (left to right, top to bottom) are separated by 0.07575 seconds. The twelve pictures make a complete cycle.

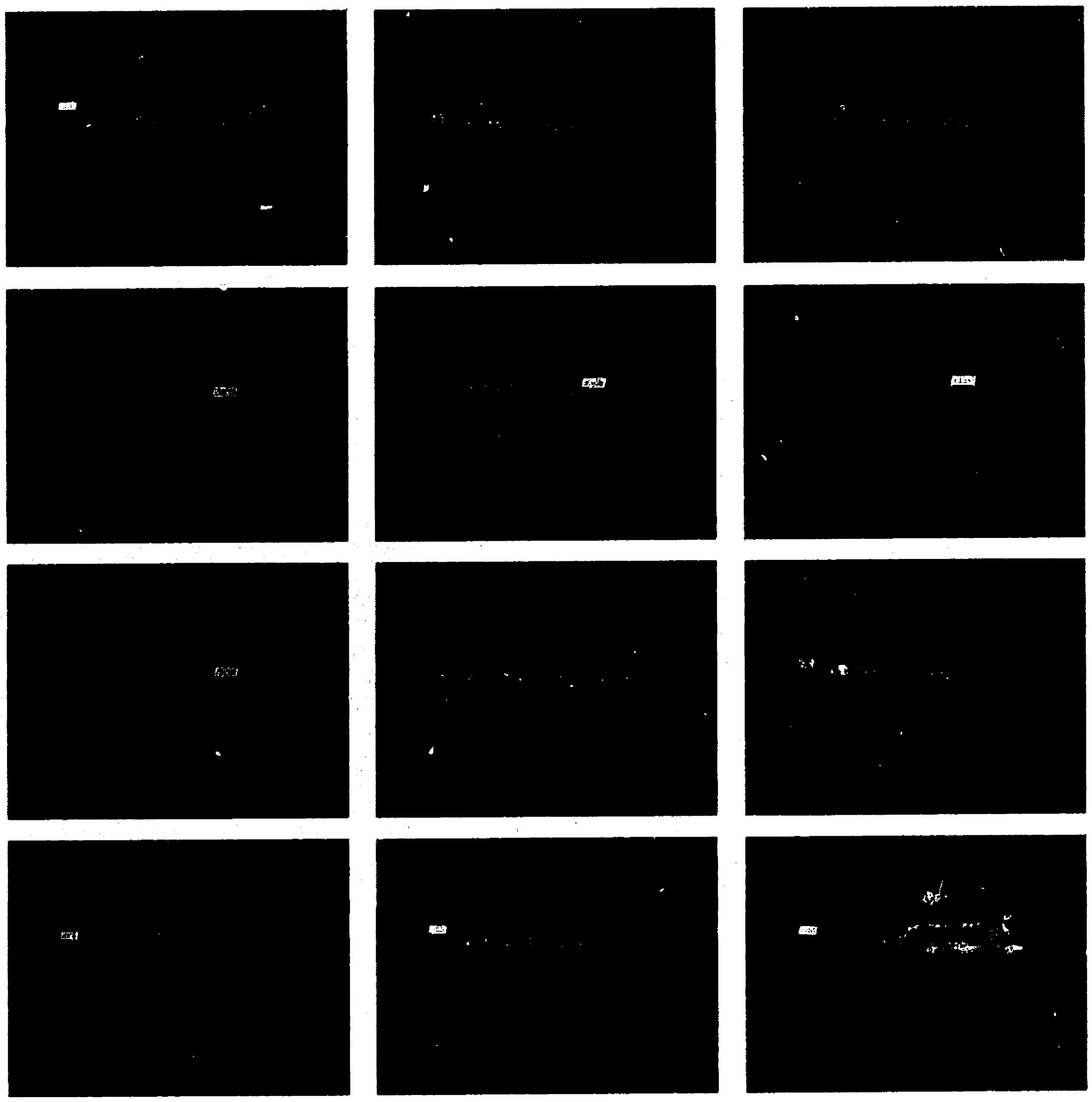

Figure B-6: The simulated quadruped, trotting. The running cycle is 0.83333 seconds long, and the individual pictures (left to right, top to bottom) are separated by 0.07575 seconds. The twelve pictures make a complete cycle.

