Design and Control of a Planar Robot to Study Quadrupedal Locomotion

by

Benjamin T. Krupp

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2000

Signature of Author ....
Department of Mechanical Engineering
May 5, 2000

Certified by ....
Gill A. Pratt
Assistant Professor of Electrical Engineering and Computer Science, MIT
Thesis Supervisor

Certified by ....
Ernesto Blanco
Adjunct Professor of Mechanical Engineering, MIT
Thesis Supervisor

Accepted by ....
Ain Sonin
Chairman, Departmental Committee on Graduate Students

Design and Control of a Planar Robot to Study Quadrupedal Locomotion

by

Benjamin T. Krupp

Submitted to the Department of Mechanical Engineering
on May 5, 2000, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

In this thesis we present the design of a planar robot with characteristics favorable for control of running gaits. The robot possess four degrees of freedom robot and is constrained to motion in a plane. It was designed to take advantage of known biological mechanisms which allow for efficient locomotion. Specifically, the natural dynamics of the swing leg and the use of passive springy feet are incorporated into the design. An intuitive control scheme is used to control standing and pronking of the experimental robot.

Thesis Supervisor: Gill A. Pratt
Title: Assistant Professor of Electrical Engineering and Computer Science, MIT

Thesis Supervisor: Ernesto Blanco
Title: Adjunct Professor of Mechanical Engineering, MIT

Acknowledgments

I would like to personally thank everyone in the Leg Lab for their support of this project. It has been both an honor and a pleasure to work with all the wonderful engineers in the Leg Lab. I would especially like to thank Gill Pratt for advising me on this work. I was always amazed by your uncanny ability to step back from the problem and point out subtleties that I had missed. CornDog would not have been possible without the support of others in the Leg Lab. Jerry Pratt's previous work on Spring Turkey and Spring Flamingo proved to be indispensable. Dave Robinson designed CornDog's actuators, allowing this work to be finished in a timely fashion. A special thanks to Chris Morse for helping me design CornDog. I would like to thank Dan Paluska for designing much of the electronic hardware for CornDog. Allen Parsegian helped me considerably with my simulations. I owe much to Mike Wessler for the development of the GUI interface to CornDog. Peter Dilworth's harassment often solidified my determination.

There are many who offered moral support during the past two years. I would like to thank my friends and family for their continued love and support over the past two years. I owe everything I have to my parents. Mom and Dad, you have been incredible through it all! Finally, I would like to dedicate this thesis to my best friend, Emily Bay. You selflessly encouraged me to go to MIT and stood by my side during the tough times. Without you this thesis would not have been possible. Your love and generosity will never be forgotten.

This research was supported in part by the Defense Advanced Research Projects Agency under contract number N39998-00-C-0656 and the National Science Foundation under contract numbers IBN-9873478 and IIS-9733740.

Contents

1 Introduction **11**
- 1.1 Goals of Thesis . . . 11
- 1.2 Summary of Thesis . . . 11

2 Design Inspiration **13**
- 2.1 Energetics of Walking . . . 13
- 2.2 Energetics of Running . . . 14
- 2.3 Natural Dynamics . . . 15
- 2.4 Other Important Biological Elements . . . 16
- 2.5 Design Recommendations . . . 16

3 Spring Selection **18**
- 3.1 Modeling . . . 18
  - 3.1.1 An Ideal Model . . . 18
  - 3.1.2 A Realistic Model . . . 23
- 3.2 Simulation . . . 25
  - 3.2.1 Verification of Simulation . . . 25
  - 3.2.2 Efficiency Analysis and Simulation . . . 27
  - 3.2.3 The Virtual Spring Effect . . . 27
  - 3.2.4 The Alpha Effect . . . 34
  - 3.2.5 Controllability Analysis and Simulation . . . 35
- 3.3 Recommendations . . . 40
  - 3.3.1 Selection and Use of Springs . . . 40

4 Natural Dynamics **42**
- 4.1 Design for Natural Dynamics . . . 42
- 4.2 Impediments to Natural Dynamics . . . 43
- 4.3 Masking the Effects of Reflected Inertia . . . 45
- 4.4 Summary of Natural Dynamics . . . 48

5 Experimental Robot **49**
- 5.1 Planarizing Boom . . . 49
- 5.2 Size and Weight . . . 49
- 5.3 Body, Legs, and Joints . . . 52
- 5.4 Spring . . . 53
- 5.5 Actuators . . . 55
- 5.6 Electronic Components . . . 56

6 Virtual Model Control **60**
- 6.1 Introduction to Virtual Model Control . . . 60
- 6.2 Implementation of Virtual Model Control . . . 61
  - 6.2.1 Low-level Arm Controller . . . 62
  - 6.2.2 Low-level Leg Controller . . . 63

6.2.3 Superposition of Arm and Leg Controller 64
6.3 An Alternate Implementation of Virtual Model Control 67
6.3.1 Low-level Arm Controller 67
6.3.2 Virtual Force to Body Force Transformation 69
6.4 Summary of Virtual Model Control 71

7 Simulated and Experimental Control **72**
7.1 Standing Control 72
7.1.1 Standing Control Algorithm 72
7.1.2 Simulated Standing Control 74
7.1.3 Experimental Standing Control 77
7.1.4 Summary of Standing Control 80
7.2 Pronking Control 80
7.2.1 Pronking Control Algorithm 80
7.2.2 Simulated Pronking Control 86
7.2.3 Experimental Pronking Control 89
7.2.4 Summary of Pronking Control 89

8 Conclusions **97**
8.1 The Use of Springs 97
8.2 The Use of Natural Dynamics 97
8.3 The Development of Control Algorithms 98
8.4 Meeting the Power Requirements 98

List of Figures

2-1 Several strides of the compass gait shown with qualitative potential and kinetic energy curves. 14
2-2 Springs used to efficiently reverse the direction of robotic leg. 15
2-3 Galloping cat as photographed by Eadweard Muybridge. Frame 7-15 of plate 128. Reprinted with permission from Dover Publications. 16

3-1 One legged hopping robot used in simulation. The robot possess a revolute hip joint with an articulated knee and springy foot. 19
3-2 Mass-spring system representing a simple one-legged hopping robot of figure 3-1 in the fully extended knee configuration. For the ideal model, the mass is assumed to be a point mass, while the spring is assumed to be massless and lossless. This model will provide the maximum energy storage possible. 19
3-3 Maximum spring compression calculated using equation 3.11 with initial conditions $x(0) = 0.0$ and $\dot{x}(0) = \sqrt{2gh}$ where h=0.25 meters, M=7.33 kg, and K is a varying stiffness labeled on each plot above. 21
3-4 Maximum energy storage in foot spring for a mass of 7.33 kg dropped from a height of 0.25 meters. Calculated by combining equations 3.11 and 3.12. A massless spring with no damping is assumed. 22
3-5 Model of one legged hopping robot including the effects of damping, distributed mass of the links, and the unsprung mass of the spring. 23
3-6 Energy storage shown for no losses, adjusted for damping (dashed line), and adjusted for damping, distributed mass, and unsprung mass (bold line). 25
3-7 A simulated robot was dropped from a height of 0.25 meters with the knee fully extended. The solid, curved line shows the theoretical energy storage in the spring as calculated for figure 3-6. The solid least-squares line was fit to discrete data points ('o') which represent the energy storage at spring stiffnesses of 2500 N/m, 5000 N/m, 7500 N/m, and 10000 N/m. The dashed, curved line shows the theoretical energy losses in the spring. The dashed least-squares line was fit to discrete data points ('o') which represent the energy losses at spring stiffnesses of 2500 N/m, 5000 N/m, 7500 N/m, and 10000 N/m. 26
3-8 Theoretical and simulated spring efficiencies as calculated from the ratio of energy lost in damping (i.e. energy absorbed by the knee in simulation) and energy stored in the spring. 27
3-9 Spring seen by the foot during touchdown with a bent knee. 28
3-10 Geometry used to calculate virtual spring stiffness. 29
3-11 Virtual spring stiffness as produced by a constant knee torque. The virtual spring behaves like a softening spring as the virtual spring compression increases. 29
3-12 Schematic model of virtual spring in series with foot spring. 30
3-13 Relationship among force-velocity-compression for the virtual spring. Virtual spring force is shown in the vertical axis while spring compression and velocity are shown on the two horizontal axis. View I shows the relationship between Virtual Spring Force and Velocity and can be seen in figure 3-15. View II shows the relationship between Virtual Spring Force and Compression and can be seen in figure 3-14. 31

3-14 Two-dimensional projection of Force versus Compression of virtual spring as seen from View II in figure 3-13. Virtual spring stiffness is shown about the same operating point for motor speeds of 6000, 7000, and 8000 RPM. 32
3-15 Two-dimensional projection of Force vs. Compression of virtual spring as seen from View I in figure 3-13. A dashed rectangle shows the normal operating region. 32
3-16 Energy storage in the foot spring and virtual spring while hopping at 0.25 meters with a bent knee. 33
3-17 Force seen by the spring during touchdown with a bent knee. 34
3-18 Energy storage for the fully extended knee and bent knee. For the bent knee case, energy storage is the sum of foot spring storage and virtual spring storage. 36
3-19 Theoretical stance time for one legged robot with a springy foot of stiffness from 500 N/m to 12000 N/m calculated using equation 3.33. 37
3-20 Theoretical and simulated stance time for one legged robot with a springy foot. The top most figures shows the simulated data 'x' recorded for a fall from 0.25 meters on a fully extend knee. The theoretical data 'o' was calculated using equation 3.33 with $K_{eff} = K_{foot}$. The middle figure shows the simulated data recorded for a vertical fall from 0.25 meters onto a slightly bent knee. The theoretical data 'o' was calculated using equation 3.33 with $K_{virt} = 6000N/m$. The bottom figure shows the simulated data recorded for a running simulation. The theoretical data 'o' was calculated using equation 3.33 with $K_{virt} = 6000N/m$. 38
3-21 Theoretical maximum running speed for one legged at varying effective spring stiffnesses calculated using equation 3.37. 40

4-1 Standing upright with and without a knee cap. Left shows an unstable buckling configuration. Center, shows a stable but inefficient configuration. Right shows a stable and efficient configuration 43
4-2 Swing leg behavior shown with an articulated knee. An initial torque impulse at the hip swings the thigh forward. The knee joint naturally breaks and the shin swings toward the body to conserve angular momentum. Just before touchdown, a torque impulse is given at the hip joint to bring the thigh to a sudden stop. The shin continues to swing forward eventually being stopped by the knee limit stop. 44
4-3 Pendulum driven by a rotary motor. This model is used to represent a typical robot joint and link. 44
4-4 Pendulum driven by a rotary motor through a gear reduction. The gear reduction is necessary on most robot joints to increase the torque and reduce the speed of the motor. 45
4-5 Position control of motor inertia in an attempt to compensate for the effects of reflected inertia. The angular position of the motor is sensed and compared to the angular position of the pendulum. Any discrepancy between the two measurements will result in a torque on the motor which accelerates the motor inertia toward the position of the pendulum. 46
4-6 A torsional spring has been placed in series between the load (i.e. the pendulum) and gear reduction. This is known as Series Elasticity and decouples pendulum from motor inertia, allowing for compensation of reflected inertia. Additionally, Series Elasticity can provide accurate torque control of joints and high shock tolerance. 47

5-1 Joint schematic of a planar quadruped with two degrees of freedom in each leg, a passive spring at the foot, and a boom to allow body pitch but prevent tipping from side to side (frontal plane). 50
5-2 Photograph of the experimental robot. Rear leg on the left and front leg on the right. 50

5-3 Photograph of planar robot with arrows showing degrees of freedom achievable when attached to a boom. 51
5-4 Photograph of planarizing boom attached to the two legged experimental robot. . . 51
5-5 Photograph showing the range of motion for the rear knee joint. 52
5-6 Photograph of showing the range of motion for the rear hip joint. 53
5-7 Photograph rear leg showing the knee motor location with respect to the hip joint. . 54
5-8 Photograph of the Raibert quadruped showing prismatic knee joints. 54
5-9 Photograph of the Cheetah Foot used for smooth energy storage and transfer during running. A string potentiometer is attached to measure compression of the spring. . 55
5-10 Computer rendering of series elastic actuator to be used in the robot. 57
5-11 Schematic representation of the electronics system of the experimental robot. 58
5-12 Photograph of electrical components of the robot. Starting in the top left and moving clockwise: Integrated A/D, D/A, and DSP, motor amplifier, force control board/signal conditioner, analog break out board, power board. 59

6-1 Representation of equivalent force systems. A single force (shown in the left figure) can be described by an equivalent force and moment at any location on the beam (shown in the right the figure). 61
6-2 Representation of equivalent force systems. A vector forces at the center of mass can be described by torques at the joints. The Jacobian is need to transform forces at the center of mass to joint torques. 61
6-3 Schematic of the kinematic configuration of the experimental robot. θ_a, θ_k, and θ_h represent the ankle, knee and hip angles respectively. θ_w, θ_e, and θ_s represent the wrist, elbow, and shoulder angles respectively. ϕ_{arm} and ϕ_{leg} is the sum of the joint angles of the arm and leg respectively. L_1 is the lower arm and leg length. L_2 is the upper arm and leg length. L_3 is the length from the center of mass to both the hip and shoulder. 62
6-4 Depiction of an alternate location for virtual forces. 67
6-5 Schematic of the kinematic configuration of the experimental robot. θ_k and θ_h represent the knee and hip angles respectively. θ_e and θ_s represent the elbow and shoulder angles respectively. R_{arm} and R_{leg} is the line connecting the hand to the shoulder and the foot to the hip, respectively. α_{arm} and α_{leg} are the angles measured from vertical to R_{arm} and R_{leg}, respectively. L_1 is the lower arm and leg length, L_2 is the upper arm and leg length, and L_3 is the distance from the hip and shoulder to the center of mass. 68
6-6 Schematic used to transform leg forces into body forces. 70

7-1 Experimental robot with virtual springs and dampers controlling f_x, f_z, and f_ϕ. The virtual springs and dampers hold the robot solidly in a standing posture, but are compliant enough to withstand position and force disturbances. 73
7-2 Control flow of standing algorithm. *Robot State Information* enters the High-Level Controller where the state machine determines the *state* of the robot. Based on the *state*, some control action calculates the *desired force*. Next, the *desired force* enters the Low-Level Controller, where the *joint torques* are calculated using the Jacobian. Finally, the *joint torques* are applied to the robot and new *robot state information* is gathered. 75
7-3 Left column shows simulated X step response of 0.05 meters with $k_x = 1000$, $b_x = 100$. Middle column shows simulated Z step response of 0.05 meters with $k_z = 1000$, $b_z = 50$. Right column shows simulated pitch step response of 0.1 radians with $k_phi = 100$, $b_phi = 10$. Joint torques and associated joint power are also shown for the knee, elbow, hip and shoulder in each column. 76

7-4 Left column shows simulated X frequency response at an amplitude of 0.05 meters and frequency of 0.5 hertz with $k_x = 100$, $b_x = 10$. Middle column shows simulated Z frequency response at an amplitude of 0.05 meters and frequency of 0.5 hertz with $k_z = 1000$, $b_z = 50$ Right column shows simulated pitch frequency response at an amplitude of 0.1 radians and frequency of 0.5 hertz with k_phi=100, $b_phi = 10$. Joint torques and associated joint power are also shown for the knee, elbow, hip and shoulder in each column. 78
7-5 Left column shows experimental X step response of 0.05 meters with $k_x = 1000$, $b_x = 100$. Middle column shows experimental Z step response of 0.05 meters with $k_z = 1000$, $b_z = 50$. Right column shows experimental pitch step response of 0.1 radians with $k_phi = 100$, $b_phi = 10$. Joint torques and associated joint power are also shown for the knee, elbow, hip and shoulder in each column. 79
7-6 Left column shows experimental X frequency response at an amplitude of 0.05 meters and frequency of 0.5 hertz with $k_x = 100$, $b_x = 10$. Middle column shows experimental Z frequency response at an amplitude of 0.05 meters and frequency of 0.5 hertz with $k_z = 1000$, $b_z = 50$. Right column shows experimental pitch frequency response at an amplitude of 0.1 radians and frequency of 0.5 hertz with k_phi=100, $b_phi = 10$. Joint torques and associated joint power are also shown for the knee, elbow, hip and shoulder in each column. 81
7-7 State machine used in pronking algorithm. 82
7-8 x, z, and ϕ position of simulated robot over five pronking cycles. 88
7-9 Data from single pronking cycle for simulated robot. 90
7-10 Torque data over five pronking cycles for simulated robot. 91
7-11 Power data over five pronking cycles for simulated robot. 92
7-12 x, z, and ϕ position of experimental robot over five pronking cycles. 93
7-13 Power data over five pronking cycles for experimental robot. 94
7-14 Torque data over five pronking cycles for experimental robot. 95

List of Tables

5.1 Weight distribution of planar quadruped. 53
5.2 Detailed joint information 56
5.3 Physical Properties of Series Elastic Actuator. 57

7.1 Summary of Torque and Power Requirements for Simulated Step Responses 75
7.2 Summary of Torque and Power Requirements for Simulated Frequency Responses . . 77
7.3 Summary of Torque and Power Requirements for Experimental Step Responses . . . 77
7.4 Summary of Torque and Power Requirements for Experimental Frequency Responses 80
7.5 State Machine with Sensor Values Indicated 83
7.6 State Machine and Associated Control Actions 87

Chapter 1

Introduction

Humans have always admired the strength, agility, endurance, and speed of quadrupedal animals. Anyone who has been to a horse track, seen a cheetah run down a gazelle, or witnessed a deer leap over a fence would not likely argue the superhuman quality of these ambulatory feats. In the last century, scientist have begun trying to understand how animals, and quadrupeds in particular, are capable of such tremendous physical challenges, both from a power and control standpoint. By designing and building a robot which attempts to mimic these animals, we hope to better understand the energetics and control of locomotion.

1.1 Goals of Thesis

This thesis addresses the design and control a two legged planar robot to study the quadrupedal locomotion. By restricting the motion of the robot to a plane, quadrupedal locomotion can be closely approximated without the added complexity of four legs and control of balance in three dimensions. The purpose of this robot is four fold:

1. To investigate the use of springs in the feet for smooth and efficient means of energy transfer.
2. To investigate the use of natural dynamics in the swing leg of the robot to achieve high efficiency.
3. To develop intuitive control algorithms which can easily be scaled to a three dimensional quadrupedal robot.
4. To determine the torque and power requirements for quadrupedal running.

1.2 Summary of Thesis

- *Chapter 2* investigates biological mechanisms used in running and walking and reviews literature which documents the use of biological mechanisms in actual robots. Design recommendations to include natural dynamic mechanisms and springs are made.
- *Chapter 3* develops guidelines for selecting foot spring stiffness of running robots based on efficiency and controllability of a one legged hopping robot.
- *Chapter 4* explains how to design and implement natural dynamic elements in running robots.
- *Chapter 5* details the design and construction of the experimental robot.
- *Chapter 6* describes Virtual Model Control, comparing it to a puppeteer and a marionette puppet. A Virtual Model Controller is derived for the experimental robot.

- *Chapter 7* describes algorithms used to control standing and pronking for simulation and laboratory experiments. Standing and pronking experiments are conducted on the experimental robot and power requirements to achieve running are set forth.
- *Chapter 8* summarizes the findings of this thesis and proposes several areas for future work.

Chapter 2

Design Inspiration

The design and construction of a robot to study quadrupedal locomotion is a challenging task. The success of such a project requires an understanding of engineering principles and robotics. An understanding of biology is also important because it provides insight into proven mechanical designs and methods of control.

Incorporating mother nature's design elements, such as springs and natural dyanmics, into legged robots is not a new idea. In fact, there are many examples of legged robots using biological mechanisms to efficienctly walk, run, and even do gymnastics. In this chapter, we investigate the basic biological principals of efficient locomotion and review a few walking and running robots that utilize natural mechanisms to achieve efficient and natural movements. This information is used to generate basic design requirements for the experimental robot to be built.

2.1 Energetics of Walking

A natural starting point for studying locomotion is the bipedal gait. Specifically, we begin by studying the energetics of a simplified human gait known as the compass gait. The compass gait model consists of two rigid, massless legs attached at the hip with a revolute joint. A point mass is modeled at the hip joint. With only one degree of freedom located at the hip, the body is forced to follow an arc defined by the length of the leg. The compass gait can be seen in figure 2-1.

Since only one foot is in contact with the ground during stance and the body is considered to be a point mass. This gait can be modeled as a simple inverted pendulum. Calculating the external work on the model, namely the potential and kinetic energy of the system, we find

$$PE = MgLcos\theta \tag{2.1}$$

$$KE = MgL(1 - cos\theta) \tag{2.2}$$

Graphing the equations 2.1 and 2.2 in figure 2-1, the cyclical nature of potential energy and kinetic energy can be seen. Note that the kinetic energy is maximum when the potential energy is minimum and vice versa. Similar cyclical transition of potential and kinetic energy was recorded in human subjects by McMahon (1984) using force plate data to calculate the changes in mechanical energy of the body's center of mass.

Intuitively, this exchange of kinetic and potential energy has been likened to an egg rolling end over end [Margaria (1976)]. The egg will speed up and slow down as it gains and losses potential energy. Energetically, this means that no external work is performed if no energy is lost to friction, thus a perfectly efficient gait can be achieved. Of course friction is present in our joints, muscles, and ground contact, so there will be some losses.

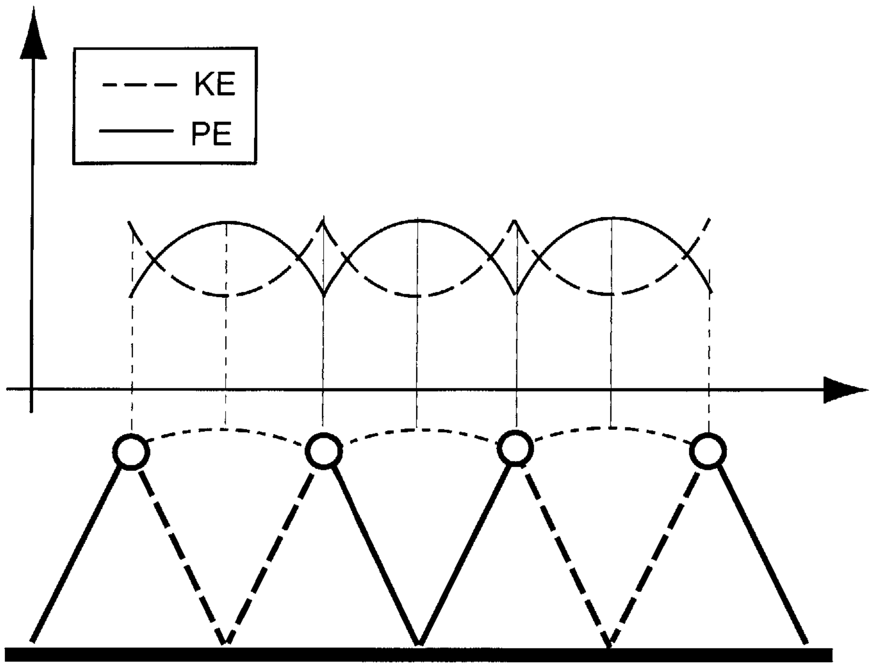

Figure 2-1: Several strides of the compass gait shown with qualitative potential and kinetic energy curves.

Although this model lacks many of the most basic human movements, such as knee and ankle flexion and extension and pelvic tilt and rotation, it is analytically valuable because it demonstrates the cyclical nature of potential and kinetic energy during walking. It is this key feature that makes human walking efficient.

2.2 Energetics of Running

It is known from oxygen consumption measurements on animals that running can be as efficient as walking [McMahon (1984)]. From the study of walking in section 2.1, it can be inferred that efficient running requires a similar cyclical transfer of energy. However, running is dynamically dissimilar to walking for both humans and quadrupeds. Running is defined by a phase when all feet leave the ground. With a little thought, one can see that potential energy and kinetic energy are no longer 180 degrees out of phase as they were in walking. In fact, during running potential and kinetic energy reach a maximum at the peak of the ballistic phase and a minimum sometime during touchdown.

Since the animal cannot transfer kinetic and potential energy back and fourth as was the case in human walking, some other mechanism must be used to store energy and achieve efficient running gaits. In fact, animals use their muscles and tendons to store elastic energy at touchdown and release it before lift-off. In this way, animals reduce the external work required during running. With the power limitations of todays motors, it is imperative to design similar elasticity into our robot if we want it to be able to run at all.

It is believed that animals not only use elastic storage at touchdown, but also during other phases of running. For example, Alexander (1990) believes that it is feasible that animals use elastic storage elements to quickly and efficiently reverse the angular velocity of the leg while transitioning from stride to swing and vice versa (see figure 2-2). Alexander (1990) also believes that animals may use soft foot pads to reduce the high impact forces experienced while running. Furthermore, foot pads

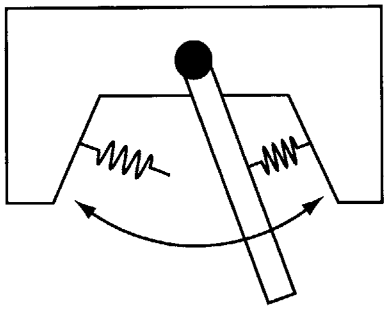

Figure 2-2: Springs used to efficiently reverse the direction of robotic leg.

may also be used to reduce high frequency vibrations of the foot on the ground, known as chatter. Clearly, springs play an important role in running of animals.

Springs have been used as energy storage devices in several robots to date. Most notable were the use of pnuematic springs in one, two and four legged running and hopping robots by Raibert [Raibert. (1986)]. Carrying this idea forward, Buehler (1996) developed a monopod with a springy foot, similar to the one used in Raibert's one legged hopper, and hip return springs, as in figure 2-2. The monopod had an average power consumption of only 125W, making it much more efficient than its predecessors.

2.3 Natural Dynamics

Another important mechanism that is used by animals to achieve efficient running and walking gaits is natural dynamics. Natural dynamics describe the use of natural, unactuated motions to achieve controlled movements. As a simple example, a pendulum clock uses its natural dynamics, dictated by the mass and length of the pendulum and the local gravity constant, to keep track of time. The pendulum is given tiny pulses of energy at the both extremities of the swing when the velocity is zero. Between these pulses, the natural natural dynamics of the system take over and move it to the next state. In this way, a pendulum clock is a very efficient mechanism.

This example is not dissimilar from the way we use natural dynamics to walk. At the beginning of swing phase, we give our hip an impulse of energy and then let the natural pendulum motion carry it forward to extension. Just before extension, we give the hip another tiny impulse of energy in the opposite direction to bring it to a sudden halt before touchdown. Electromyographic data records by McMahon (1984) show little electrical activity in the leg muscles of humans during the swing phase at normal walking speeds. This, of course, suggests that the leg is swinging freely during this period. As a sanity check, electromyographic records show significant electrical activity in leg muscles during stance.

Natural dyanmics have also found there way into many robots. Much research has been done on completely passive walking machines [Garcia et al. (1998), Adolfsson et al. (1998), Fowble & Kuo (1996), McGeer (1990)]. Passive walking robots use Earth's gravity as a power supply and rely on special geometry to achieve gaits similar to the compass gait described in figure 2-1. By exploiting natural dynamic elements, a robot can achieve a high overall efficiency. In the limit, a robot can rely entirely on natural dynamic motions and thus require no external power source. Natural legged robots such as these have been constructed and successfully demonstrated.

Natural dynamics were successfully implemented in actuated robots as well. Pratt & Pratt (1999) developed algorithms that use the natural dynamics of the swing leg to efficiently control a planar bipedal robot. More recently, Pratt has developed three-dimensional simulations of a bipedal walking robot using natural elements such as a knee stop, compliant ankle, and passive swing leg to

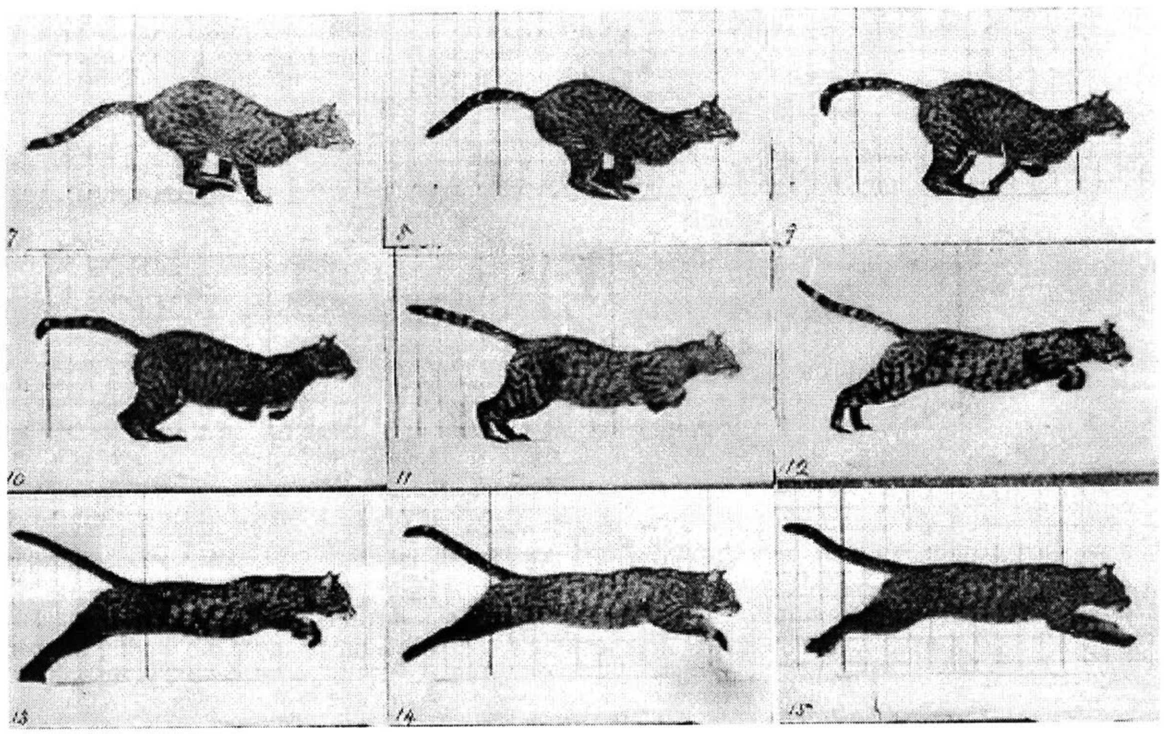

Figure 2-3: Galloping cat as photographed by Eadweard Muybridge. Frame 7-15 of plate 128. Reprinted with permission from Dover Publications.

achieve natural looking and efficient walking.

Gymnastic maneuvers have also been accomplished in several robots using natural dynamic principals. Hodgins (1987) used natural dynamics to get a two legged planar robot to do a forward flip while running. Playter (1995) used the natural dynamics of springy arms to successfully control layout somersaults of a wooden doll.

2.4 Other Important Biological Elements

Yet another biological mechanism that is important to the animal during running is back flexion. Back flexion is shown dramatically in photographs by Muybridge [Muybridge (1957)] in figure 2-3. By flexing and extending its back, the cat in the picture is gaining two advantages. First, it is significantly lengthening its stride length and thus increasing its speed. Second, it has been pointed out that back flexion allows the animal to place its foot closer to the center of gravity projection of the body [Raibert. (1986)]. This feature gives the animal a higher degree of stability when only one foot is contacting the ground. It has also been suggested that the back may be acting as an elastic storage device as we discussed in 2.2.

Little research has been done in this area, with the exception of Leeser (1996) who used an articulated spine to successfully control a planar quadrupedal robot.

2.5 Design Recommendations

From this literature survey, one can clearly see that a cyclical energy transfer mechanism is imperative for efficient running. As such, a springy element, which will provide smooth energy transfer during running will be designed into the experimental robot. Spring selection will be discussed in chapter 3 Also, it is apparent that natural dynamics play an important role in efficient and natural locomotion, thus natural dynamic mechanisms will be considered. Designing natural dynamic elements will be detailed in 4. Finally, the literature review also suggest that a springy back can potentially provide more energy storage and natural movements. However, because of the complexity such an element would add to the design, it has been delayed for possible future work.

Chapter 3

Spring Selection

From the study of the energetics of walking and running, as reviewed in Chapter 2, we know that smooth energy transfer is critical for efficient locomotion. In walking, the energy is naturally cycled between kinetic and potential energy due to the geometry of walking gaits. During running, potential and kinetic energy are nearly in phase, requiring a much different energy transfer mechanism. Springs, in the form of tendons, and to some extent muscles, provide the compliance necessary for smooth storage and transfer of kinetic and potential energy during running.

Designing compliance into running robots is not an easy task. Tendons are highly efficient, light weight springs with non-linear properties. Furthermore, animals have the ability to change their compliance depending on environmental conditions. Clearly, we will be unable to replicate such complex behavior with the springs available to us. However, with a little thought, we can make wise decisions when selecting a spring that will result in a robot that is both energy efficient and controllable.

In this chapter we investigate the choice of linear spring stiffness for both efficiency and controllability of a one legged hopping and running robot. A one legged robot is simulated because its energetics and control can easily be applied to bipedal and quadrupedal robots [Raibert. (1986)]. This task can be divided into two phases: modeling and simulation.

In the modeling phase, we develop an ideal mass-spring model to understand energy storage in springs and place an upper bound on the maximum energy storage possible. Next, a realistic model is developed which includes spring losses and mass distribution. This mass-spring-damper system closely models the one legged robot used in the simulation phase. In this way, the realistic model provides base-line data to insure the validity of data collected from our simulated robot.

After a thorough understanding of expected spring behavior is developed, simulation experiments are conducted on a one legged hopping robot with a springy foot and articulated knee. A schematic of the one legged robot can be seen in figure 3-1. First, a simulation experiment is conducted to check for agreement between the mathematical model and the simulated robot. After verification, the efficiency of the robot is tested during a hopping experiment. Next, controllability is investigated during hopping and running experiments. Based on these results, recommendations are made regarding the selection and efficient use of springs in running and hopping robots.

3.1 Modeling

3.1.1 An Ideal Model

As a first pass at modeling the one legged robot, the spring in the foot is assumed to be massless and lossless. The ideal model can be seen in figure 3-2. From this model, we can calculate the maximum energy savings that can be realized with springs in running robots.

The total energy at any moment in time can be written as

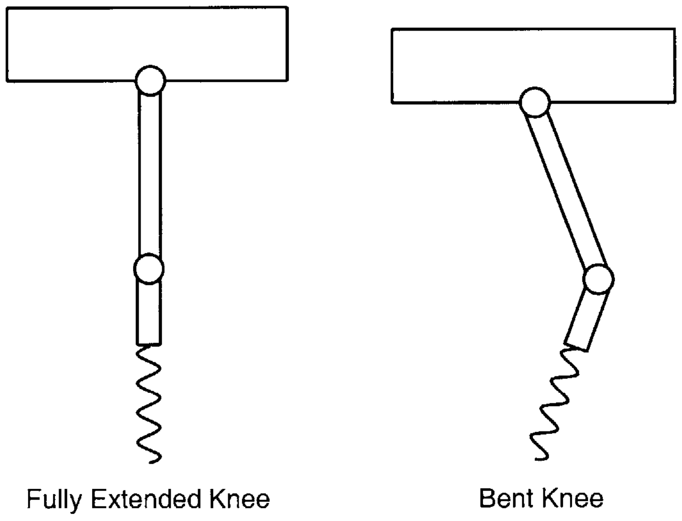

Figure 3-1: One legged hopping robot used in simulation. The robot possess a revolute hip joint with an articulated knee and springy foot.

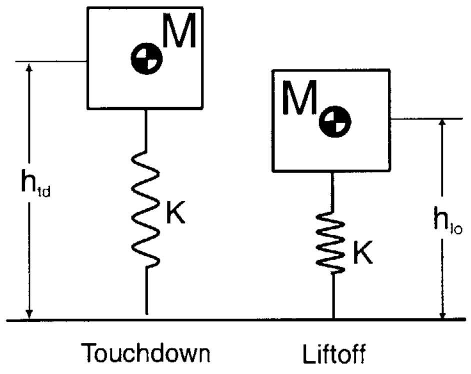

Figure 3-2: Mass-spring system representing a simple one-legged hopping robot of figure 3-1 in the fully extended knee configuration. For the ideal model, the mass is assumed to be a point mass, while the spring is assumed to be massless and lossless. This model will provide the maximum energy storage possible.

$$E_{total} = E_{potential} + E_{kinetic} + E_{spring} \tag{3.1}$$

Just before the foot of the simplified model touches the ground the total energy is

$$E_{td} = Mgh_{td} + \frac{1}{2}Mv_{td}^2 + 0 \tag{3.2}$$

Just before the robot begins lift-off, the foot spring is fully compressed and vertical velocity is zero, resulting in a total energy of

$$E_{lo} = Mgh_{lo} + 0 + \frac{1}{2}Kx_{lo}^2 \tag{3.3}$$

Since energy is conserved, we can equate equations 3.2 and 3.3 and solve for the energy storage in the spring.

$$\frac{1}{2}Kx_{lo}^2 = Mg(h_{td} - h_{lo}) + \frac{1}{2}Mv_{td}^2 \tag{3.4}$$

From equation 3.4, it can be seen that the maximum energy storage in the spring is equal to the kinetic energy just before touchdown and the change in potential energy from touchdown to lift-off. If no foot spring is present, all of this energy will be dissipated by the ground. Immediately, we can see the significant energy savings springs provide.

Equation 3.4 provides an qualitative value of energy storage, but it is useful to have a quantitative measure of the maximum spring storage. To do this, we need to solve the second-order differential equation which represents the spring-mass model in figure 3-2.

$$M\ddot{x} = -Kx + Mg \tag{3.5}$$

The solution is of the form

$$x = C_1 sin(\omega_n t) + C_2 cos(\omega_n t) + \frac{Mg}{K} \tag{3.6}$$

where

$$\omega_n = \sqrt{\frac{K}{M}} \tag{3.7}$$

Taking the derivative of equation 3.6 yields

$$\dot{x} = C_1 \omega_n cos(\omega_n t) - C_2 \omega_n sin(\omega_n t) \tag{3.8}$$

Using the initial conditions

$$x(0) = 0 \tag{3.9}$$
$$\dot{x}(0) = u = \sqrt{2gh} \tag{3.10}$$

and substituting $x(0)$ and $\dot{x}(0)$ into equation 3.6 and 3.8 respectively, and solving for C_1 and C_2 yields

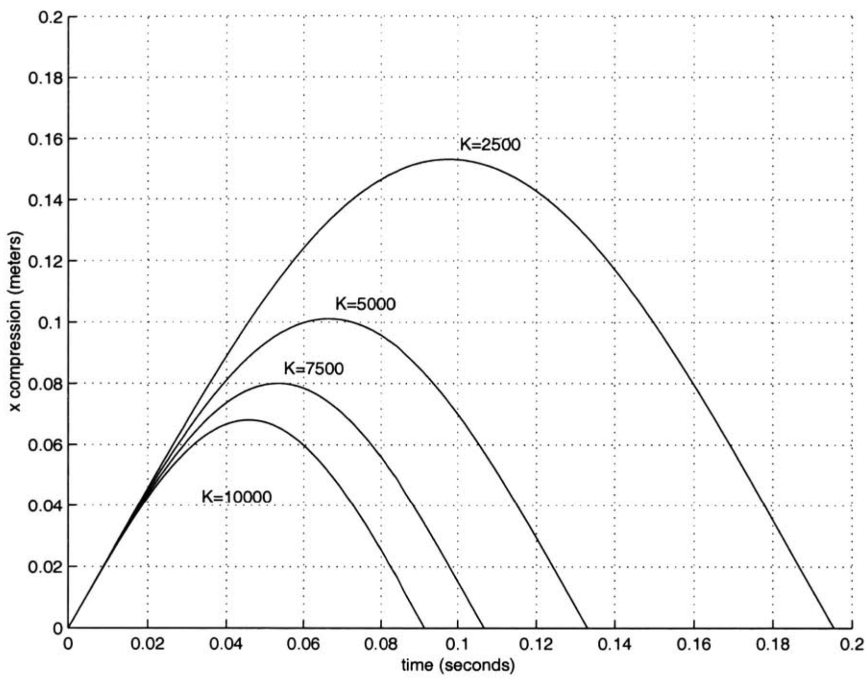

Figure 3-3: Maximum spring compression calculated using equation 3.11 with initial conditions $x(0) = 0.0$ and $\dot{x}(0) = \sqrt{2gh}$ where h=0.25 meters, M=7.33 kg, and K is a varying stiffness labeled on each plot above.

$$x = \frac{\sqrt{2gh}}{\omega_n} sin(\omega_n t) + \frac{Mg}{K}(1 - cos(\omega_n t)) \tag{3.11}$$

From equation 3.11, we can calculate the compression of the spring for a range of spring constants. This data can be seen in figure 3-3. It is not surprising that softer spring constants result in larger spring compression.

The energy storage of a spring can be written as

$$E_{spring} = \frac{1}{2}Kx^2 \tag{3.12}$$

Again, using equation 3.11 to calculate the maximum spring compression and substituting into equation 3.12 we find the maximum energy storage in the spring. This value has been plotted for spring stiffness up to 12000 N/m in figure 3-4. It should be noted that since there are no losses in the system, figure 3-4. represents the *maximum* energy storage possible in the spring. To check the validity of this result, equation 3.4 has also been plotted in figure 3-4. It cannot be seen because it is coincident with the maximum energy stored in the spring.

At first glance, intuition tells us that since there are no losses in our theoretical model, all springs stiffnesses should store the same amount of energy. However, closer investigation of equation 3.4 reveals that the energy stored in the spring is a function of spring stiffness. Consider two cases:

1. For a stiff spring, the maximum compression will be small, thus the difference between h_{td} and h_{lo} will be small, resulting in a small ΔPE.

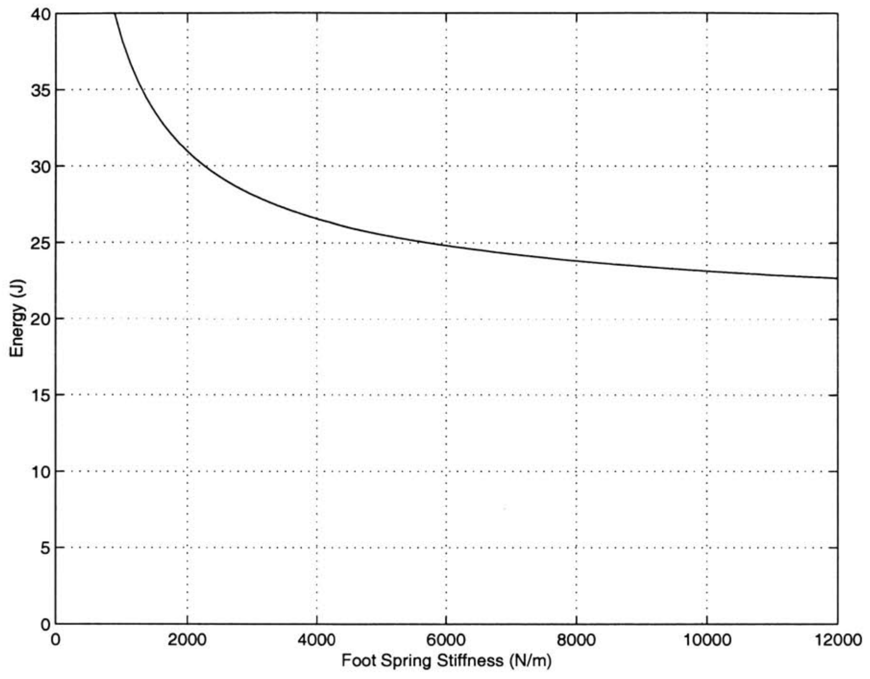

Figure 3-4: Maximum energy storage in foot spring for a mass of 7.33 kg dropped from a height of 0.25 meters. Calculated by combining equations 3.11 and 3.12. A massless spring with no damping is assumed.

2. For soft spring, the maximum compression will be large, thus the the difference between h_{td} and h_{lo} will be large, resulting in a large ΔPE and more energy storage.

Since the kinetic energy of the system just before touchdown is constant, and if we assume equal efficiency for all spring stiffnesses, then it follows that as stiffness increases, spring energy storage will decrease. This conclusion agrees with the data seen in figure 3-4. From here forward, this effect will be known as the ΔPE Effect.

The question remains: how does spring storage relate to efficiency of the system? Initially, one might presume that more energy storage in the foot spring results in a more efficient spring-mass system. In fact, this is not the case. Even though a soft spring stores more energy than a stiff spring, all masses will rebound to the same height because the greater energy storage in the soft spring comes at the expense of potential energy. Said another way, the soft spring stores more energy during touchdown, but must provide that extra energy during lift-off to rebound to the original height.