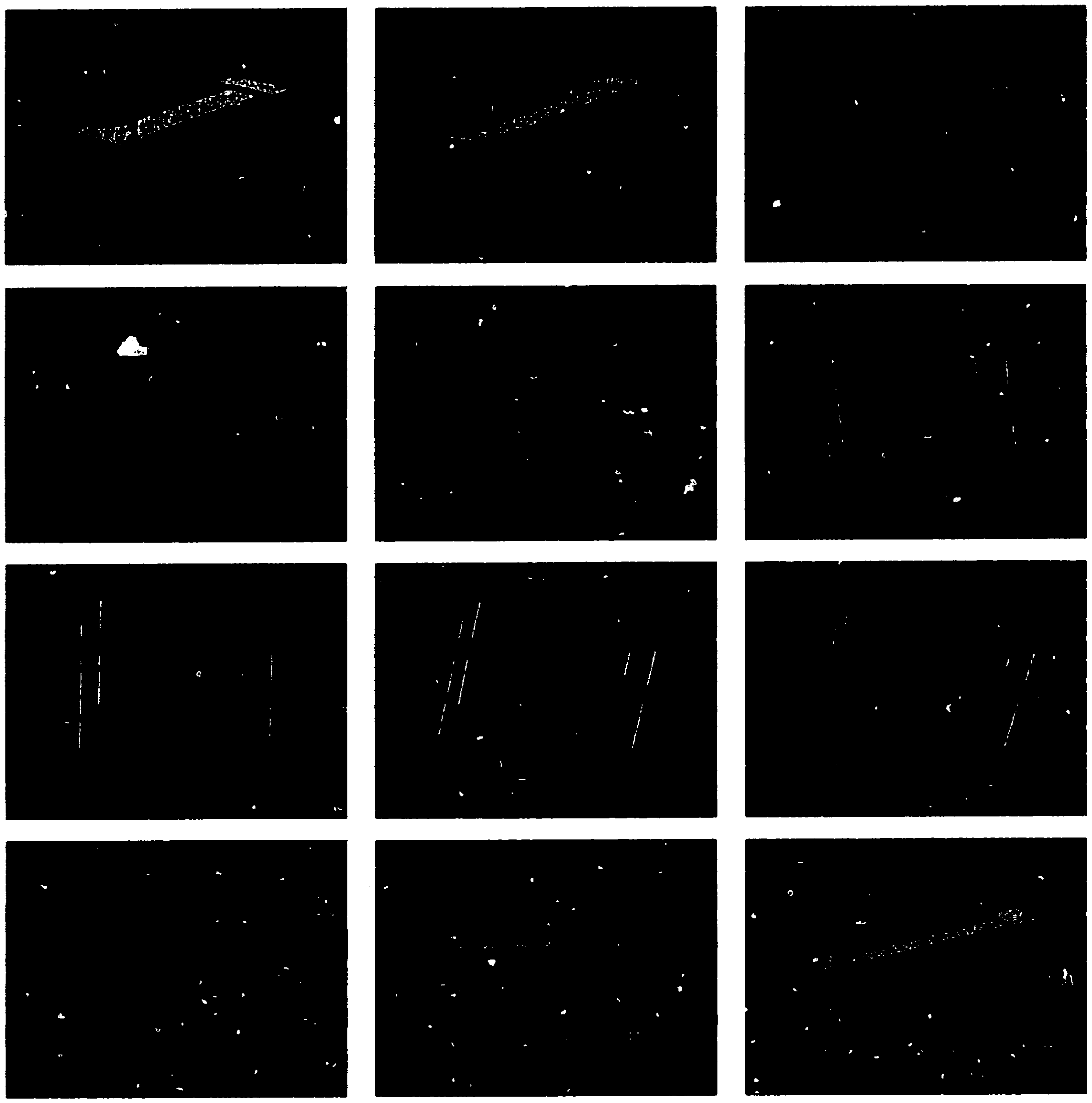

Figure B-7: The simulated quadruped, galloping (rotary). The running cycle is 0.83333 seconds long, and the individual pictures (left to right, top to bottom) are separated by 0.07575 seconds. The twelve pictures make a complete cycle.