To summarize the important features of the ideal model:

- Springs are capable of storing all of the kinetic energy and change of potential energy of a spring-mass system.
- Kinetic energy is dependent only on initial height, not spring stiffness.
- Potential energy storage is larger for soft springs than for stiff springs. This has been dubbed the ΔPE Effect.
- As a result of the ΔPE Effect, the total energy storage will decrease with increasing spring stiffness as seen in figure 3-4.

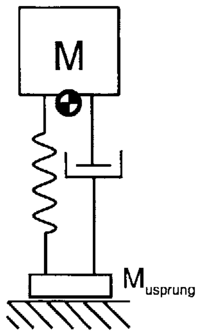

Figure 3-5: Model of one legged hopping robot including the effects of damping, distributed mass of the links, and the unsprung mass of the spring.

- Even though energy storage decreases as stiffness increases, the energy efficiency of the system, in this case a spring and mass, is independent of spring constant.

3.1.2 A Realistic Model

The lossless and massless model of section 3.1.1 is an excellent starting point to begin studying the use of compliance in running robots. It provides an upper-bound on the amount of energy savings that can be realized in a robot with compliance and provides insight into basic spring behavior. Unfortunately, because the simplified mass-spring model ignores the distributed mass and losses of our simulated robot, it cannot provide an accurate prediction of spring and robot behavior. The simulated robot, as can be seen in figure 3-1, is significantly different than the simple spring-mass model of figure 3-2 in three ways.

1. The simulated robot has damping due to friction at the knee joint and losses in the foot spring. Energy dissipation due to damping will result in less than maximum energy storage.
2. The mass of the simulated robot is distributed in the links and the body. Distribution of the mass in the leg and spring will lower the center of gravity of the robot, decreasing its potential energy as compared to the point mass in the first mass-spring model of figure 3-2.
3. Since our spring is no longer massless, we must include the losses due to its unsprung mass. Unsprung mass is defined as the mass which comes instantly to zero velocity when striking the ground. The robot will lose all of the kinetic energy associated with the unsprung mass when it strikes the ground.

These differences can be added to our spring-mass model to provide a more accurate base-line to compare simulation data. Figure 3-5 shows damping, distributed mass, and unsprung mass added to the simple model.

To quantify the effects of the damper, we need to solve a second-order differential equation very similar to equation 3.5

$$M\ddot{x} = -B\dot{x} - Kx + Mg \tag{3.13}$$

The solution is of the form

$$x = C_1 e^{at} sin(\omega_d t) + C_2 e^{at} cos(\omega_d t) + \frac{Mg}{K} \tag{3.14}$$

where

$$\omega_d = \sqrt{\frac{K}{M} - \left(\frac{B}{2M}\right)^2} \tag{3.15}$$

$$a = \frac{-B}{2M} \tag{3.16}$$

Taking the derivative of equation 3.14 yields

$$\dot{x} = C_1 a e^{at} sin(\omega_d t) + C_1 e^{at} cos(\omega_d t) + C_2 a e^{at} cos(\omega_d t) - C_2 e^{at} sin(\omega_d t) \tag{3.17}$$

Using the initial conditions

$$x(0) = 0 \tag{3.18}$$

$$\dot{x}(0) = u = \sqrt{2gh} \tag{3.19}$$

and substituting $x(0)$ and $\dot{x}(0)$ into equation 3.14 and 3.17 respectively, and solving for C_1 and C_2 yields

$$x = \frac{\sqrt{2gh} - \frac{Bg}{2K}}{\omega_d} e^{\frac{-B}{2M}t} sin(\omega_d t) + \frac{Mg}{K}(1 - e^{\frac{-B}{2M}t} cos(\omega_d t)) \tag{3.20}$$

As a check, setting B=0 in equation 3.20 yields equation 3.11.

From equation 3.20 and its derivative, we can calculate the spring energy storage with a damper in the system according to

$$E_{spring} = \frac{1}{2}Kx^2 - B\dot{x}^2 \tag{3.21}$$

Assuming constant damping of 25 kg/s, the energy storage, adjusted for damping, can be seen as the dashed line in figure 3-6 labeled E_{stored} *(Adjusted for Damping)*.

Distributing the mass to the leg and spring of the robot also requires adjustments to our original model. The center of gravity of the robot with distributed mass is 0.055 meters below the original point mass. This reduces the potential energy according to $PE = Mgh$. This accounts for 3.95 Joules. This is a constant effect, so we can just subtract 3.95 Joules from the energy stored at all spring stiffness.

The unsprung mass has been modeled to be 0.1 kg. Every time the spring strikes the ground the unsprung mass dissipates its kinetic energy to the ground according to $KE = \frac{1}{2}Mv^2$. This accounts for a loss of about 0.25 Joules on each cycle. Again, this can simply be subtracted from the energy stored at all spring stiffness.

Thus, the total adjustment due to the distributed mass and unsprung mass is a loss 4.20 Joules. This has been subtracted from energy storage curve adjusted for damping. The final result can be seen in figure 3-6 as the bold line labeled E_{stored} *(Adjusted for Damping, Distributed Mass, and Unsprung Mass)*. This curve will provide the baseline to verify our simulation results.

To summarize the important features of the realistic model:

- The energy storage curve for the realistic model is very similar in shape to the energy storage curve for the ideal model, but is shifted down approximately 5.0 Joules to account for damping and an additional 4.2 Joules to account for distributed mass in the links and the unsprung mass of the spring.

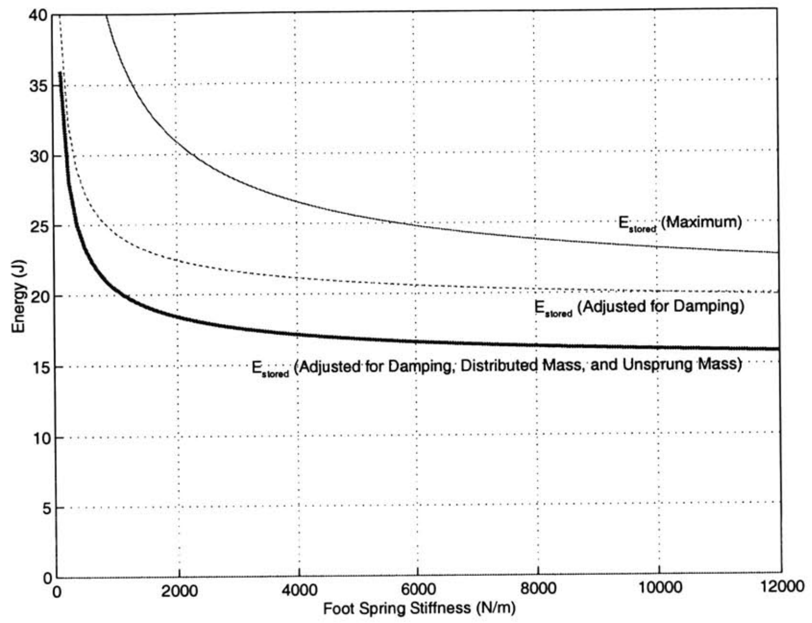

Figure 3-6: Energy storage shown for no losses, adjusted for damping (dashed line), and adjusted for damping, distributed mass, and unsprung mass (bold line).

3.2 Simulation

3.2.1 Verification of Simulation

To verify the agreement of the realistic model and the simulated robot, a simple simulation experiment is conducted. The simulated robot is dropped from a height of 0.25 meters with a fully extended knee onto foot spring stiffnesses of 2500 N/m, 5000 N/m, 7500 N/m, and 10000 N/m and constant damping term of b=25 kg/s. With these initial conditions, the simulation data should match the realistic model well. The results of the simulation experiment can be seen in figure 3-7.

Reasonable agreement can be seen between the theoretical energy storage and the least squares fit to the simulated energy storage. Discrepancies between the two models can be explained by errors caused by estimations in the distributed mass of the robot. Excellent agreement can be seen between the theoretical energy dissipated and the least squares fit to the simulated energy dissipated. Efficiency, as defined in equation 3.22, is also a good measure to verify the agreement between theoretical and simulated results.

$$Efficiency = 100\left(1 - \frac{E_{losses}}{E_{storage}}\right) \tag{3.22}$$

Efficiency has been graphed in figure 3-8. Nearly perfect agreement can be seen between theoretical and simulated efficiencies.

With this experiment, we conclude that our model is an accurate representation of the simulated robot. From here forward, we will use the least-squares fit lines in figure 3-7 as the base-line to compare future simulation results.

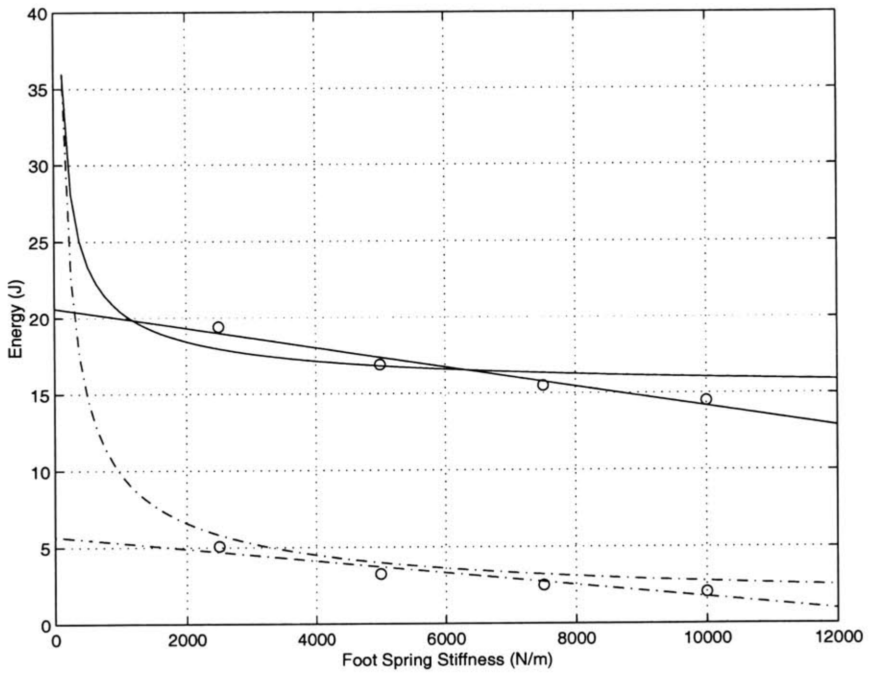

Figure 3-7: A simulated robot was dropped from a height of 0.25 meters with the knee fully extended. The solid, curved line shows the theoretical energy storage in the spring as calculated for figure 3-6. The solid least-squares line was fit to discrete data points ('o') which represent the energy storage at spring stiffnesses of 2500 N/m, 5000 N/m, 7500 N/m, and 10000 N/m. The dashed, curved line shows the theoretical energy losses in the spring. The dashed least-squares line was fit to discrete data points ('o') which represent the energy losses at spring stiffnesses of 2500 N/m, 5000 N/m, 7500 N/m, and 10000 N/m.

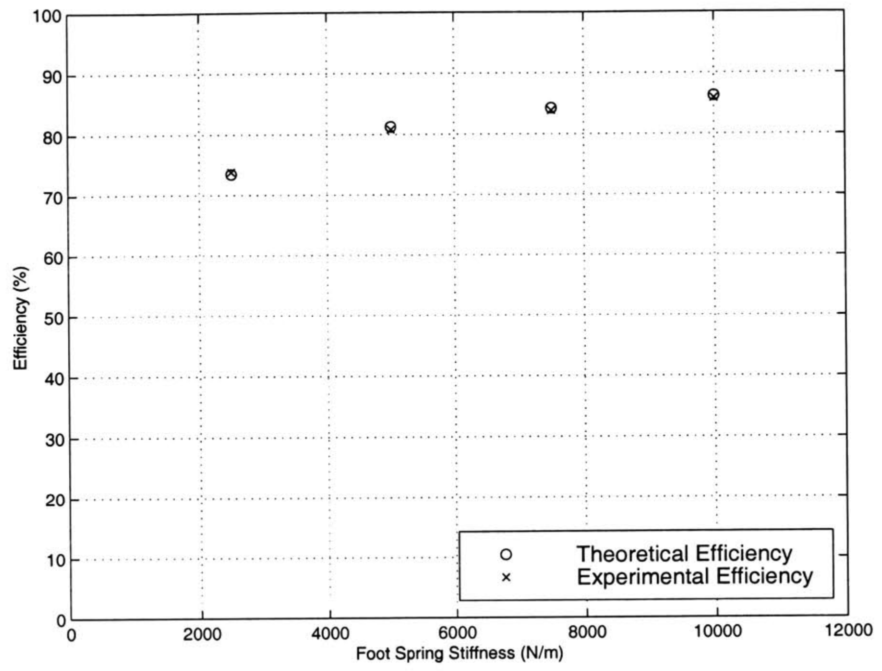

Figure 3-8: Theoretical and simulated spring efficiencies as calculated from the ratio of energy lost in damping (i.e. energy absorbed by the knee in simulation) and energy stored in the spring.

3.2.2 Efficiency Analysis and Simulation

With a clear understanding of expected spring behavior and a verified model of a hopping robot, we can now consider the issue at hand: selection of foot spring stiffness. First, we will consider selection of foot spring stiffness based on efficiency of a hopping robot.

Due to inherent losses in the system, the one legged hopping robot must inject energy into the system to maintain hopping height. This is achieved by thrusting with knee or hip torque during stance. Assuming that the hip torque is used only for adjustment of body pitch, then if the knee is fully extended at touch down, the robot has no way of pushing on the ground to maintain hopping height. Thus, the one legged hopping robot, which uses hip torque only for adjustments in body pitch, *must* land with a bent knee if hopping height is to be maintained. Unfortunately, landing with a bent knee affects the energy storage in the spring in two ways:

- In order to store energy in the foot spring, the bent knee must present a very high impedance to the foot spring. The maximum impedance is limited by speed-torque curves of the motor driving the knee joint. The impedance produced by the knee joint and seen at the foot spring is described by a virtual spring of varying stiffness. This is called the Virtual Spring Effect and is described in detail in section 3.2.3.
- Because of geometry, the bent knee reduces the maximum energy storage possible in the foot spring. This is called the Alpha Effect and is described in detail in section 3.2.4.

3.2.3 The Virtual Spring Effect

First, we consider the affects of the virtual spring on efficiency. A proportional controller at the knee ($\tau_{knee} = K_p\theta_{knee}$) looks like a virtual spring between the foot and the hip. This situation can

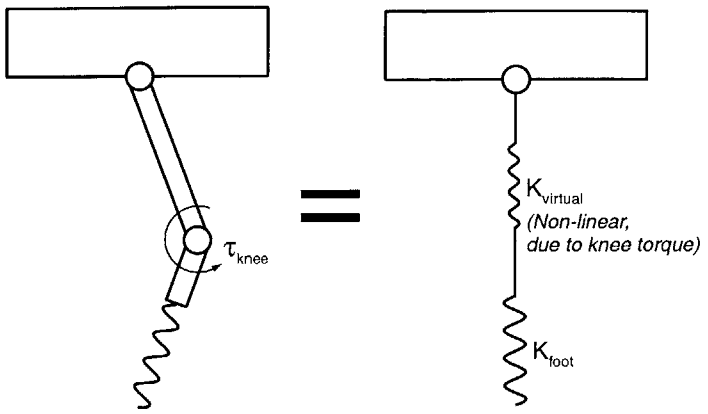

Figure 3-9: Spring seen by the foot during touchdown with a bent knee.

be seen in figure 3-9 The virtual spring stiffness can be calculated using the geometry in figure 3-10. Assuming a constant torque at the knee we find that the virtual spring force is

$$F_{virtual} = \frac{\tau_{knee}}{Lsin\left(\frac{\theta_{knee}}{2}\right)} \tag{3.23}$$

The compression of the virtual spring can be written as

$$x_{virtual} = 2L\left(1 - cos\left(\frac{\theta_{knee}}{2}\right)\right) \tag{3.24}$$

Using equations 3.23 and 3.24, a curve showing the characteristics of the virtual spring behavior has been graphed and can be seen in figure 3-11. We can see that the virtual force approaches infinity at zero spring compression (θ_{knee}=0.0) and sharply decreases as virtual spring compression increases. Thus, for a constant torque the virtual spring behaves like a softening spring as compression increases (θ_{knee} increases).

Two springs of different stiffness in series, as in figure 3-12 will not equally distribute energy storage. Using force equilibrium on the springs we find

$$F_1 = K_1x_1 = K_2x_2 \tag{3.25}$$

Solving for x_1 and x_2 we find

$$x_1 = \frac{F_1}{K_1} \tag{3.26}$$

$$x_2 = \frac{F_1}{K_2} \tag{3.27}$$

The energy storage in each spring is

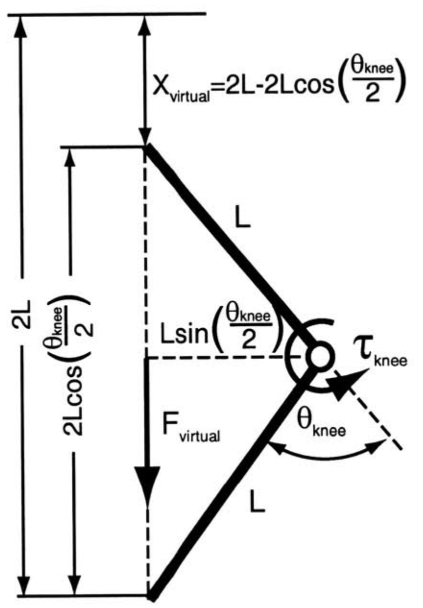
$X_{virtual}=2L-2L\cos\left(\frac{\theta_{knee}}{2}\right)$
L
$2L$
$2L\cos\left(\frac{\theta_{knee}}{2}\right)$
$L\sin\left(\frac{\theta_{knee}}{2}\right)$
τ_{knee}
$F_{virtual}$
θ_{knee}
L

Figure 3-10: Geometry used to calculate virtual spring stiffness.

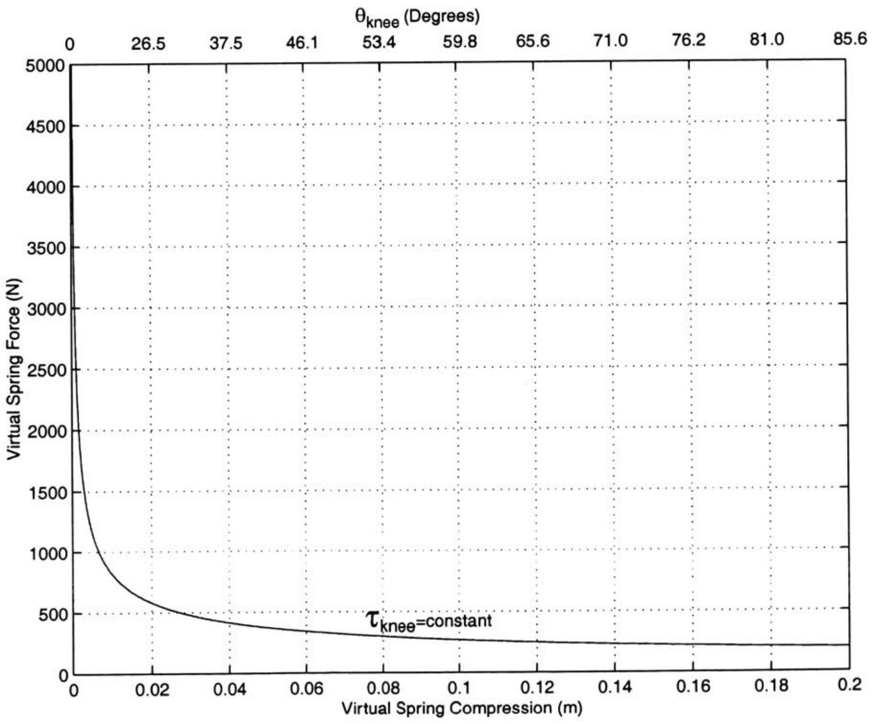

Figure 3-11: Virtual spring stiffness as produced by a constant knee torque. The virtual spring behaves like a softening spring as the virtual spring compression increases.

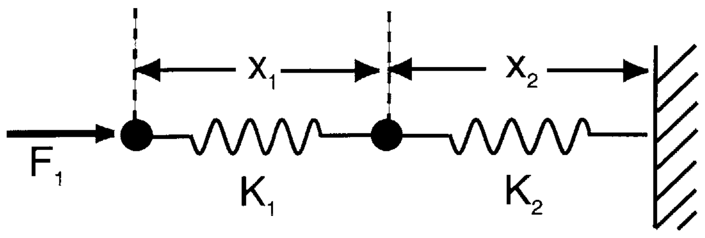

Figure 3-12: Schematic model of virtual spring in series with foot spring.

$$\begin{aligned} E_1 &= \frac{1}{2}K_1 x_1^2 \\ &= \frac{1}{2}K_1\left(\frac{F_1}{K_1}\right)^2 \\ &= \frac{F_1^2}{2K_1} \end{aligned} \tag{3.28}$$

$$\begin{aligned} E_2 &= \frac{1}{2}K_2 x_2^2 \\ &= \frac{1}{2}K_2\left(\frac{F_1}{K_2}\right)^2 \\ &= \frac{F_1^2}{2K_2} \end{aligned} \tag{3.29}$$

From equations 3.28 and 3.29, we can see that the softer of the two springs will store more energy than the stiffer spring. In this way, the foot spring competes with the virtual spring to "store" energy. We say "store" energy when referring to the virtual spring because the virtual spring cannot actually store energy, rather, it can only dissipate energy. Clearly, we want the foot spring to win this competition, but who does win?

To answer this question, we must place a value on the maximum virtual spring stiffness. Virtual spring stiffness is limited by the torque-speed characteristics of the motor which generates the virtual spring. That is to say, the virtual spring may not exceed the power of the motor. The torque-speed curves of the motor have been mapped to force-velocity-compression curve of the virtual spring. A third dimension, spring compression, is needed to map the motor's torque-speed curve to the virtual spring power curve because both the velocity and force of the virtual spring depend on virtual spring compression. Figure 3-13 shows the relationship among force-velocity-compression for the virtual spring. Immediately, we see that virtual spring force decreases with increasing compression and increasing velocity. Intuitively, this makes sense. As compression increases the knee losses mechanical advantage. As velocity increases, force must decrease due to power limitations on the motor.

The virtual spring stiffness at any point can be measured from the slope of the force versus compression curve as seen by viewing from View II in figure 3-13. The two-dimensional projection of View II can be seen in figure 3-14. To deduce the virtual spring stiffness, one needs to specify both a virtual spring compression and an appropriate motor speed curve.

Virtual spring compression is approximately 0.05 meters at touchdown. Recall, it is necessary to land with a bent knee (virtual spring compression > 0.0) in order to be able to provide energy during lift off. This operating point can be seen as a dashed vertical line in figure 3-14

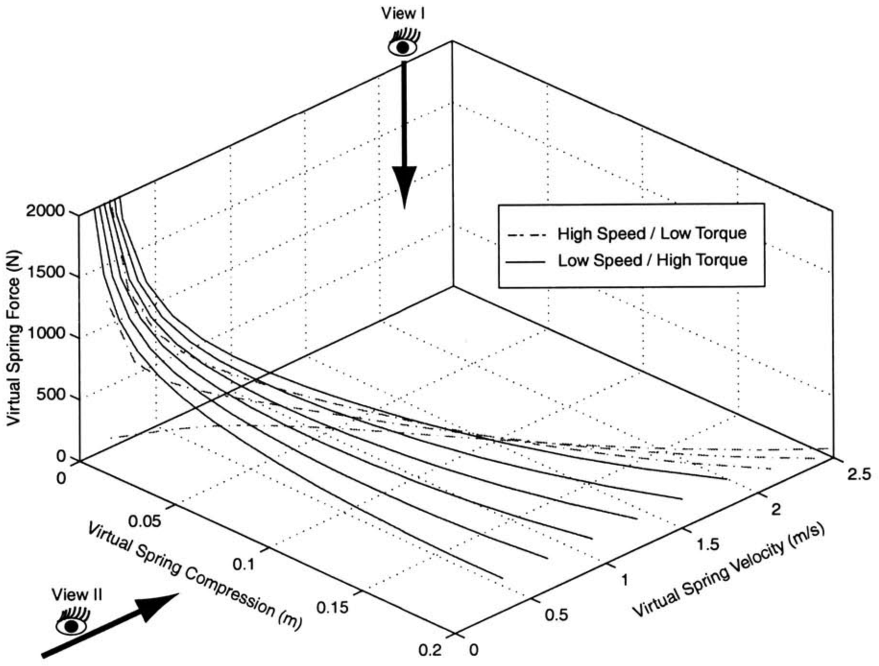

Figure 3-13: Relationship among force-velocity-compression for the virtual spring. Virtual spring force is shown in the vertical axis while spring compression and velocity are shown on the two horizontal axis. View I shows the relationship between Virtual Spring Force and Velocity and can be seen in figure 3-15. View II shows the relationship between Virtual Spring Force and Compression and can be seen in figure 3-14.

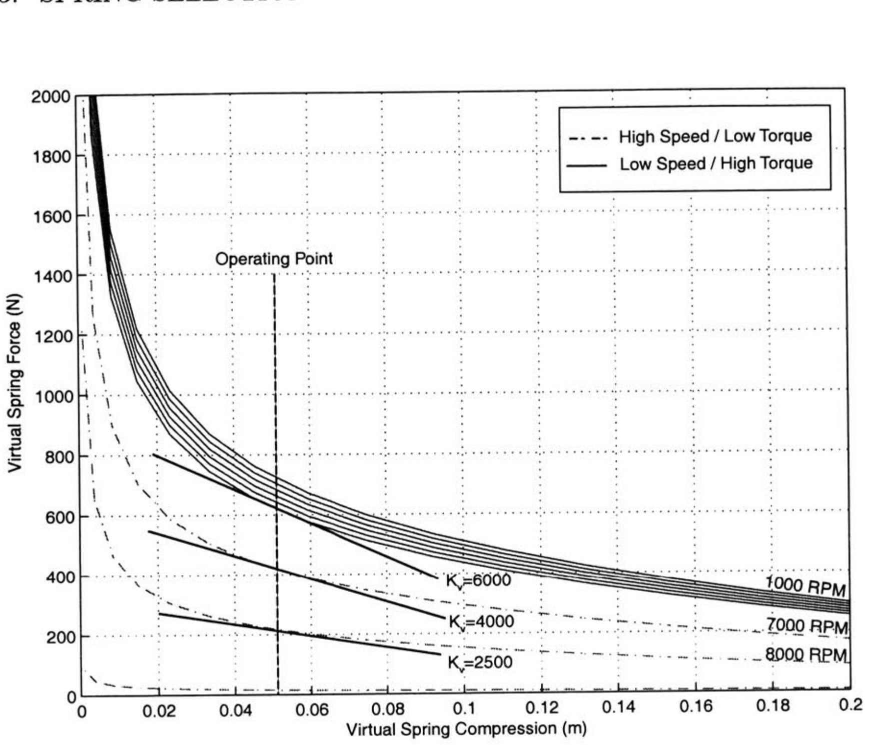

Figure 3-14: Two-dimensional projection of Force versus Compression of virtual spring as seen from View II in figure 3-13. Virtual spring stiffness is shown about the same operating point for motor speeds of 6000, 7000, and 8000 RPM.

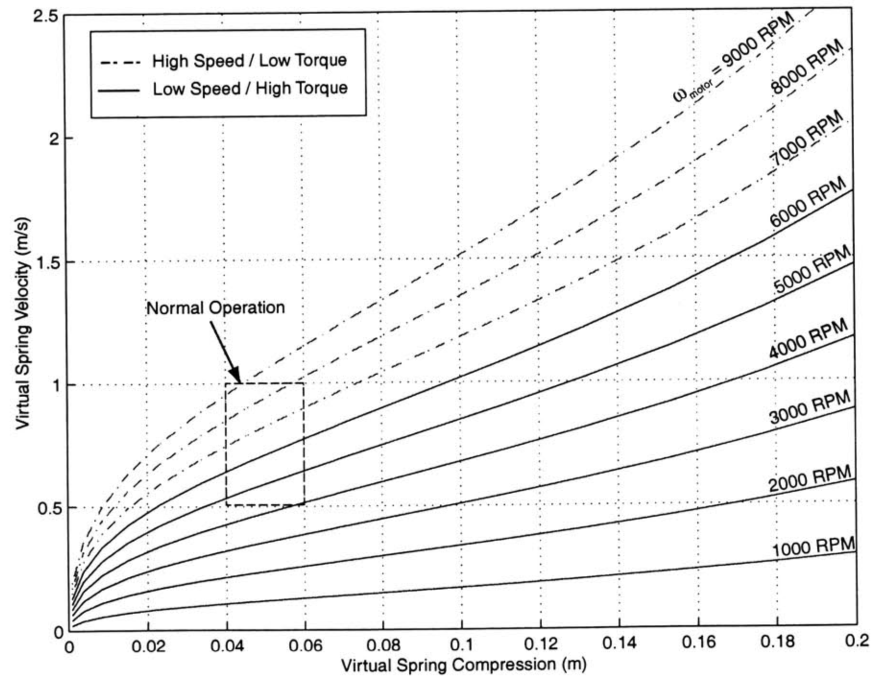

Figure 3-15: Two-dimensional projection of Force vs. Compression of virtual spring as seen from View I in figure 3-13. A dashed rectangle shows the normal operating region.

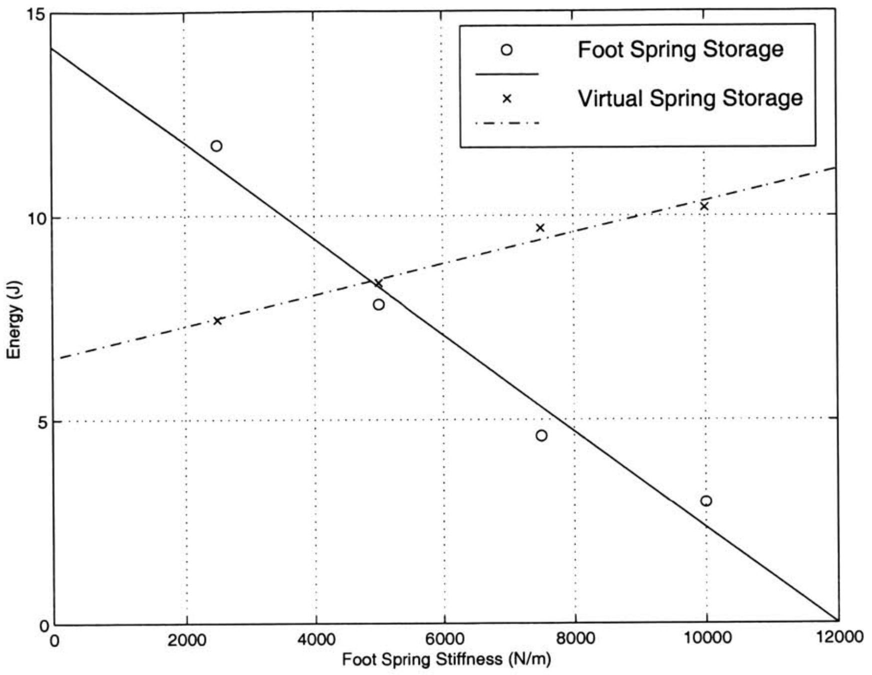

Figure 3-16: Energy storage in the foot spring and virtual spring while hopping at 0.25 meters with a bent knee.

Next, we need to choose the appropriate motor speed curve. To aid in this selection, see figure 3-15 which shows a two-dimensional projection of velocity and compression as seen from View I in figure 3-13. In figure 3-15 we have drawn a dashed box around a conservative estimate of the normal operating region of the spring. In fact, during touchdown, the virtual spring velocity will be near zero if the force in the foot spring does not exceed the maximum virtual spring force capable of being applied by the motor. To operate in this region, we can see that the motor speed must operate between 5000 and 8000 RPM. Normal motor operation is approximately centered on 6000 RPM.

Using this information and the compression operating point provided earlier, we can calculate the virtual spring stiffness from the slopes of the appropriate curves in figure 3-14. We see that the slope of RPM curves ranges from about 2500 N/m at 8000 RPM to 6000 N/m at 6000 RPM. Since normal operation is centered on 6000 RPM, from figure 3-15, we assume that 6000 N/m is a reasonable estimate of maximum virtual spring stiffness at touchdown.

Returning now to our original question: who wins the battle of energy storage? At foot spring stiffness of 2500 N/m, the virtual spring is significantly stiffer than our foot spring, suggesting that the foot spring will store more energy. Near 6000 N/m, the foot spring stiffness is well matched to the virtual spring stiffness and we expect to see equal energy storage. As foot spring stiffness continues to increase, it is no longer well matched with the stiffness of the virtual spring. This should result in increased storage in the virtual spring and decreased storage in the foot spring.

To test these predictions, our one legged hopping robot was controlled so as to maintain a hopping height of 0.25 meters and a virtual spring stiffness of 6000 N/m. The energy storage in the foot spring was recorded just before lift-off for 10 seconds and averaged. The energy stored in the virtual spring, which is equal to the absolute value of the energy dissipated by the knee joint, was also recorded just before lift-off. The data can be seen in figure 3-16.

Figure 3-16 clearly shows the effect of the virtual spring stiffness. At low foot spring stiffness

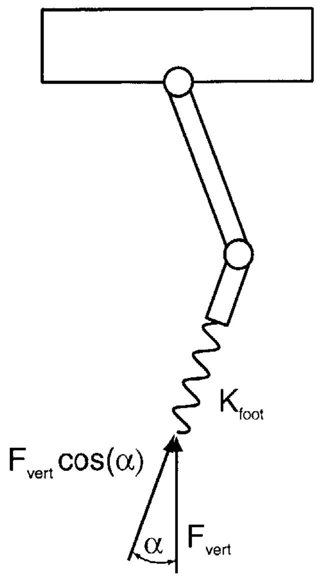

Figure 3-17: Force seen by the spring during touchdown with a bent knee.

the foot spring stores more energy than the virtual spring. At high foot spring stiffness, the virtual spring stores more energy that the foot spring. The two lines have equal energy storage at 5000 N/m, very close to our predicted value of 6000 N/m.

It is important to note from figure 3-14 that the maximum virtual spring stiffness can be considerably increased by simply decreasing the virtual spring compression (decreasing θ_{knee}). However, desired hopping height limits our ability to decrease the virtual spring compression below some threshold. Consider the case with zero spring compression (θ_{knee}=0.0). The virtual spring stiffness is infinite, but the robot is unable to inject energy to maintain hopping height.

To summarize the Virtual Spring Effect:

- The Virtual Spring Effect depends on the relative stiffness of the virtual spring and the foot spring.
- The virtual spring stiffness is limited by the torque-speed characteristics of our motors. Virtual spring stiffness increases significantly as knee flexion decreases, but reduced knee flexion limits maximum hopping height.
- It is wise to choose a foot spring much softer than the maximum stiffness of the virtual spring.

3.2.4 The Alpha Effect

Next, consider the Alpha Effect. When landing with a bent knee, the forces acting on the spring are no longer directly in line with the spring. This means that the spring will no longer see the entire force of impact. The glancing blow will not maximally compress the spring, resulting in less than maximum energy storage. This situation can be seen in figure 3-17.

From this figure, we can see that the force on the spring will be

$$F_{foot} = F_{vert}cos(\alpha) \tag{3.30}$$

Thus, deflection of the spring can be written as

$$x_{foot} = \frac{F_{vert}cos(\alpha)}{K_{foot}} \tag{3.31}$$

and the energy storage in the spring will be

$$E_{spring} = \frac{1}{2}K_{foot}\left(\frac{F_{vert}cos(\alpha)}{K_{foot}}\right)^2 \tag{3.32}$$

Assuming the impact forces at the foot are equal to those in the fully extended knee case, the energy will be decreased by a factor of ($cos^2\alpha$). Typically, α is about 20^o, which results in a 12 percent reduction in stored energy. This predicted loss is only dependent on α, thus should not affect our choice of spring selection. However, it is important to note the Alpha Effect worsens as knee flexion increases, suggesting that the most efficient landing configuration is with a minimal knee flexion.

For proof of the Alpha Effect, the one legged hopping robot was controlled so as to maintain a hopping height of 0.25 meters for ten seconds. The energy storage in the foot spring was recorded just before lift-off and averaged. The energy stored in the virtual spring, which is equal to the absolute value of the energy dissipated by the knee joint, was also recorded just before lift-off and averaged. The total energy storage was obtained by summing the virtual spring storage and the foot spring storage and was plotted in figure 3-18. To provide a base-line for comparison, we have also plotted the energy stored in the foot spring while dropping on a fully extended knee (least-squares fit in figure 3-7).

The Alpha Effect can be clearly be seen in figure 3-18 when comparing the bent knee and extended knee cases. The energy storage slightly decreases when going from the extended knee case to the bent case. Although the least-squares fit lines seem to diverge as stiffness increases, the discrete data points ('x' and 'o') clearly show a slight loss of energy. This downward shift of the data points is a direct result of the Alpha Effect.

To summarize the Alpha Effect:

- The Alpha Effect decreases the maximum energy storage of the foot spring and thus efficiency of the robot.
- The Alpha Effect is independent of foot spring stiffness. Rather, it is a function knee flexion: worsening as knee flexion increases.

3.2.5 Controllability Analysis and Simulation

When selecting spring stiffness for a running robot, one must carefully consider how the spring stiffness will affect the controllability of the robot. Three critical body motions must be successfully controlled for a robot to run and hop. These include hopping height, body pitch and running speed.

Hopping height is determined by how much energy can be injected into the robot during stance. As we have already discussed in section 3.2.2, a robot can inject no energy when the knee if fully extended. In this configuration we have no control over hopping height. As the knee flexion increases, we can inject more and more energy into the robot during stance, allowing for much better control of hopping height. Unfortunately, at large knee flexion angles, the maximum virtual spring stiffness is poor resulting in poor robot efficiency due to the Virtual Spring Effect. If a very soft foot spring is chosen, the Virtual Spring Effect can be reduced at large knee flexion angles. In this way, high efficiency and good hopping height control can be obtained.

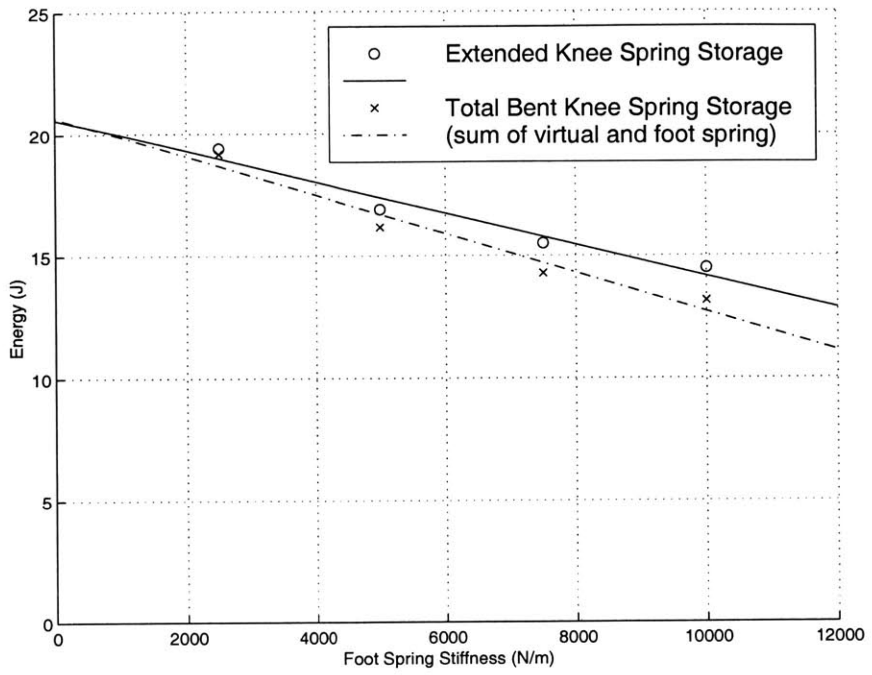

Figure 3-18: Energy storage for the fully extended knee and bent knee. For the bent knee case, energy storage is the sum of foot spring storage and virtual spring storage.

Body pitch is controlled during stance of the robot by applying torque between the hip and body of the robot. As stance time decreases, the robot has less time to correct the pitch of its body. To consider how foot spring stiffness affects stance time, we will refer to equation 3.17. It has been repeated here for convenience.

$$\dot{x} = C_1 a e^{at} sin(\omega_d t) + C_1 e^{at} cos(\omega_d t) + C_2 a e^{at} cos(\omega_d t) - C_2 e^{at} sin(\omega_d t)$$

Setting $\dot{x}$=0 and solving for t will yield the time to reach full spring compression. Doubling that time will yield the total stance time. Doing so, we find

$$t_{stance} = 2 \left(\frac{tan^{-1}\left(-\frac{C_1 + C_2 a}{C_1 a - C_2}\right)}{\omega_d} \right) \tag{3.33}$$

where

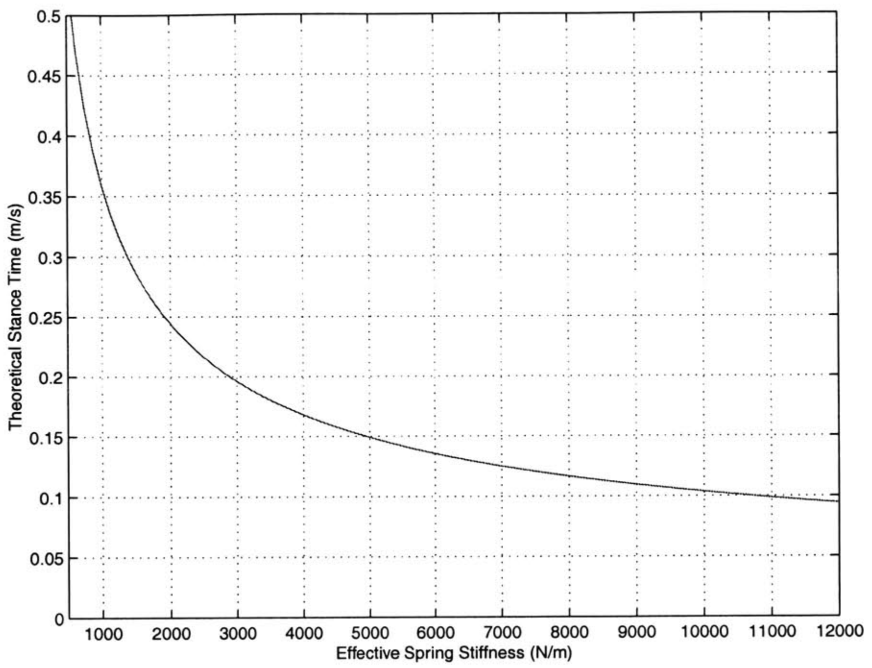

Figure 3-19: Theoretical stance time for one legged robot with a springy foot of stiffness from 500 N/m to 12000 N/m calculated using equation 3.33.

$$
\begin{aligned}
C_1 &= \frac{\sqrt{2gh} - \frac{Bg}{2K_{eff}}}{\omega_d} \\
C_2 &= \frac{-Mg}{K_{eff}} \\
\omega_d &= \sqrt{\frac{K_{eff}}{M} - \left(\frac{B}{2M}\right)^2} \\
a &= \frac{-B}{2M} \\
K_{eff} &= \frac{K_{virt}K_{foot}}{K_{virt} + K_{foot}}
\end{aligned}
$$

Using equation 3.33, the theoretical stance time was calculated for effective stiffness from 500 N/m to 12000 N/m. Figure 3-19 shows us that increasing effective spring stiffness reduces stance time. Such a relationship suggest that softer foot spring stiffness will result in better pitch control.

To verify the accuracy of equation 3.33 for our model, we measured the stance time for foot spring stiffnesses of 2500 N/m, 5000 N/m, 7500 N/m, and 10000N/m and a constant virtual spring stiffness of 6000 N/m in three different experiments: 1) Dropping the simulated robot on an extended knee ($K_{eff} = K_{foot}$) 2) Hopping on a bent knee (K_{eff} = 1750, 2727, 3333, and 3750) 3) Running on a bent knee (K_{eff} = 1750, 2727, 3333, and 3750). These measurements were compared with stance times calculated using equation 3.33. The results can be seen in figure 3-20. Good agreement is obtained between the calculated stance time ('o') and the measured stance time ('x') in all three experiments. Thus, a soft spring should be chosen for good pitch control.