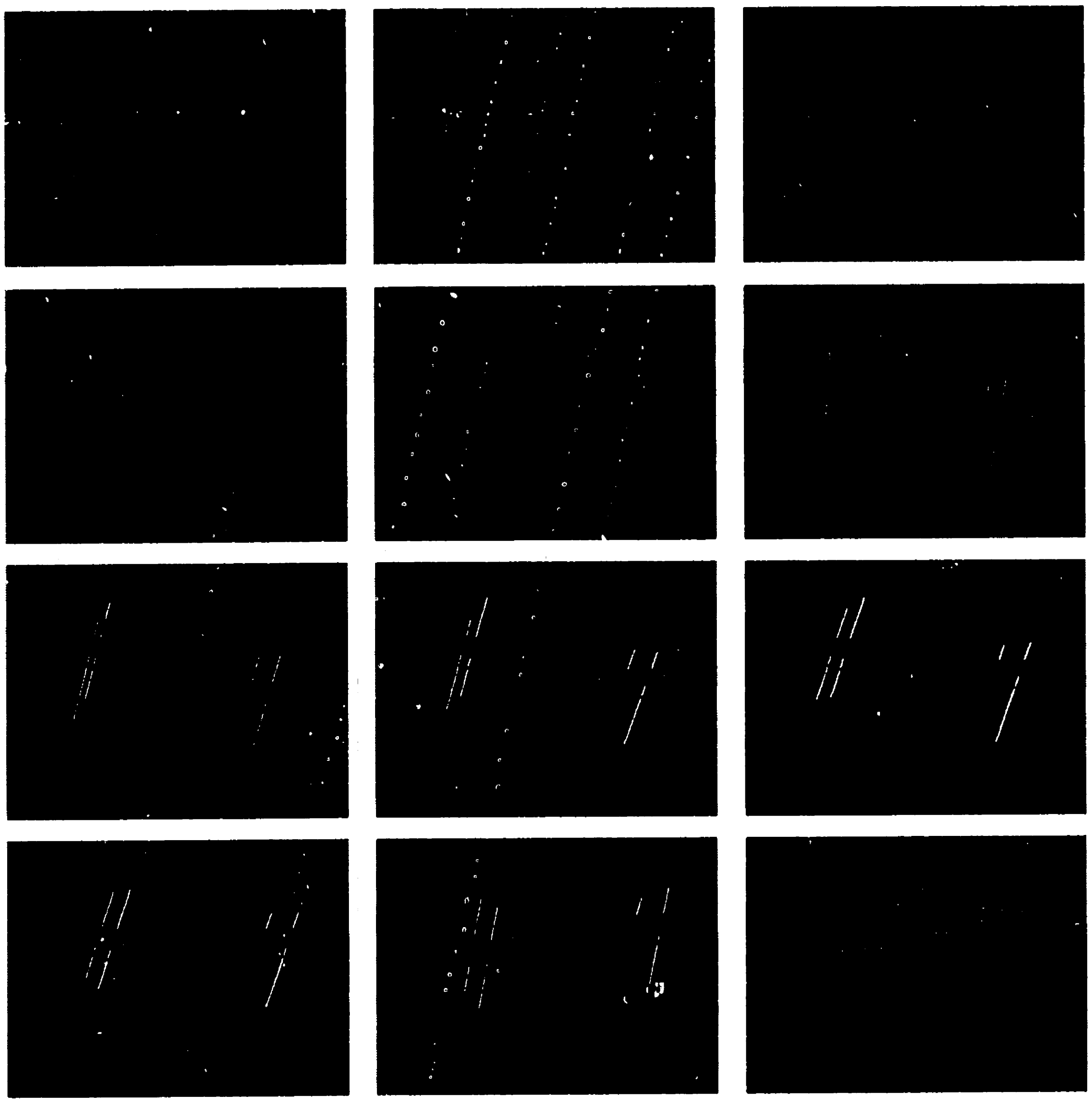

Figure B-8: The simulated quadruped, galloping (transverse). The running cycle is 0.83333 seconds long, and the individual pictures (left to right, top to bottom) are separated by 0.07575 seconds. The twelve pictures make a complete cycle.

Bibliography

Baraff, D. 1989. Analytical methods for dynamic simulation of non-penetrating rigid bodies. In *Computer Graphics*, volume 23, 223–232. ACM.

Baraff, D. 1990. Curved surfaces and coherence for non-penetrating rigid body simulations. In *Computer Graphics*, volume 24, 19–28. ACM.

Buhler, M. 1990. *Robotic Tasks with Intermittent Dynamics.* Ph.D. Dissertation, Yale University.