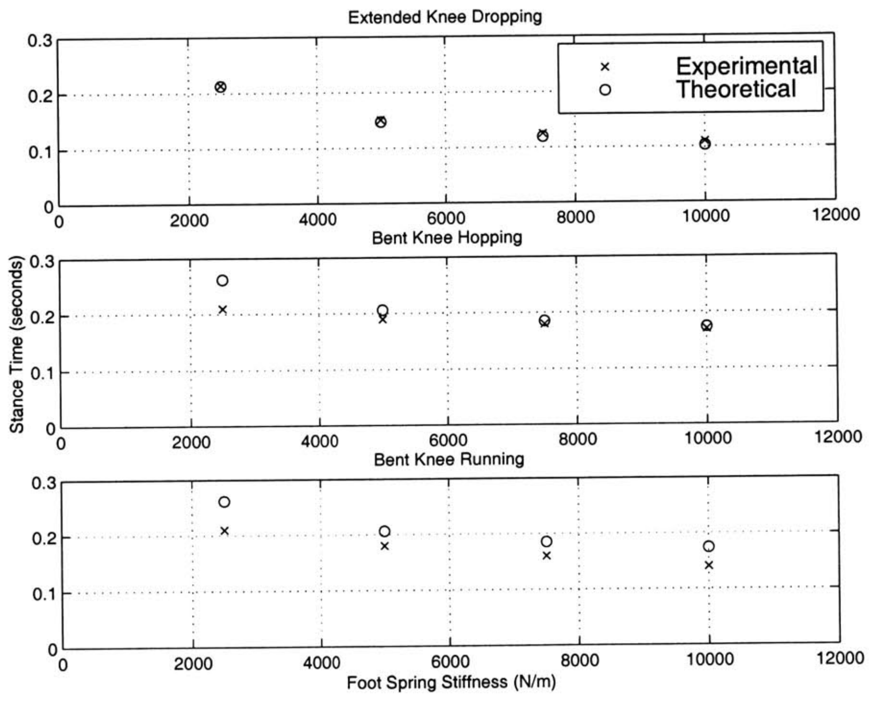

Figure 3-20: Theoretical and simulated stance time for one legged robot with a springy foot. The top most figures shows the simulated data 'x' recorded for a fall from 0.25 meters on a fully extend knee. The theoretical data 'o' was calculated using equation 3.33 with $K_{eff} = K_{foot}$. The middle figure shows the simulated data recorded for a vertical fall from 0.25 meters onto a slightly bent knee. The theoretical data 'o' was calculated using equation 3.33 with $K_{virt} = 6000N/m$. The bottom figure shows the simulated data recorded for a running simulation. The theoretical data 'o' was calculated using equation 3.33 with $K_{virt} = 6000N/m$.

Our final control parameter, running speed, is also a function of stance time, and thus spring stiffness.

$$V_{robot} = \frac{L_{stride}}{t_{stance}} \tag{3.34}$$

where L_{stride} is

$$L_{stride} = 2L_{leg}sin\left(\frac{\theta_{sweep}}{2}\right) \tag{3.35}$$

and L_{leg} can be calculated from figure 3-10

$$L_{leg} = 2Lcos\left(\frac{\theta_{knee}}{2}\right) \tag{3.36}$$

Assuming small knee angles, then velocity of the robot can be written as

$$V_{robot} = \frac{4Lsin\left(\frac{\theta_{sweep}}{2}\right)}{t_{stance}} \tag{3.37}$$

From equation 3.37 we can see that the running speed can be increased by either increasing sweep angle or decreasing stance time. Sweep angle is limited by physical limit stops in the robot. Stance time, on the other hand, can be reduced by simply increasing the effective spring stiffness. Solving Equation 3.37 for effective stiffnesses ranging from 500 N/m to 12000 N/m we see that the theoretical velocity increases with increasing effective stiffness (See figure 3-21).

To test this equation, Raibert's three part running algorithm [Raibert. (1986)] was implemented on the one legged hopping robot. For each foot spring stiffness, 2500 N/m, 5000 N/m, 7500 N/m, and 10000N/m and a constant virtual spring stiffness of 6000 N/m, the running speed was increased until the robot fell down. The simulation data can also be seen in figure 3-21 as discrete 'o's. The simulation data shows a general trend of increasing velocity with increasing stiffness. The discrepancy between the theoretical maximum running speed and the simulated running speed can be blamed on the imperfect control algorithm. Still, it can be seen that increasing stiffness yields wider velocity range for a given controller.

To summarize the controllability analysis:

- Controllability of hopping height improves as the angle of knee flexion increases. The robot can operate at large knee flexion angles only if the foot spring is much softer than the virtual spring.
- Controllability of body pitch improves as the stance time increases. Stance time can be increased by reducing the stiffness of the foot spring.
- Maximum running speed increases as stance time decreases. Improve controllability of running speed calls for a stiff spring.

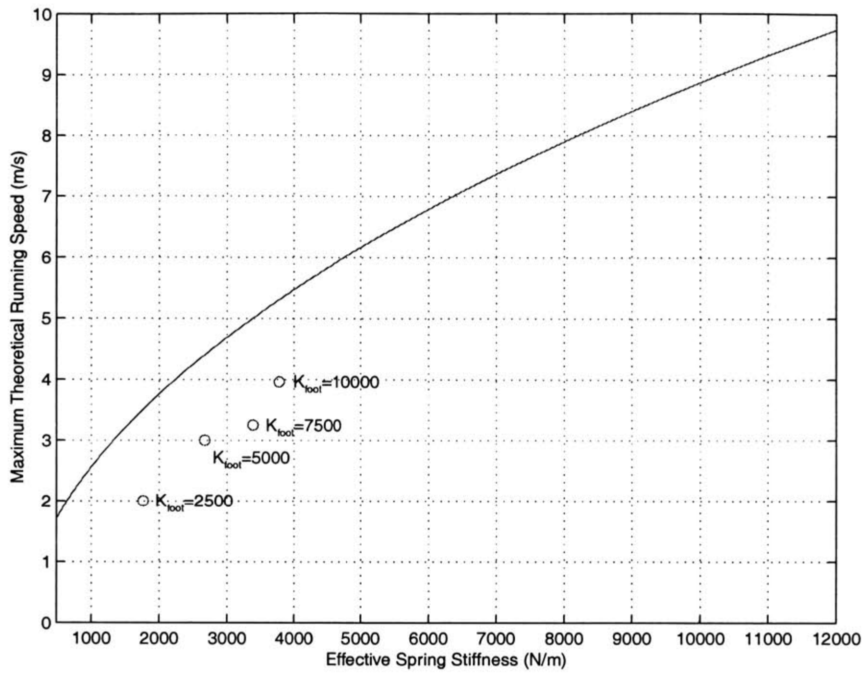

Figure 3-21: Theoretical maximum running speed for one legged at varying effective spring stiffnesses calculated using equation 3.37.

3.3 Recommendations

3.3.1 Selection and Use of Springs

Finally, from the analysis and simulation conducted we know that:

- Soft foot springs provide robustness against the Virtual Spring Effect, resulting in a more efficient robot.
- Soft foot springs allow us to operate efficienctly at large knee flexion angles, thus allowing for a wider range of hopping heights.
- The Alpha Effect degrades energy storage in the foot spring at large knee flexion angles.
- Soft foot springs provide longer stance time, thus better pitch controllability.
- Stiff foot springs provide shorter stance time, thus higher maximum speeds.

Said another way, soft springs allow for efficient running and good control of pitch and hopping height at the cost of speed. Based on these conclusions, we have developed this simple rule of thumb for choosing spring stiffnesses.

Choose the softest foot spring that is capable of reaching the maximum desired running speed using figure 3-21. Keep in mind that greedily choosing maximum running speed will severely degrade efficiency, and controllability of height and pitch.

Careful choice of spring stiffness does not guarantee an efficient robot. The spring must also be operated in the appropriate region for maximum benefit. A basic guideline for using the spring follows.

Choose the maximum desired hopping height of the robot so that it is just enough for the foot to clear the ground. Doing so will minimize knee flexion during touchdown resulting in reduction of the Alpha Effect and the Virtual Spring Effect.

Chapter 4

Natural Dynamics

Natural dynamics were briefly reviewed in Chapter 2. Recall that natural dynamics refers to the natural, unactuated motions of robotic and biological systems. It was shown that natural dynamics play an important role in both robotic and biological creatures. For this reason, the experimental robot will contain natural dynamic elements.

In this chapter, we expand on these ideas, showing why natural dynamics are important and how to design robots capable of natural dynamic behaviors. The effect of reflected motor inertia on natural dynamic behavior is discussed. Finally, an actuation method to mask the effects of reflected inertia is set forth.

4.1 Design for Natural Dynamics

Natural dynamics are valuable to legged robots for two reasons. First, since natural dynamic motions occur naturally, no computation or control is required to determine or follow a natural looking trajectory. Second, since natural dynamics are driven by gravity rather than by the robot's power source, natural dynamic motion can be used to achieve high efficiency.

Two natural dynamic elements used by humans will be incorporated into the experimental robot. The first natural dynamic element is the knee-stop. Humans use the tendons to lock the knee joint, allowing us to stand upright for long periods of time while expending very little energy. Without tendons to provide a hard limit-stop, the seemingly simple task of standing would become extremely tiring. Consider figure 4-1. On the left, without a knee-stop, an unstable buckling configuration can be seen. In order to stabilize this configuration the knee torque must switch between positive and negative at a very high rate, causing "chatter". In the center, a stable configuration can be seen with a slightly bent knee. This configuration requires continuous knee torque and is thus always consuming power. Finally, on the right, we see a configuration which is both stable and efficient because of the knee-stop. To maintain an upright position, a small torque is applied at the knee, which locks it in place against the knee-stop. This is what we do when we stand in place for long periods of time. This element will be realized in the experimental robot with a hard limit-stop at the knee's fully extended position.

Passive swing leg behavior is the second important natural dynamic element which will be implemented on the experimental robot. To avoid stubbing the toe on the ground during swing phase, the leg must be shortened. It is possible to actively shorten the leg, however, passive swing leg behavior uses natural dynamic elements to shorten the leg during swing. Consider figure 4-2. In the left figure the hip joint receives a torque impulse to initiate the swing phase. In the two center figures, zero torque is applied at the hip and knee. During these two phases, the thigh continues to swing forward while the shin is drawn toward the body, obeying the conservation of angular momentum. Finally, the last frame of figure 4-2 shows a hip torque impulse in the opposite direction to bring the thigh to a sudden stop. Since there is zero torque at the knee joint, the shin naturally swings forward and hits the knee-stop.

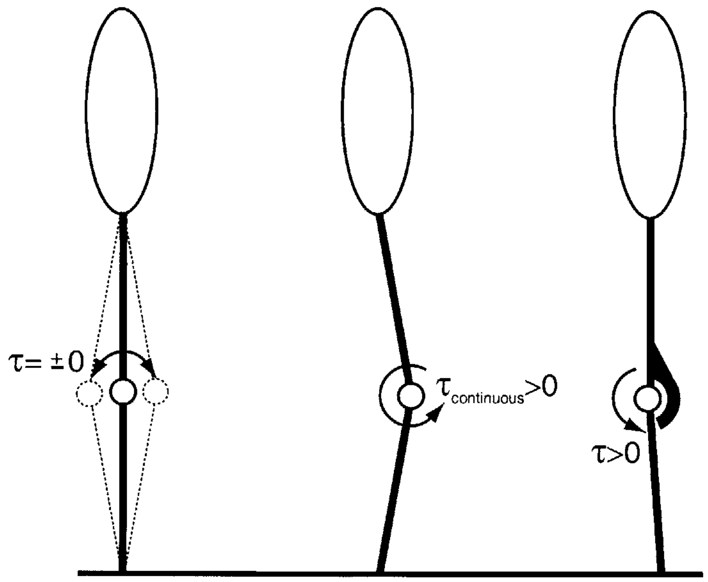

Figure 4-1: Standing upright with and without a knee cap. Left shows an unstable buckling configuration. Center, shows a stable but inefficient configuration. Right shows a stable and efficient configuration

To achieve passive swing leg behavior, the robot must possess an articulated knee joint as shown in figure 4-2. This natural dynamic element is not possible with telescoping legs or legs with inverted knee joints, as in flamingos. Note also, that passive swing leg behavior would not be possible without the knee-stop to bring the shin to a hault without actuation.

4.2 Impediments to Natural Dynamics

It is a relatively simple task to design natural dynamics into a robot by incorporating an articulated knee with a knee-stop. However, allowing natural dynamics to dominate the behavior of a robot is not a simple task. Reflected motor inertia, gear train backlash, and joint friction all act as impediments to the natural dynamics of the robot. Of these three effects, reflected inertia tends to dominate. To further investigate how reflected inertia hinders the natural dynamics of robots, a simple pendulum driven by a rotary motor will be used to represent a typical robot joint and link. This model can be seen in figure 4-3.

The natural dynamic behavior of the pendulum shown in figure 4-3 is simple harmonic motion, similar to the tick-tock of a clock. To achieve this natural motion efficiently and without computation or control, one might suggest killing the power to the motor, thus allowing the pendulum to swing to and fro unactuated. There is a difficulty with this simple scheme: the inertia and friction of the motor will disrupt the natural frequency (i.e. natural dynamics) of the pendulum.

$$J_{effective} = J_{pendulum} + J_{motor} \tag{4.1}$$

$$\omega_n = \sqrt{\frac{J_{effective}}{mgL}} \tag{4.2}$$

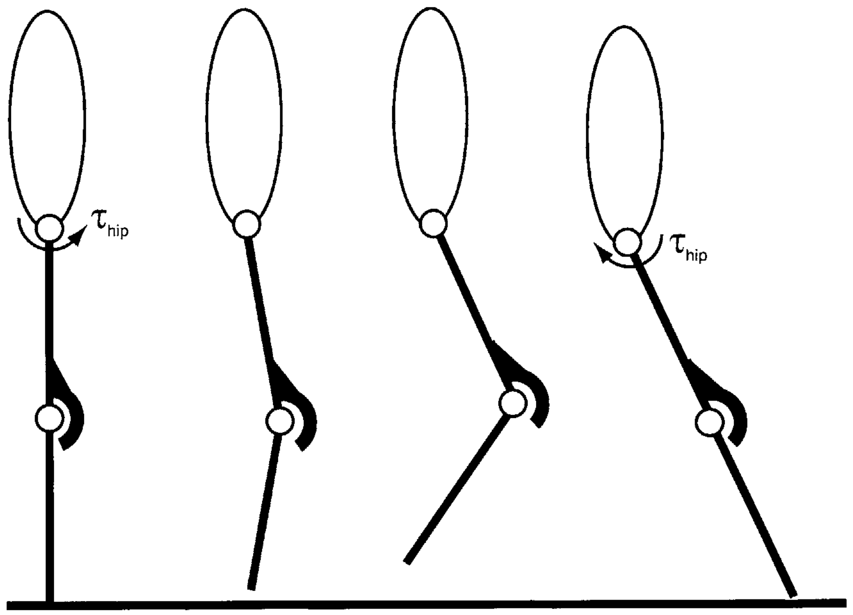

Figure 4-2: Swing leg behavior shown with an articulated knee. An initial torque impulse at the hip swings the thigh forward. The knee joint naturally breaks and the shin swings toward the body to conserve angular momentum. Just before touchdown, a torque impulse is given at the hip joint to bring the thigh to a sudden stop. The shin continues to swing forward eventually being stopped by the knee limit stop.

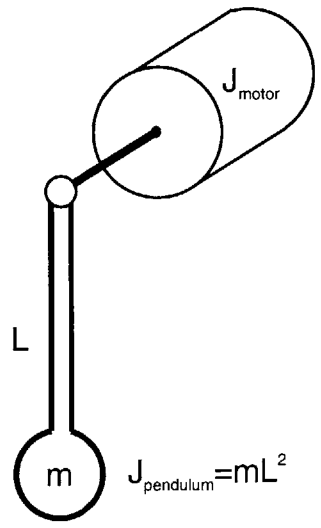

Figure 4-3: Pendulum driven by a rotary motor. This model is used to represent a typical robot joint and link.

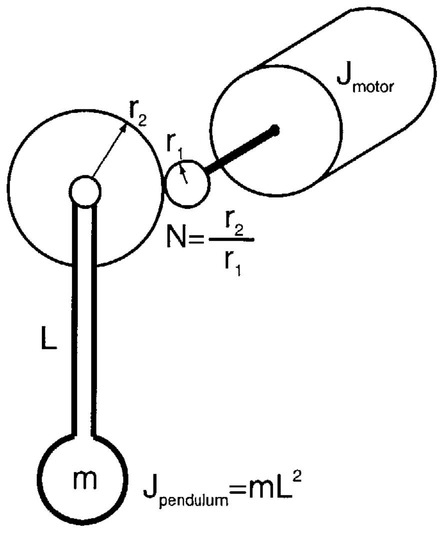

Figure 4-4: Pendulum driven by a rotary motor through a gear reduction. The gear reduction is necessary on most robot joints to increase the torque and reduce the speed of the motor.

For example, if J_{motor} is equal to $J_{pendulum}$ then the natural frequency of the pendulum would be altered by a factor of $\sqrt{2}$.

In fact, the situation is typically much worse. Most motors operate most efficiently at high speeds and low torques. Conversely, biological systems generally operate at low speeds and high torques. To correct this disparity, legged robots rely on large gear reductions, typically on the order of hundreds or even thousands, at their joints to produce high torque and low speed. The gear reduction has a severe effect on the reflected inertia of motor.

If the pendulum is attached to the motor through a gear reduction, as in figure 4-4, the effective inertia of the pendulum arm will become

$$J_{effective} = J_{pendulum} + N^2 J_{motor} \tag{4.3}$$

Where N is the ratio of the gear reduction and $N^2 J_{motor}$ is the reflected inertia of the motor.

From equation 4.3, it can be seen that for large N, the reflected inertia of the motor may dominate the inertia of the pendulum. In this way, the natural frequency of the pendulum (equation 4.2) will be significantly altered by the motor inertia. Clearly, natural dynamic behaviors, such as swing leg, will be impossible to implement with motors attached to the joints through large gear reductions.

4.3 Masking the Effects of Reflected Inertia

As we have seen, the reflected inertia of the motor can significantly alter the natural dynamics of a robot link. Fortunately, a clever actuating scheme, known as Series Elastic Actuation [Pratt & Williamson (1995), Williamson (1995), Robinson et al. (1999)], can be employed to mask the effects of reflected motor inertia. Such a scheme allows the motor to present a very low impedance (i.e. low apparent inertia) to the joint. With a low impedance at the joint, the robot link will dominate the dynamics instead of the motor's reflected inertia. By actively controlling the motor to improve output impedance we loose the efficiency benefit of natural dynamics, but are still able to achieve natural looking motions with little computation.

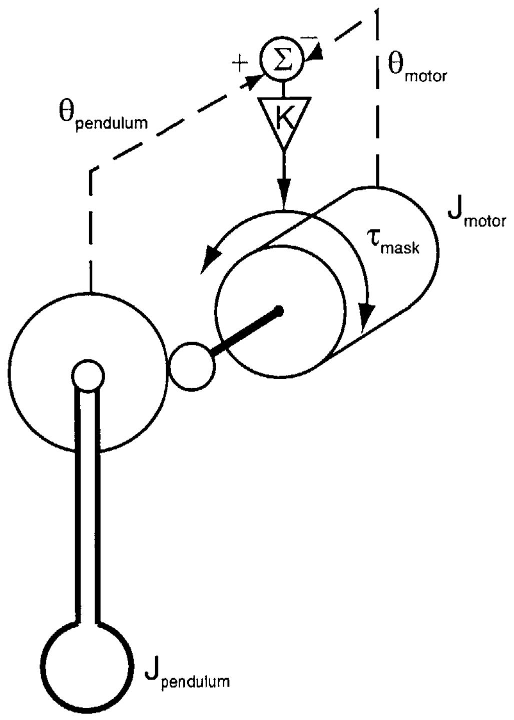

Figure 4-5: Position control of motor inertia in an attempt to compensate for the effects of reflected inertia. The angular position of the motor is sensed and compared to the angular position of the pendulum. Any discrepancy between the two measurements will result in a torque on the motor which accelerates the motor inertia toward the position of the pendulum.

Before investigating Series Elasticity, first let us consider the *inability* of position control to present a low impedance to the environment. To attempt to mask reflected inertia of the motor with position control, one can apply a torque to the motor inertia to accelerate it in the direction of the natural dynamic motion. Figure 4-5 shows schematic representation of the sensing and control to achieve this.

In its simplest form, the control law would look like

$$\tau_{mask} = K(\theta_{pendulum} - \theta_{motor}) \tag{4.4}$$

With this control law, a torque is applied to the motor inertia if θ_{motor} is *not equal* to $\theta_{pendulum}$. In this way, a positive angular displacement of the pendulum results in a positive torque on the motor inertia. Conversely, a negative angular displacement of the pendulum results in a negative torque on the motor inertia. This method would work well if the pendulum was decoupled from the motor inertia; unfortunately, because $\theta_{pendulum}$ is mechanically coupled to θ_{motor}, τ_{mask} will always be zero. Thus, position control does nothing to lower the impedance of the joint.

Similar to position control, Series Elastic Actuation applies a torque to the motor inertia to accelerate it in the direction of the natural dynamic motion. The key difference between position control and Series Elastic Actuation is a spring placed between the pendulum and the gear reduction as shown in figure 4-6. The spring effectively decouples the motion of the pendulum from the motor inertia.

The deflection of the spring is measured and used in a control law according to

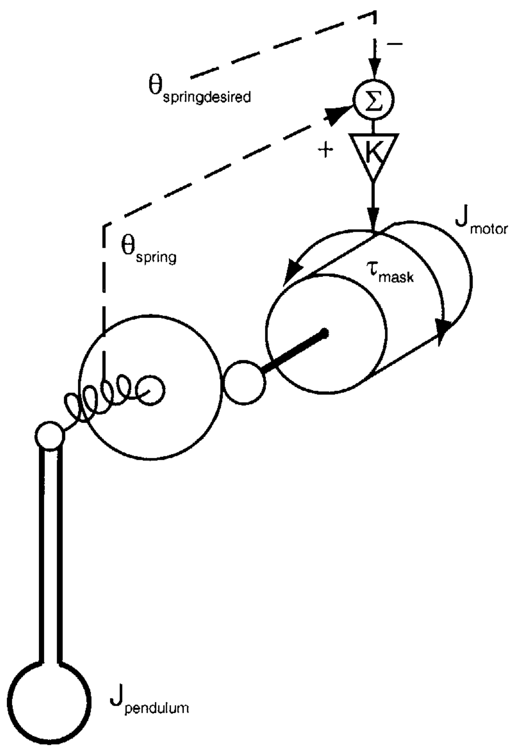

Figure 4-6: A torsional spring has been placed in series between the load (i.e. the pendulum) and gear reduction. This is known as Series Elasticity and decouples pendulum from motor inertia, allowing for compensation of reflected inertia. Additionally, Series Elasticity can provide accurate torque control of joints and high shock tolerance.

$$\tau_{mask} = K(\theta_{spring} - \theta_{springdesired}) \tag{4.5}$$

If $\theta_{springdesired} = 0$, τ_{mask} will begin to accelerate the motor inertia in the direction of θ_{spring} as soon as a deflection in the spring is detected. In this way, the pendulum never feels the effect of the motor inertia because the motor generates its own compensating torque.

Series Elastic Actuation can also provide a joint with accurate torque control. For example, if $\theta_{springdesired}$ is set to a non-zero value, say 1.57 radians, then τ_{mask} will provide a torque to insure that the compression of the spring is 1.57 radians at all times. If the torsional spring constant and angular defection are known, the torque at the joint can be precisely known Using Hookes Law ($\tau = k\theta$). Inversely, Hookes Law can be used to determine a desired spring compression, $\theta_{springdesired}$, for a given desired torque. As you will see in Chapter 6, accurate force control will simplify the overall control of the robot.

Series Elastic Actuation also provides the robot's joints with high shock tolerance. Imagine hitting the pendulum in figure 4-5 with a hammer. The high impulse force of the hammer is likely to damage the gear train. On the other hand, if the pendulum in figure 4-6 was struck with a hammer, the spring would act like a low pass filter, protecting the gear reduction from the impulse load.

4.4 Summary of Natural Dynamics

In summary, natural dynamic behavior is both energetically and computationally efficient for legged robots. Natural dynamic behaviors, such a swing leg and knee-stop, can be easily incorporated into the morphology of legged robots. However, without the ability to mask reflected inertia of the motor, the natural dynamics will be significantly altered by the inertia of the motor.

Series Elastic Actuation provides the abililty mask the inertia of the motor, accurately control joint torques, and reject shock loads. These benefits come at the expense of energy consumption. That is to say, active control to compensate for motor inertia eliminates the efficiency benefits of natural dynamics. Still, we can profit from the computational ease and natural looking motion realized through natural dynamics.

Chapter 5

Experimental Robot

In chapter 2 the importance of springs and passive dynamics was discussed. Chapter 3 and chapter 4 take an in depth look at the task of designing for springs and natural dynamics. In this chapter, we will document the mechanical hardware used to realize these two design requirements. Other key design specifications, such as the planarizing boom and the size and weight of the robot will be discussed. Additionally, the electronic system required to sense and control the robot is set forth. A schematic representation of the robot can be seen in figure 5-1 while of a photograph of the hardware can be seen in figure 5-2.

5.1 Planarizing Boom

Because of the complexity associated with a fully three dimensional robot, in both design and control, it was decided early in the project to build a simplified robot to act as a test bed for design considerations and control algorithms.

To simplify the robot, it is restricted to motion in the sagittal plane by a planarizing boom. That is to say, the robot is restricted to horizontal and vertical translation and rotation about its center of gravity in a vertical plane passing through its spine. The degrees of freedom in the constrained motion can be seen in figure 5-3. In this way, the robot can not tip to its left or its right but retains the ability to pitch its body. As an analogy, imagine walking or running down an extremely narrow corridor where your shoulders are pressed firmly against the walls to your left and right sides. You can fall forward and backwards but not left or right.

Planarizing the robot allows for the removal of several degrees of freedom in the body, greatly simplifying mechanical design while also easing control issues. Since a boom provides lateral stability, only two legs, not four, are required. Two legs are simpler to accommodate in design of the body and much easier to keep track of during control. Furthermore, the two legs that are required need only one degree of freedom at the hip to sweep the leg forward and backward. The degree of freedom which allows for abduction of the leg from side to side, normally required for lateral balance, is not necessary.

This planar testbed robot is used to investigate key design issues, such as spring selection, and also as a tool to develop simple two dimensional running algorithms which can later be applied to a fully three dimensional quadruped. The boom with the robot attached can be seen in figure 5-4

5.2 Size and Weight

A large dog, weighing 75 pounds and standing 30 inches high, was chosen to provide rough guidelines for size and weight. This size is convenient for two reasons. First, a larger robot would be difficult and dangerous to handle, while a smaller robot would require miniature parts and become tedious and expensive. Second, most of the data available, such as power requirements and joint ranges, is

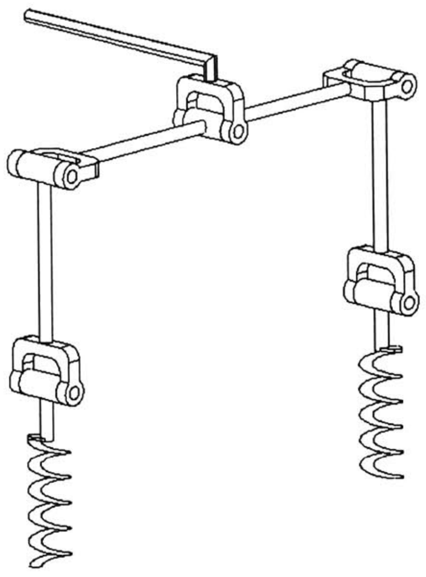

Figure 5-1: Joint schematic of a planar quadruped with two degrees of freedom in each leg, a passive spring at the foot, and a boom to allow body pitch but prevent tipping from side to side (frontal plane).

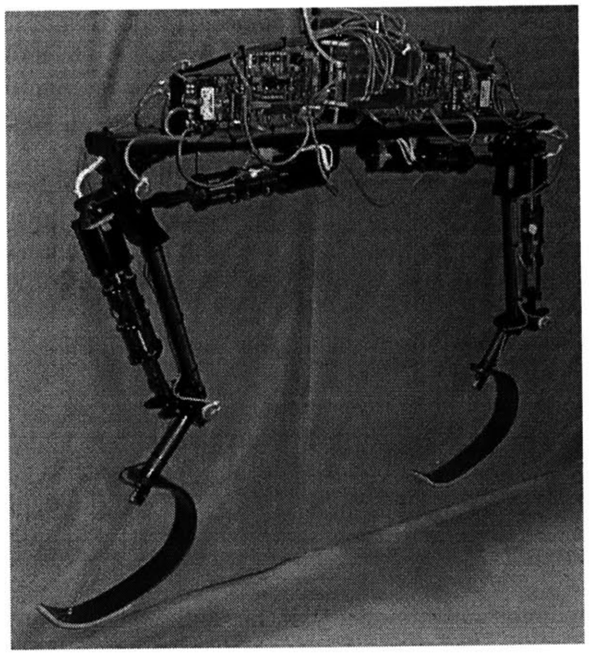

Figure 5-2: Photograph of the experimental robot. Rear leg on the left and front leg on the right.

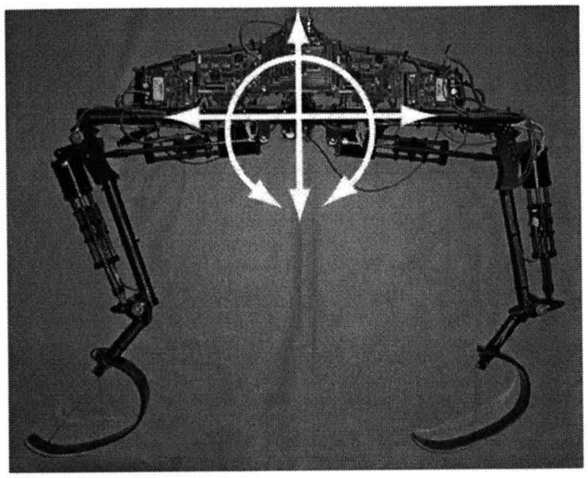

Figure 5-3: Photograph of planar robot with arrows showing degrees of freedom achievable when attached to a boom.

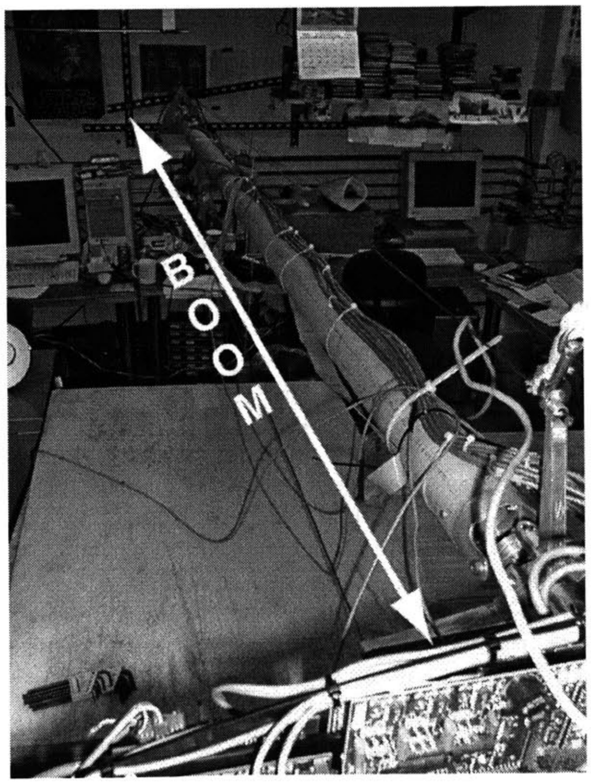

Figure 5-4: Photograph of planarizing boom attached to the two legged experimental robot.

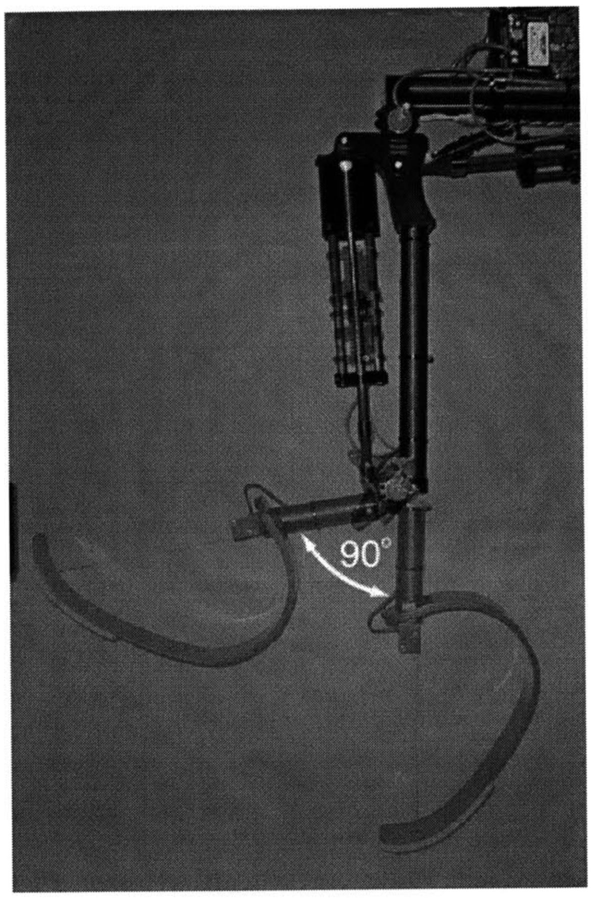

Figure 5-5: Photograph showing the range of motion for the rear knee joint.

from robots and animals of similar size. Since our robot will be planar (i.e. only one half of a dog) the weight target can be adjusted to about 37 lbs.

5.3 Body, Legs, and Joints

Traditionally, successful running robots have relied on prismatic, or pogo-stick style, knee joints [Raibert. (1986)]. This style of knee joint was used on the Raibert quadruped shown in figure 5-8. Prismatic knees are desirable for their simple mechanical design and especially, for their ease of control. Unfortunately, it is impossible to implement passive dynamic behaviors, such as knee cap and swing leg.behaviors shown in figures 4-1 and 4-2 of chapter 4.

For this reason, an articulated knee will be used in both the front and rear legs of the robot. The articulated knee will allow us to take advantage of passive dynamic swing leg behavior and the knee cap. A one degree of freedom revolute joint is placed at each hip. Joint ranges for the hip and knee were estimated based on joint data from computer simulations, actual robot data, and biological studies [Herr (1998), Raibert. (1986), Pandy et al. (1988), Farley et al. (1993)]. Photographs of the hind leg of the robot, showing the range of knee and hip motion can be seen in figures 5-5 and 5-6, respectively.

With the desired joint morphology set forth, the next consideration is the design of the robot's legs and body. Overall, the aim is to keep the robot as light as possible. Carbon fiber reinforced

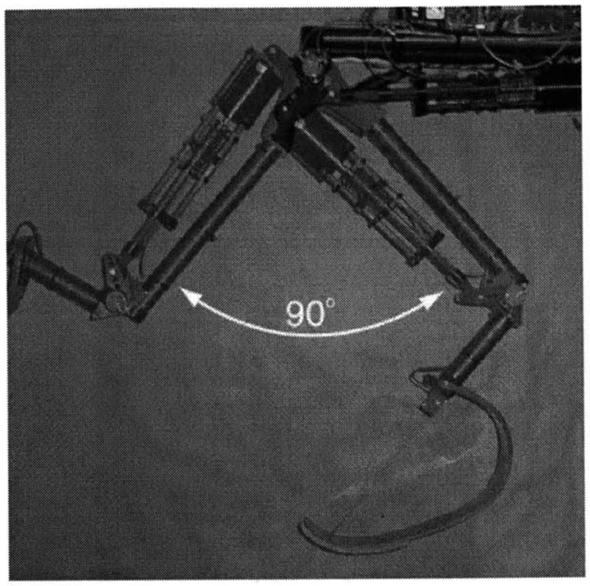

Figure 5-6: Photograph of showing the range of motion for the rear hip joint.

Table 5.1: Weight distribution of planar quadruped.

Limb	Length	Mass	Inertia
Body	36.0″	$18.8lbs$	$180.0lbs - in^2$
	1.0 m	8.5 kg	$0.057kg - m^2$
Upper	14.8″	$5.1lbs$	$157.3lbs - in^2$
Limb	0.41 m	2.3 kg	$0.046kg - m^2$
Lower	14.8″	$1.6lbs$	$64.4lbs - in^2$
Limb	0.41 m	0.7 kg	$0.019kg - m^2$

polymer tubes were used for the structural components of the body and legs. To keep the power requirements of joints as small as possible, we aimed to keep the inertia of the legs as low as possible. This was achieved by placing the knee motors as close to the hips as possible. The knee motor location relative to the hip can be seen in figure 5-7. For the body, the target inertia was approximately 10 times the combined inertia of the body. If the inertia of the body is much greater than the inertia of the legs, then the swinging legs will have little effect on pitch attitude of the body.

The mass, length, and inertial properties of the experimental robot's links have been summarized in table 5.1. At just over 35 pounds, the mass of the experimental robot came in very close to the target. Unfortunately, the inertia of the body is less than the combined inertia of the legs. This may cause control problems for the experimental robot, but the inertia of the body can always be increased at the cost of weight.

5.4 Spring

A springy foot for smooth energy transfer will be located in the foot and shin area of the robot. This location was chosen for design ease. To maintain an acceptable efficiency and good controllability

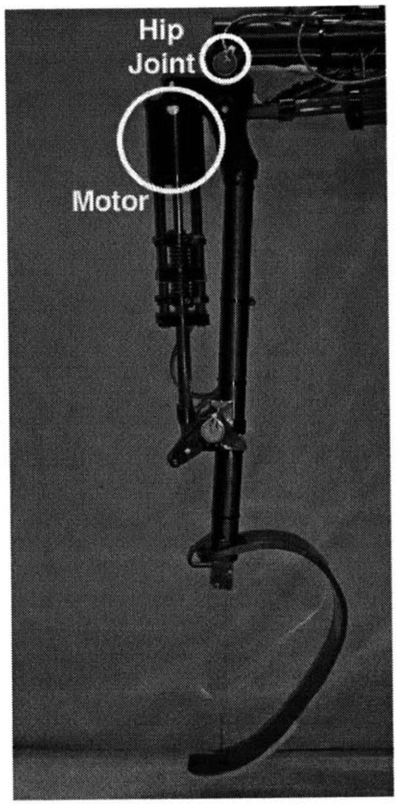

Figure 5-7: Photograph rear leg showing the knee motor location with respect to the hip joint.

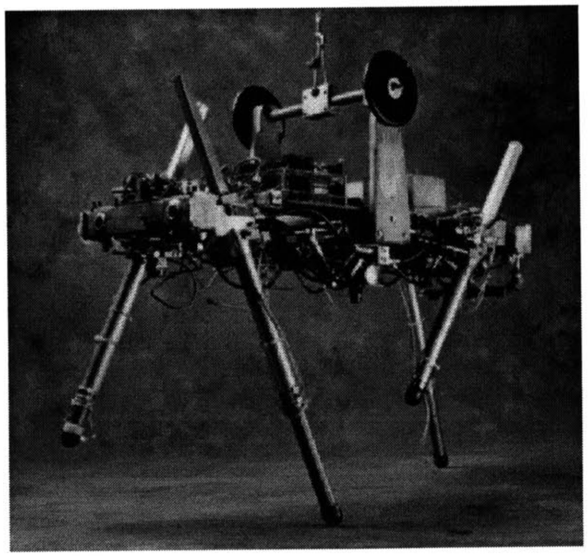

Figure 5-8: Photograph of the Raibert quadruped showing prismatic knee joints.

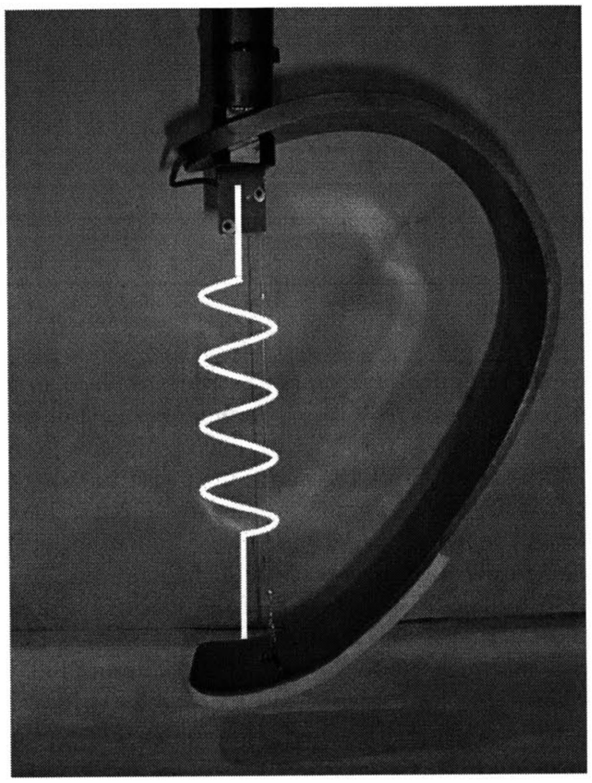

Figure 5-9: Photograph of the Cheetah Foot used for smooth energy storage and transfer during running. A string potentiometer is attached to measure compression of the spring.

while allowing the robot to reach running speeds, a spring constant of approximately 6000 N/m was chosen. The stiffness of the spring was chosen using the rules of thumb developed in chapter 3.

With spring stiffness specified, the next step was selection of spring type. Die springs were considered but quickly eliminated as an option because of weight and efficiency limitations of steel die springs. After carefully investigating various other springs, we decided upon a carbon fiber leaf spring, called a Cheetah Foot. The Cheetah Foot is designed and manufactured by Flex Foot, a company that specializes in prosthetic devices. The carbon fiber spring is a custom design based on similar springs used by amputees participating in track and field competitions. The spring itself is highly efficient and very light weight, thus a good choice where energy and weight are of the utmost importance. A photograph the Cheetah Foot can be seen in figure 5-9

5.5 Actuators

Next, power requirements and stroke lengths for the actuators were estimated. To do so, torque, swing range, and power of each actuated joint had to be determined. Torque and swing angle estimates for running quadrupeds were gathered from computer simulations, actual robot data, and biological studies [Herr (1998), Raibert. (1986), Pandy et al. (1988), Farley et al. (1993)]. The results for the hip ranged from 40 N-m to 120 N-m with swing angles of up to 90°. No information was available on knee torques or ranges. It will be assumed that similar torque and power is required from the knee joint.

We found that the estimated stride frequency for all running gaits had an upper limit of approximately 2.5 Hz [Heglund & Taylor (1988), Herr (1998)]. Unfortunately, little information is available

Table 5.2: Detailed joint information

Joint	Type	Range	Power	Torque	Frequency
Hip	revolute	$\pm 45^o$	280 Watts	$40N - m$	2.5 Hertz
Knee	revolute	$0^o - 90^o$	280 Watts	$40N - m$	2.5 Hertz
Pitch	revolute	360^o	—	passive	—
Foot	passive spring	$0'' - 3.0''$ $6000\frac{N}{m}$	—	passive	—

on actual power requirements for individual joints. In order, to place an upper bound on the power requirements, a simple calculation was performed. Assuming that foot placement is used to control forward running speed, rather than hip torque, then maximum joint power consumption will occur during the swing phase. With this in mind, the maximum power to swing the experimental robot leg $\pm 45^o$ at 3 Hz is 280 Watts, while the maximum torque was 40 N-m. These number are used at the targets for power, torque, and range of the hip and knee joints of experimental robot. A summary of joint ranges, powers, and torques can be seen in table 5.2

With the information in table 5.2 the actuator of the robot can be selected. The Leg Lab has focused significant effort on actuator development in recent years. The primary focus has been on force controllable Series Elastic Actuators [Pratt & Williamson (1995), Robinson et al. (1999)]. Series Elastic Actuators have a series elastic element between the gear train and the load which provides the gear train with high shock tolerance. More importantly, it allows for accurate control of forces through measurement of the spring deflection.

Recall from chapter 4 that force control is desirable for two reasons. First, since a robot consists of inertial elements, force control allows the natural dynamics of the robot to contribute to the robots movements. In position control, the dynamics of the robot are rigidly controlled to follow specified trajectories, thus the controller fights natural motions. Second, force control allows us to provide a compliant interface between the robot and the world. This compliance filters the external disturbances on the robot and thus allows it to adjust to stiff, high frequency, disturbances. Such disturbances occur every stride when the robot strikes the ground. On the other hand, position control presents a high impedance interface to the world. This stiff interface is not robust to external disturbances.

Fortunately, a Series Elastic Actuator [Robinson et al. (1999)] which meets the requirements of table 5.2 has already been developed. The actuator consists of a permanent magnet DC motor directly driving a ball screw. The load is attached to the output of the ball screw through a spring. The linear force of the actuator is converted into torque by a moment arm attached to the joint. The actuator is capable producing 1321 Newtons at 0.27 m/s, or 350 watts (peak). With this actuator, we can meet the maximum power requirements of the robot. The actuator can be seen in figure 5-10 while its physical properties can be seen in table 5.3

5.6 Electronic Components

The key electronic components necessary to sense and control the robot are listed below.