Burdick, J. W.; Vakakis, A. F.; and Caughey, T. K. 1989. Periodic orbits and a strange-strange attractor in the dynamics of a simplified hopping robot. Technical Report RMS-89-01, California Institute of Technology, School of Engineering and Applied Science, Pasadena, CA 91125.

Gambaryan, P. 1974. *How Mammals Run: Anatomical Adaptations.* New York: Wiley.

Gurfinkel, V. S.; Gurfinkel, E. V.; Shneider, A. Y.; Devjanin, E. A.; Lensky, A. V.; and Shitilman, L. G. 1981. Walking robot with supervisory control. *Mechanism and Machine Theory* 16:31–36.

Hahn, J. K. 1988. Realistic animation of rigid bodies. In *Computer Graphics*, volume 22, 299–308. ACM.

Hirose, S., and Umetani, Y. 1980. The basic motion regulation system for a quadruped walking vehicle. In *ASME Conference on Mechanisms.*

Hirose, S. 1984. A study of design and control of a quadruped walking vehicle. *International Journal of Robotics Research* 3:113–133.

Koditschek, D., and Buhler, M. 1991. Analysis of a simplified hopping robot. *International Journal of Robotics Research* 10(6):587–605.

Krotkov, E., and Simmons, R. 1996. Perception, planning, and control for autonomous walking with the ambler planetary rover. *The International Journal of Robotics Research* 15(2):115–180.

McGeer, T. 1989. Passive bipedal running. Technical Report CCS-IS TR 89-02, Simon Fraser University.

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

McGhee, R. B. 1968. Some finite state aspects of legged locomotion. *Math. Biosci.* 2:67–84.

McGhee, R. B. 1983. Vehicular legged locomotion. In Saridis, G. N., ed., *Advances in Automation and Robotics.* JAI Press.

McMahon, T. A.; Valiant, G.; and Frederick, E. C. 1987. Groucho running. *Americal Physiological Society* 62:2326–2337.

Moore, M., and Wilhelms, J. 1988. Collision detection and response for computer animation. In *Computer Graphics*, volume 22, 289–298. ACM.

Murthy, S. S., and Raibert, M. H. 1983. 3d balance in legged locomotion: Modelling and simulation for the one-legged case. In *Inter-Disciplinary Workshop on Motion: Representation and Perception.* ACM.

Pan American Navigation Service, 12021 Ventura Boulevard, North Hollywood, California. 1972. *The New Private Pilot*, 9 edition.

Platt, J. C., and Barr, A. H. 1988. Constraint methods for flexible models. In *Computer Graphics*, volume 22, 279–288. ACM.

Raibert, M. H., and Hodgins, J. K. 1991. Animation of dynamic legged locomotion. In *Computer Graphics*, 349–358. ACM SIGGRAPH.

Raibert, M. H.; Hodgins, J. K.; Playter, R. R.; and Ringrose, R. P. 1992. Animation of maneuvers: Jumps, somersaults, and gait transitions. In *Imagina.*

Raibert, M. H.; Chepponis, M.; and Brown, Jr., B. 1986. Running on four legs as though they were one. *IEEE Journal of Robotics and Automation* RA-2(2).

Raibert, M. 1986. *Legged Robots that Balance.* Cambridge, MA: MIT Press.

Raibert, M. H. 1990. Trotting, pacing and bounding by a quadruped robot. *Journal of Biomechanics* 23.

Ringrose, R. 1992a. The creature library. Unpublished reference guide to a C library used to create physically realistic simulations.

Ringrose, R. 1992b. Simulated creatures: Adapting control for variations in model or desired behavior. Master's thesis, Massachusetts Institute of Technology.

Rosenthal, D. E., and Sherman, M. A. 1986. High performance multibody simulations via symbolic equation manipulation and kane's method. *Journal of Astronautical Sciences* 34(3):223–239.

Schaal, S., and Atkeson, C. G. 1993. Open loop stable control strategies for robot juggling. In *IEEE International Conference on Robotics and Automation*, volume 3, 913–918.

Siljak, D. D. 1969. *Nonlinear Systems.* New York: Wiley.

Song, S.-M., and Waldron, K. J. 1989. *Machines that Walk.* Cambridge, MA: MIT Press.

Thompson, C., and Raibert, M. 1989. Passive dynamic running. In Hayward, V., and Khatib, O., eds., *International Symposium of Experimental Robotics*, 74–83. New York: Springer-Verlag.