- Rotary potentiometers to sense joint positions and body pitch attitude.
- Linear potentiometers to sense foot contact.
- Signal conditioning board to filter potentiometer signals and produce clean position and velocity signals.

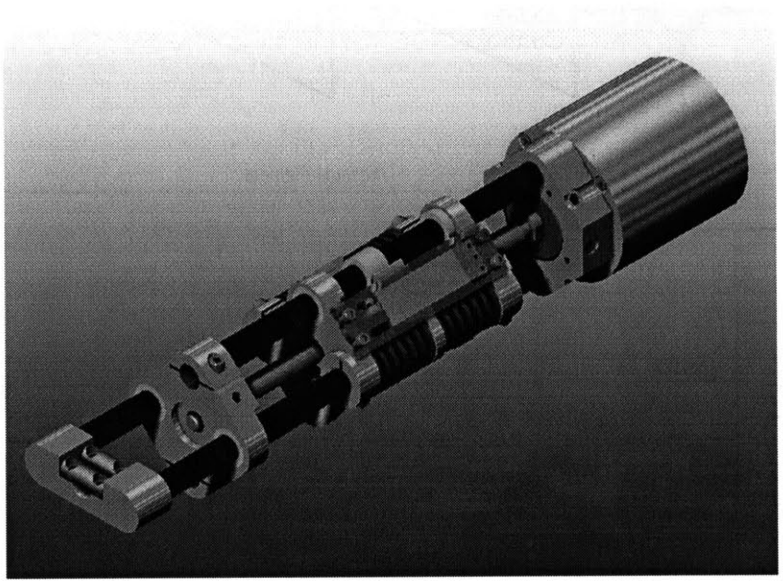
Figure 5-10: Computer rendering of series elastic actuator to be used in the robot.

Table 5.3: Physical Properties of Series Elastic Actuator.

Parameter	Value	Units
Max Force	1350	N
Cont. Force	750	N
Min Force	2.5	N
Max Speed	.25	m/s
Mass	1.13	kg
Dynamic Mass	128	kg
Spring Constant	315	kN/m
ω_n	7.9	Hz
ω_c	35	Hz
ζ_n	0.2	no units
Ω	4	no units
B	0.1	no units

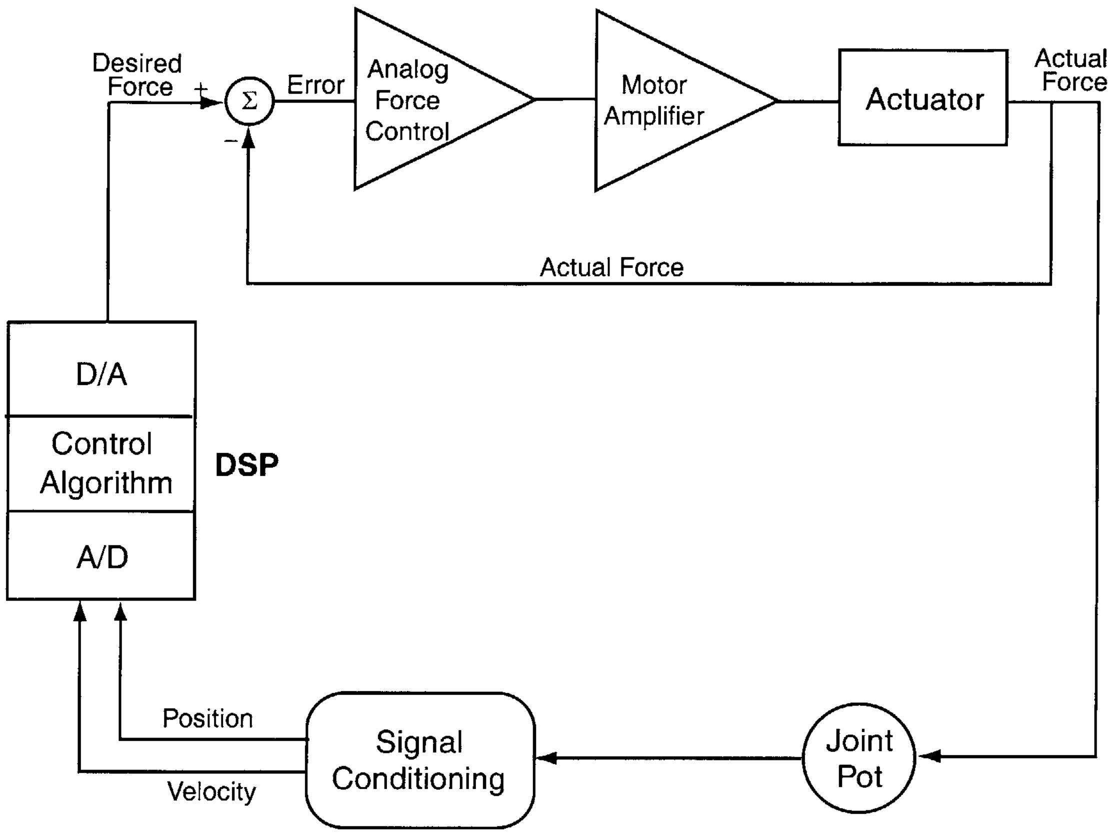

Figure 5-11: Schematic representation of the electronics system of the experimental robot.

- Analog to digital converters to receive joint position and velocity data.
- Digital signal processor to implement a control algorithm.
- Digital to analog converters to send command signals to the actuators from the DSP.
- An analog force control board to receive DSP command signal and send the appropriate command to the motor amplifier.
- Brushless motor amplifier to convert voltage signals from the force control board into current to the motor.
- Power board to supply power to DSP.

Figure 5-11 shows a schematic of the electronics on the robot. A photograph of electrical components can be seen in figure 5-12.

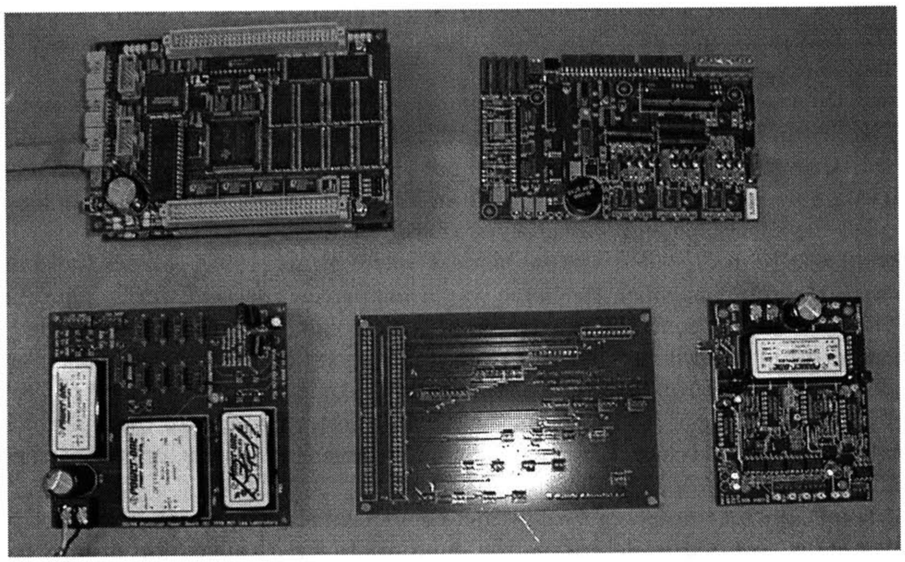

Figure 5-12: Photograph of electrical components of the robot. Starting in the top left and moving clockwise: Integrated A/D, D/A, and DSP, motor amplifier, force control board/signal conditioner, analog break out board, power board.

Chapter 6

Virtual Model Control

During the past 30 years many different schemes to control walking and running robots have been implemented, with varying degrees of success. Among the most effective are trajectory tracking methods, which rely on deriving and solving the dynamic equations of motion and using the solution to define a desired trajectory [Hirai et al. (1998), ichi Yamaguchi et al. (1993), Miura & Shimoyama (1984)]. High gain position controllers, often employing adaptive elements, are used to follow the predefined trajectory.

Trajectory tracking methods have two primary drawbacks. First, they are computationally intensive. The dynamic equations for a robot with many degrees of freedom can be very difficult to solve. Second, trajectory controlled robots are not robust to disturbances or other changes in the system. For example, if the robot collides with an object in its environment or its load changes, it must recalculate its dynamics and trajectory or it will fall over.

An alternate control tool, called Virtual Model Control [Pratt (1995)], allows the control engineer to interact with the robot in much the same way a puppeteer interacts with a marionette puppet. Instead of calculating and controlling a trajectory, the robot is intuitively controlled by external forces. This control method is much softer than typical position control methods. For example, if the "puppeteer" wanted the robot to walk or run faster he could pull it forward. If the robot began to tip to its left, the " puppeteer" could compensate with a restoring force to the right. Virtual Model Control does not require a dynamic model to define suitable trajectories, only that the appropriate force be applied at the appropriate time.

Virtual Model Control consists of two distinct components. First, a high-level control (analogous to the puppeteer) is required to determine the appropriate force application needed to achieve the desired motions. Second, a low-level controller (analogous to the strings of the puppet) is required to to apply the forces commanded by the high-level controller.

In this chapter we will discuss the low-level component of Virtual Model Control. In the next chapter the high-level component of Virtual Model Control will be described for both standing and pronking.

6.1 Introduction to Virtual Model Control

Virtual Model Control is a control tool that allows the control engineer to map "virtual forces" on a robot to real joint torques of a robot. The forces are referred to as virtual because they do not actually exists, it only appears as if they did.

The principal of Virtual Model Control is similar to the principal of equivalent force systems discussed in any engineering mechanics text. For example, a single force acting at the far end of a cantilever can be represented as an equivalent force and moment acting at any point on the cantilever (see figure 6-1). To calculate the equivalent moment at the new location, a transformation from force to moment is required. Equation 6.1 shows that the distance the force was translated (in this case d) gives the transformation from F to M.

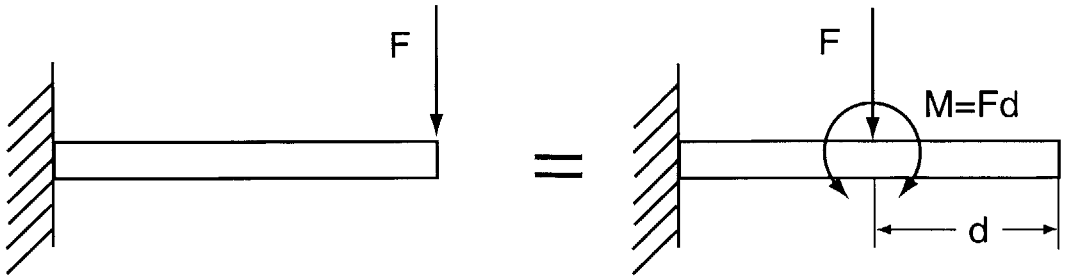

Figure 6-1: Representation of equivalent force systems. A single force (shown in the left figure) can be described by an equivalent force and moment at any location on the beam (shown in the right the figure).

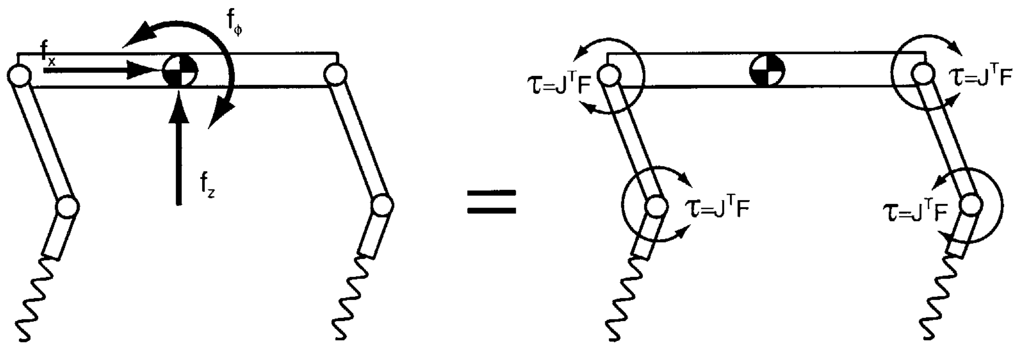

Figure 6-2: Representation of equivalent force systems. A vector forces at the center of mass can be described by torques at the joints. The Jacobian is need to transform forces at the center of mass to joint torques.

$$M = Fd \tag{6.1}$$

Similarly, the virtual forces on a robot can be mapped to the actual torques of a robot's joints. Instead of a simple transformation, d, the mapping of virtual forces to joint torques requires the transpose of the Jacobian [Craig (1989)] of the system. This transformation is depicted in figure 6-2. Equation 6.2 shows the transformation mathematically.

$$\tau = J^T F \tag{6.2}$$

To summarize, the high-level controller commands a vector of virtual forces. Actual joint torques are then calculated from the virtual force vector according to the low-level controller described by equation 6.2. Thus, the reader can see that the low-level Virtual Model Control is simply a mapping of the high-level force command to joint torques through the Jacobian of the robot. These joint torques are produced at the robot's joint with Series Elastic Actuators. The resulting behavior of the robot is as if the virtual forces were actually applied to the robot.

6.2 Implementation of Virtual Model Control

To control the experimental robot shown in figure 5-3 of chapter 5 we must be able to apply forces in each degree of freedom. In this case, the applicable virtual forces are f_x, f_z, and f_ϕ. Since we only

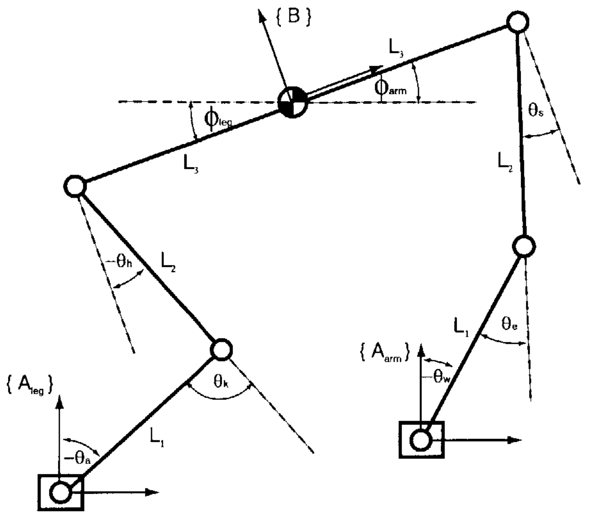

Figure 6-3: Schematic of the kinematic configuration of the experimental robot. θ_a, θ_k, and θ_h represent the ankle, knee and hip angles respectively. θ_w, θ_e, and θ_s represent the wrist, elbow, and shoulder angles respectively. ϕ_{arm} and ϕ_{leg} is the sum of the joint angles of the arm and leg respectively. L_1 is the lower arm and leg length. L_2 is the upper arm and leg length. L_3 is the length from the center of mass to both the hip and shoulder.

have control of joint torques, we must map the desired forces (f_x, f_z, f_ϕ) to individual joint torques according to equation 6.2 above. This is referred to as the low-level control. The experimental robot has four degrees of freedom, making it is possible to control the three forces (f_x, f_z, f_ϕ) independently.

Since both the arm and the leg will contribute to f_x, f_z, and f_ϕ, our plan is to develop low-level Virtual Model Controllers independently for each leg of the robot. These two low-level Virtual Model Controllers will be superposed on one another to provide a single low-level Virtual Model Controller which will map desired f_x, f_z, and f_ϕ to the joint torques of the robot.

6.2.1 Low-level Arm Controller

Figure 6-3 illustrates the kinematics of the experimental robot. Note that the front leg of the robot will be referred to as the arm and its associated joints (wrist, elbow, and shoulder) will be referred to as θ_w, θ_e, and θ_s. The rear leg will be referred to as the leg and its associated joints (ankle, knee, hip) will be referred to as θ_a, θ_k, and θ_h. To model point contact with the ground, an unactuated pin joint is shown at the ankle and wrist of the leg and arm.

The first step in calculating the low-level Virtual Model Controller (i.e. the Jacobian) is to derive the forward kinematic map. Writing the forward kinematic map for the arm from frame $\{A_{arm}\}$ to frame {B} yields in figure 6-3.

$$ {}^{A}_{B}\vec{P}_{arm} = \begin{bmatrix} x_{arm} \\ z_{arm} \\ \phi_{arm} \end{bmatrix} = \begin{bmatrix} -L_1 \sin(\theta_w) - L_2 \sin(\theta_w + \theta_e) - L_3 \cos(\theta_w + \theta_e + \theta_s) \\ L_1 \cos(\theta_w) + L_2 \cos(\theta_w + \theta_e) - L_3 \sin(\theta_w + \theta_e + \theta_s) \\ -\theta_w - \theta_e - \theta_s \end{bmatrix} \quad (6.3) $$

Partial differentiation produces the Jacobian for the arm,

$$ {}^{A}_{B}J_{arm} = \begin{bmatrix} A & B & C \\ D & E & F \\ -1 & -1 & -1 \end{bmatrix} \tag{6.4}$$

where

$$\begin{aligned}
A &= -L_1 \cos(\theta_w) - L_2 \cos(\theta_w + \theta_e) + L_3 \sin(\theta_w + \theta_e + \theta_s) \\
B &= -L_2 \cos(\theta_w + \theta_e) + L_3 \sin(\theta_w + \theta_e + \theta_s) \\
C &= +L_3 \sin(\theta_w + \theta_e + \theta_s) \\
D &= -L_1 \sin(\theta_w) - L_2 \sin(\theta_w + \theta_e) - L_3 \cos(\theta_w + \theta_e + \theta_s) \\
E &= -L_2 \sin(\theta_w + \theta_e) - L_3 \cos(\theta_w + \theta_e + \theta_s) \\
F &= -L_3 \cos(\theta_w + \theta_e + \theta_s)
\end{aligned}$$

The Jacobian maps the virtual forces to joint torques according to

$$\vec{\tau} = ({}^{A}_{B}J)^T \ ({}^{A}_{B}\vec{F}) \tag{6.5}$$

Finally, the low-level Virtual Model Control maps the contribution of the arm ($f_{x_{arm}}$, $f_{z_{arm}}$, $f_{\phi_{arm}}$) to the associated joint torques of the arm (τ_w, τ_e, τ_s) according to

$$\begin{bmatrix} \tau_w \\ \tau_e \\ \tau_s \end{bmatrix} = \begin{bmatrix} A & D & -1 \\ B & E & -1 \\ C & F & -1 \end{bmatrix} \begin{bmatrix} f_{x_{arm}} \\ f_{z_{arm}} \\ f_{\phi_{arm}} \end{bmatrix} \tag{6.6}$$

6.2.2 Low-level Leg Controller

Similarly, writing the forward kinematic map for the leg from frame $\{A_{leg}\}$ to frame {B} in figure 6-3 yields.

$$ {}^{A}_{B}\vec{P}_{leg} = \begin{bmatrix} x_{leg} \\ z_{leg} \\ \theta_{leg} \end{bmatrix} = \begin{bmatrix} -L_1 \sin(\theta_a) - L_2 \sin(\theta_a + \theta_k) + L_3 \cos(\theta_a + \theta_k + \theta_h) \\ L_1 \cos(\theta_a) + L_2 \cos(\theta_a + \theta_k) + L_3 \sin(\theta_a + \theta_k + \theta_h) \\ -\theta_a - \theta_k - \theta_h \end{bmatrix} \tag{6.7}$$

Partial differentiation produces the Jacobian for the leg,

$$ {}^{A}_{B}J_{leg} = \begin{bmatrix} M & N & O \\ P & Q & R \\ -1 & -1 & -1 \end{bmatrix} \tag{6.8}$$

where

$$\begin{aligned}
M &= -L_1\cos(\theta_a) - L_2\cos(\theta_a+\theta_k) - L_3\sin(\theta_a+\theta_k+\theta_h)\\
N &= -L_2\cos(\theta_a+\theta_k) - L_3\sin(\theta_a+\theta_k+\theta_h)\\
O &= -L_3\sin(\theta_a+\theta_k+\theta_h)\\
P &= -L_1\sin(\theta_a) - L_2\sin(\theta_a+\theta_k) + L_3\cos(\theta_a+\theta_k+\theta_h)\\
Q &= -L_2\sin(\theta_a+\theta_k) + L_3\cos(\theta_a+\theta_k+\theta_h)\\
R &= +L_3\cos(\theta_a+\theta_k+\theta_h)
\end{aligned}$$

Again, the Jacobian maps the virtual forces to joint torques according to

$$\vec{\tau} = ({}^A_BJ)^T\,({}^A_B\vec{F})$$

Finally, the low-level Virtual Model Control maps the contribution of the leg ($f_{x_{leg}}$, $f_{z_{leg}}$, $f_{\phi_{leg}}$) to the associated joint torques of the leg (τ_a, τ_k, τ_h) according to

$$\begin{bmatrix}\tau_a\\ \tau_k\\ \tau_h\end{bmatrix} = \begin{bmatrix} M & P & -1\\ N & Q & -1\\ O & R & -1\end{bmatrix}\begin{bmatrix} f_{x_{leg}}\\ f_{z_{leg}}\\ f_{\phi_{leg}}\end{bmatrix} \tag{6.9}$$

6.2.3 Superposition of Arm and Leg Controller

Combining equations 6.6 and 6.9 into a single matrix yields

$$\begin{bmatrix}\tau_w\\ \tau_e\\ \tau_s\\ \tau_a\\ \tau_k\\ \tau_h\end{bmatrix} = \begin{bmatrix} A & D & -1 & 0 & 0 & 0\\ B & E & -1 & 0 & 0 & 0\\ C & F & -1 & 0 & 0 & 0\\ 0 & 0 & 0 & M & P & -1\\ 0 & 0 & 0 & N & Q & -1\\ 0 & 0 & 0 & O & R & -1\end{bmatrix}\begin{bmatrix} f_{x_{arm}}\\ f_{z_{arm}}\\ f_{\phi_{arm}}\\ f_{x_{leg}}\\ f_{z_{leg}}\\ f_{\phi_{leg}}\end{bmatrix} \tag{6.10}$$

For simplicity, it is preferable to specify a single force along each direction, x, z, and ϕ. Because of our selection of a coincident reaction frame at {B} for both the arm and the leg, we can easily sum the forces from the arm and the leg to obtain a single virtual force along each direction. Thus, superposition of the arm and leg forces yields

$$\begin{bmatrix} f_x\\ f_z\\ f_\phi\end{bmatrix} = \begin{bmatrix} f_{x_{arm}}\\ f_{z_{arm}}\\ f_{\phi_{arm}}\end{bmatrix} + \begin{bmatrix} f_{x_{leg}}\\ f_{z_{leg}}\\ f_{\phi_{leg}}\end{bmatrix} \tag{6.11}$$

Now, we wish to control only three forces (f_x , f_z, and f_ϕ), but we apparently have six actuated degrees of freedom (τ_w, τ_e, τ_s, τ_a, τ_k, τ_h). This implies that three additional constraints must be imposed to solve for f_x , f_z, and f_ϕ in closed form.

It is known that the ankle and wrist joints are unactuated point contacts with the ground. Thus, the wrist and ankle torques must be zero.

$$\tau_w = 0 \tag{6.12}$$

$$\tau_a = 0 \tag{6.13}$$

The third constraint can be specified by the designer to give a desired performance. Here, we choose to match the hip and shoulder torques.

$$\tau_s = \tau_h \tag{6.14}$$

Other possibilities exist. For example, we could have specified a certain ratio between hip and shoulder torques, or maintained equal torque at the knee and elbow.

The constraints in equations 6.11, 6.12, 6.13, and 6.14 can be rewritten in vector form as follows.

$$\begin{bmatrix} f_x \\ f_z \\ f_\phi \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ A & D & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & M & P & -1 \\ C & F & -1 & -O & -R & +1 \end{bmatrix} \begin{bmatrix} f_{x_{arm}} \\ f_{z_{arm}} \\ f_{\phi_{arm}} \\ f_{x_{leg}} \\ f_{z_{leg}} \\ f_{\phi_{leg}} \end{bmatrix} \tag{6.15}$$

Inverting the constraint matrix in equation 6.15 and solving for the virtual forces exerted by the individual legs yields

$$\begin{bmatrix} f_{x_{arm}} \\ f_{z_{arm}} \\ f_{\phi_{arm}} \\ f_{x_{leg}} \\ f_{z_{leg}} \\ f_{\phi_{leg}} \end{bmatrix} = \begin{bmatrix} K_1 & K_2 & K_3 \\ K_4 & K_5 & K_6 \\ K_7 & K_8 & K_9 \\ K_{10} & K_{11} & K_{12} \\ K_{13} & K_{14} & K_{15} \\ K_{16} & K_{17} & K_{18} \end{bmatrix} \begin{bmatrix} f_x \\ f_z \\ f_\phi \end{bmatrix} \tag{6.16}$$

where

$$\begin{aligned}
K_1 &= \frac{+OP+2MB-MR-MF-BO}{K_{det}}\\
K_2 &= \frac{+2BP-BR-FP}{K_{det}}\\
K_3 &= \frac{P+R-B+F}{K_{det}}\\
K_4 &= \frac{-2MA+AO+MC}{K_{det}}\\
K_5 &= \frac{-2AP+AR+OP+CP-MR}{K_{det}}\\
K_6 &= \frac{A+M-O-C}{K_{det}}\\
K_7 &= \frac{MBC-MRA-MFA+AOP}{K_{det}}\\
K_8 &= \frac{-AFP+BOP+CBP-BRM}{K_{det}}\\
K_9 &= \frac{-AP+AR+AF+MB-BO-CB}{K_{det}}\\
K_{10} &= \frac{-2AP+AR+AF+CP-CB}{K_{det}}\\
K_{11} &= \frac{-2BP+BR+FP}{K_{det}}\\
K_{12} &= \frac{P-R+B-F}{K_{det}}\\
K_{13} &= \frac{+2MA-AO-MC}{K_{det}}\\
K_{14} &= \frac{AF+2MB-MF-BO-CB}{K_{det}}\\
K_{15} &= \frac{A-M+O+C}{K_{det}}\\
K_{16} &= \frac{-MBC+MRA+MFA-AOP}{K_{det}}\\
K_{17} &= \frac{AFP-BOP-CBP+BRM}{K_{det}}\\
K_{18} &= \frac{-MR+MF+MB+OP-AP+CP}{K_{det}}\\
K_{det} &= -MR+2MB-MF+OP-BO-CB-2AP+AR+AF+CP
\end{aligned}$$

Substituting equation 6.16 into equation 6.10 and simplifying, gives the virtual force to joint torque relation. Dropping τ_w and τ_a we arrive at equation 6.17.

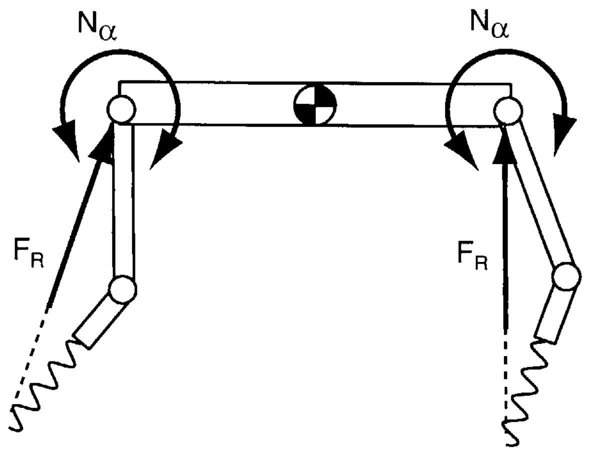

Figure 6-4: Depiction of an alternate location for virtual forces.

$$\begin{bmatrix} \tau_e \\ \tau_s \\ \tau_k \\ \tau_h \end{bmatrix} = \begin{bmatrix} BK_1 + EK_4 - K_7 & BK_2 + EK_5 - K_8 & BK_3 + EK_6 - K_9 \\ CK_1 + FK_4 - K_7 & CK_2 + FK_5 - K_8 & CK_3 + FK_6 - K_9 \\ NK_{10} + QK_{13} - K_{16} & NK_{11} + QK_{14} - K_{17} & NK_{12} + QK_{15} - K_{18} \\ OK_{10} + RK_{13} - K_{16} & OK_{11} + RK_{14} - K_{17} & OK_{12} + RK_{15} - K_{18} \end{bmatrix} \begin{bmatrix} f_x \\ f_z \\ f_\phi \end{bmatrix} \tag{6.17}$$

Equation 6.17 is the low-level Virtual Model Controller for the experimental robot. With it, we can directly calculate joint torques given the desired forces, f_x, f_z, and f_ϕ from the high-level Virtual Model Controller. In other words, 6.17 is the "string" that allows the puppeteer to control the robot.

6.3 An Alternate Implementation of Virtual Model Control

In section 6.2 a low-level Virtual Model Controller was developed which mapped a force vector, f_x, f_z, and f_ϕ, at the center of mass of the robot to desired joint torques τ_e, τ_s, τ_k, τ_h. Although this method is elegant, it couples the behavior of the arm and the leg, making it difficult to understand how the arm and the leg individually contribute to the behavior of the robot.

Other low-level Virtual Model Controllers can be derived which allow for different levels of abstraction. In this section, we will derive a low-level Virtual Model Controller which creates virtual forces F_R and N_α at the hip and shoulder of the robot (see figure 6-4). In this way, the behavior of the arm and leg are decoupled from one another. This is especially convenient when the robot has only one leg in contact with the ground, or any other time you wish to independently control the behavior of the arm and leg. Using geometry, these virtual forces can be mapped to forces at the center of mass of the robot.

6.3.1 Low-level Arm Controller

Figure 6-5 illustrates the kinematics of the experimental robot. Note that the front leg of the robot will be referred to as the arm and its associated joints (elbow and shoulder) will be referred to as

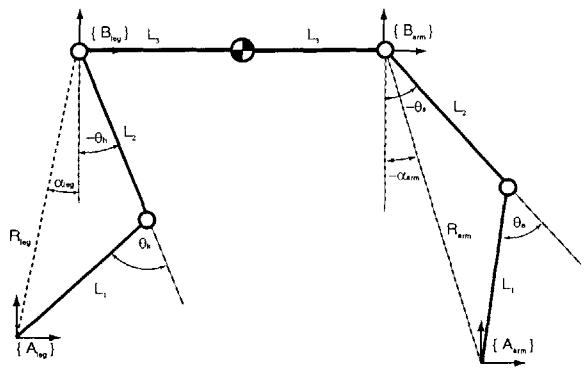

Figure 6-5: Schematic of the kinematic configuration of the experimental robot. θ_k and θ_h represent the knee and hip angles respectively. θ_e and θ_s represent the elbow and shoulder angles respectively. R_{arm} and R_{leg} is the line connecting the hand to the shoulder and the foot to the hip, respectively. α_{arm} and α_{leg} are the angles measured from vertical to R_{arm} and R_{leg}, respectively. L_1 is the lower arm and leg length, L_2 is the upper arm and leg length, and L_3 is the distance from the hip and shoulder to the center of mass.

θ_e and θ_s. The rear leg will be referred to as the leg and its associated joints (knee and hip) will be referred to as θ_k and θ_h.

The first step in calculating the low-level Virtual Model Controller is to derive the forward kinematic map. Writing the forward kinematic map for the arm from frame $\{A_{arm}\}$ to frame $\{B_{arm}\}$ yields.

$$^{A}_{B}\vec{P}_{arm} = \begin{bmatrix} \alpha_{arm} \\ R_{arm} \end{bmatrix} = \begin{bmatrix} tan^{-1}(\frac{X_{arm}}{Z_{arm}}) \\ (X_{arm}^2 + Z_{arm}^2)^{\frac{1}{2}} \end{bmatrix} \tag{6.18}$$

where, after simplification

$$X_{arm} = L_1 \sin(\theta_s) + L_2 \sin(\theta_s + \theta_e) \tag{6.19}$$

$$Z_{arm} = L_1 \cos(\theta_s) + L_2 \cos(\theta_s + \theta_e) \tag{6.20}$$

Partial differentiation of α_{arm} and R_{arm} with respect to θ_e and θ_s produces the Jacobian for the arm.

$$^{A}_{B}J_{arm} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \tag{6.21}$$

where

$$A = 1.0 \tag{6.22}$$

$$B = \frac{L_2^2 + L_1 L_2 cos(\theta_e)}{(L_1^2 + L_2^2 + 2L_1 L_2 cos(\theta_e))^{\frac{1}{2}}} \tag{6.23}$$

$$C = 0.0 \tag{6.24}$$

$$D = \frac{-L_1 L_2 sin(\theta_e)}{(L_1^2 + L_2^2 + 2L_1 L_2 cos(\theta_e))^{\frac{1}{2}}}$$

The Jacobian maps the virtual forces to joint torques according to

$$\vec{\tau} = ({}^A_B J)^T \; ({}^A_B \vec{F}) \tag{6.25}$$

Finally, the low-level Virtual Model Control maps the virtual forces F_R and N_α to the associated joint torques of the arm (τ_e and τ_s) according to

$$\begin{bmatrix} \tau_s \\ \tau_e \end{bmatrix} = \begin{bmatrix} A & C \\ B & D \end{bmatrix} \begin{bmatrix} N_{\alpha_{arm}} \\ F_{R_{arm}} \end{bmatrix} \tag{6.26}$$

The derivation of the leg Jacobian is identical to that of the arm, with θ_k replacing θ_e and θ_h replacing θ_s.

6.3.2 Virtual Force to Body Force Transformation

To gain further insight into the control problem, it is important to understand how the leg forces act on the body of the robot. To simplify this transformation, we have assumed that the legs are massless and treat each leg as if it possessed a prismatic knee. Figure 6-6 shows a free body diagram of the planar quadruped assuming both feet are in contact with the ground.

Summing forces in the x, z, and ϕ directions for the arm and the leg yields

Arm

$$f_{x_{arm}} = -F_{R_{arm}} sin(\alpha_{arm}) - F_{N\alpha arm} cos(\alpha_{arm}) \tag{6.27}$$

$$f_{z_{arm}} = -F_{R_{arm}} cos(\alpha_{arm}) + F_{N\alpha arm} sin(\alpha_{arm}) \tag{6.28}$$

$$f_{\phi_{arm}} = -N_{\alpha_{arm}} - F_{R_{arm}} cos(\alpha_{arm}) L_3 - F_{N\alpha arm} sin(\alpha_{arm}) L_3 \tag{6.29}$$

where

$$F_{N\alpha arm} = \frac{N_{\alpha_{arm}}}{R_{arm}} \tag{6.30}$$

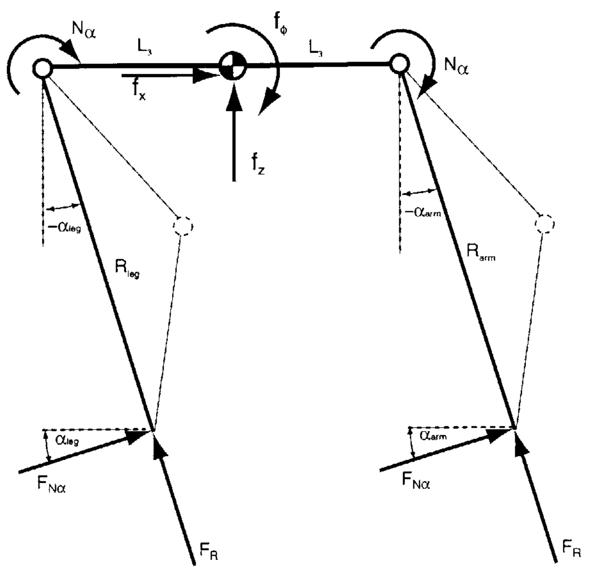

Figure 6-6: Schematic used to transform leg forces into body forces.

Leg

$$f_{x_{leg}} = -F_{R_{leg}} sin(\alpha_{leg}) - F_{N\alpha leg} cos(\alpha_{leg}) \tag{6.31}$$

$$f_{z_{leg}} = -F_{R_{leg}} cos(\alpha_{leg}) + F_{N\alpha leg} sin(\alpha_{leg}) \tag{6.32}$$

$$f_{\phi_{leg}} = -N_{\alpha_{leg}} - F_{R_{leg}} cos(\alpha_{leg}) L_3 - F_{N\alpha leg} sin(\alpha_{leg}) L_3 \tag{6.33}$$

where

$$F_{N\alpha leg} = \frac{N_{\alpha_{leg}}}{R_{leg}} \tag{6.34}$$

Thus, the sum of the forces and moments on the robot is

$$f_x = f_{x_{arm}} + f_{x_{leg}} \tag{6.35}$$

$$f_z = f_{z_{arm}} + f_{z_{leg}} \tag{6.36}$$

$$f_\phi = f_{\phi_{arm}} + f_{\phi_{leg}} \tag{6.37}$$

6.4 Summary of Virtual Model Control

Control of a walking or running robot using trajectory tracking can be computationally intensive and is susceptible to disturbances. A more intuitive, softer method, known as Virtual Model Control, uses a high-level controller to generate desired virtual forces on the robot. The virtual forces f_x, f_z and f_ϕ are mapped to joint torques of the robot using equation 6.17. Alternately, the virtual N_α and F_R can be mapped to joint torques with equation 6.26. In this way, the control engineer can specify desired virtual forces and the joint torques will be mapped to give these forces. This control scheme has been likened to the interaction of a puppeteer and a marionette puppet. The puppeteer being analogous to the high-level controller and the elastic strings attached to the puppet being analogous to the low-level controller. In the next chapter we will have a closer look at the high-level controller.

Chapter 7

Simulated and Experimental Control

In Chapter 6 Virtual Model Control is described in some detail. The low-level controller is compared to the strings of a marionette puppet, which can have forces applied to them by a high-level controller, analogous to the puppeteer. The low-level controller to map desired virtual forces to joint torques was derived for the experimental robot.

In this chapter, we describe the high-level control component of Virtual Model Control for standing and pronking algorithms. These simple controllers lay the foundation for future work on quadrupedal locomotion and are used on a simulated robot and the experimental robot to estimate the minimum torque and power requirements for quadrupedal locomotion.

It should be noted that the spring and natural dynamics discussed in chapters 3 and 4 will not be considered in our control algorithms. For the time being, it makes sense to establish a benchmark which to compare future improvements. Furthermore, eliminating these unknowns will simplify the controller. For this reason, the Cheetah Foot shown in figure 5-9 of chapter 5 was replaced with rigid feet.

7.1 Standing Control

7.1.1 Standing Control Algorithm

We will begin our study of high-level control with the development of an algorithm to control standing of the experimental robot. The high-level controller generally consists of two key elements.

1. A *state machine* to read input data and determine the current state of the robot.
2. A *control action* to calculate desired forces based on the current state.

For the simulated robot, standing consists of only one state. That is, no matter what the sensor data of the robot reads, the same control action is taken. For this reason, the state machine exists, but plays no role in the high-level control. To control the experimental robot it is necessary to control each planar degree of freedom, namely x, z, and ϕ with virtual forces f_x, f_z, and f_ϕ. For standing, coupling between the arm and the leg does not matter, therefore, we will use the low-level Virtual Model Controller derived in section 6.2 of chapter 6.

Any function can be written to describe f_x, f_z, and f_ϕ. Intuitively, it is easiest to describe these virtual forces in terms of physical components that are already well understood. For standing, it makes sense that the forces applied to the robot should behave like springs and dampers. A virtual spring and damper attached at each degree of freedom will tolerate disturbances, but also return the robot to its original position. A depiction of the desired virtual forces (as described by springs and dampers) can be seen in figure 7-1. With virtual springs and dampers attached in this configuration, the robot can withstand large force or position disturbances.

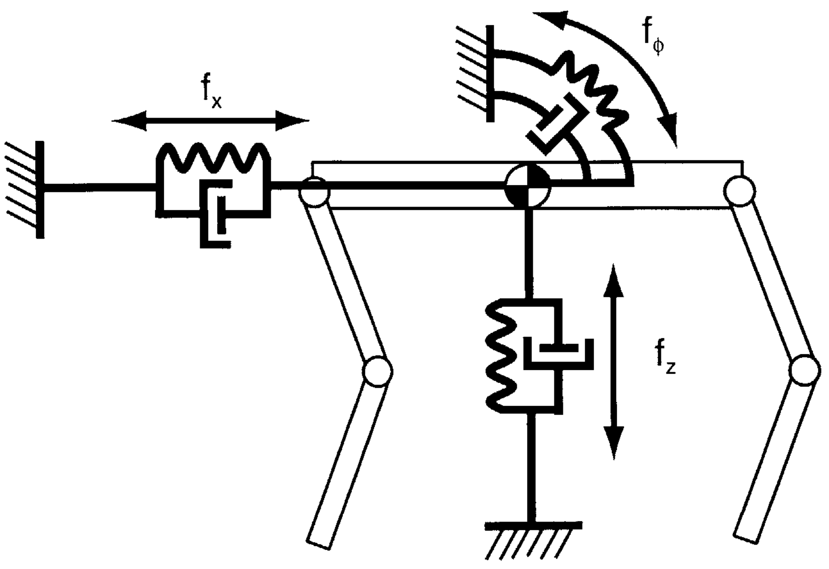

Figure 7-1: Experimental robot with virtual springs and dampers controlling f_x, f_z, and f_ϕ. The virtual springs and dampers hold the robot solidly in a standing posture, but are compliant enough to withstand position and force disturbances.

It is a simple task to write the equations which describe these virtual forces. Let us begin with a generic formulation of a virtual force describing a spring and damper. The force equations will take the form

$$f_d = k(q_d - q) - b\dot{q} + f_{ff} \tag{7.1}$$

$$q_d = q_{DC} + Asin(2\pi f t) \tag{7.2}$$

where q is a generic position variable and q_d is a desired position, consisting of a DC offset and a sine wave of amplitude A and frequency f as in equation 7.2. In this way, $k(q_d - q)$ represents a spring with a variable set point q_d, $-b\dot{q}$ represents a damper, and f_{ff} is a feed forward term.

Using the generic formulation of 7.1 and 7.2, control equations can be written for f_x, f_z, and f_ϕ.

Horizontal Control

To control the horizontal position of the robot during standing, a force equation is written to describe the virtual spring and damper in the x direction.

```
Fx=k_x*(q_d.x-q.x)-b_x*qd.x;
```

where the desired horizontal position for the robot's center of mass is calculated according to

```
q_d.x=X_set+X_amp*sin(2.0*PI*X_freq*t);
```

With this equation, the set point, amplitude and frequency of the desired position can be controlled.

Vertical Control

Similarly, to control the vertical position of the robot during standing, a force equation is written to describe the virtual spring and damper in the z direction. Additionally, a feed forward term is included to compensate for the weight of the robot.

```
Fz=k_z*(q_d.z-q.z)-b_z*qd.z+Fz_ff;
```

where the desired position for the robot's center of mass is calculated according to

```
q_d.z=Z_set+Z_amp*sin(2.0*PI*Z_freq*t);
```

With this equation, the set point, amplitude and frequency of the desired position can be controlled.

The feed forward term is calculated to balance the weight of the robot.

```
Fz_ff=body_mass*9.81;
```

Pitch Control

The pitch of robot is controlled in a similar way. A force equation is written to describe the torsional virtual spring and damper.

```
Fphi=k_phi*(q_d.phi-q.phi)-b_phi*qd.phi;
```

where the desired pitch angle is calculated according to

```
q_d.phi=PHI_set+PHI_amp*sin(2.0*PI*PHI_freq*t);
```

Again, the set point, amplitude and frequency of the desired body pitch can easily be controlled.

Since there is only a single state, each of these forces is continuously applied to the robot. Using the forces calculated above, the appropriate joint torques are mapped according to equation 6.17. The joint torques are applied to the robot to achieve the spring and damper behavior. Position is fed-back to the high-level controller and new virtual force commands are generated. A schematic showing the flow of this control scheme can be seen in figure 7-2.

7.1.2 Simulated Standing Control

The above algorithm was implemented in simulation. To test the validity of the algorithm, the simulated robot was commanded to execute a position step of 0.05 meters in the x and z directions and an angular step of 0.1 radians in the ϕ direction. The spring constant (k), damping term (b), and feed forward force (f_{ff}) were tuned to get the appropriate second-order step response. Row one of figure 7-3 shows the simulated step responses in the x, z, and ϕ directions. It should be noted that the z step response has a large steady state error because the feed forward term was not large enough. Each of the figures closely resembles a typical second-order step response. From this data, we can infer that the Virtual Model Control is working as expected.

Joint torque and power were also measured for each step response. Graphically, the data from the experiments can also be seen in figure 7-3. For convenience, a summary of the maximum joint torque and maximum joint power for the simulated step response can be seen in table 7.1. From this data, it can be seen that the robot uses significant torque at the knee and elbow for positional steps. Still, the power consumption at the knee and elbow remain low. Shoulder and hip torque and power are nearly insignificant for the z and ϕ directions, but much larger for the positional steps in x.

To further investigate controllability of standing and power consumption, an additional simulation experiment was conducted. By changing the amplitude (A) and the frequency (f) of equation

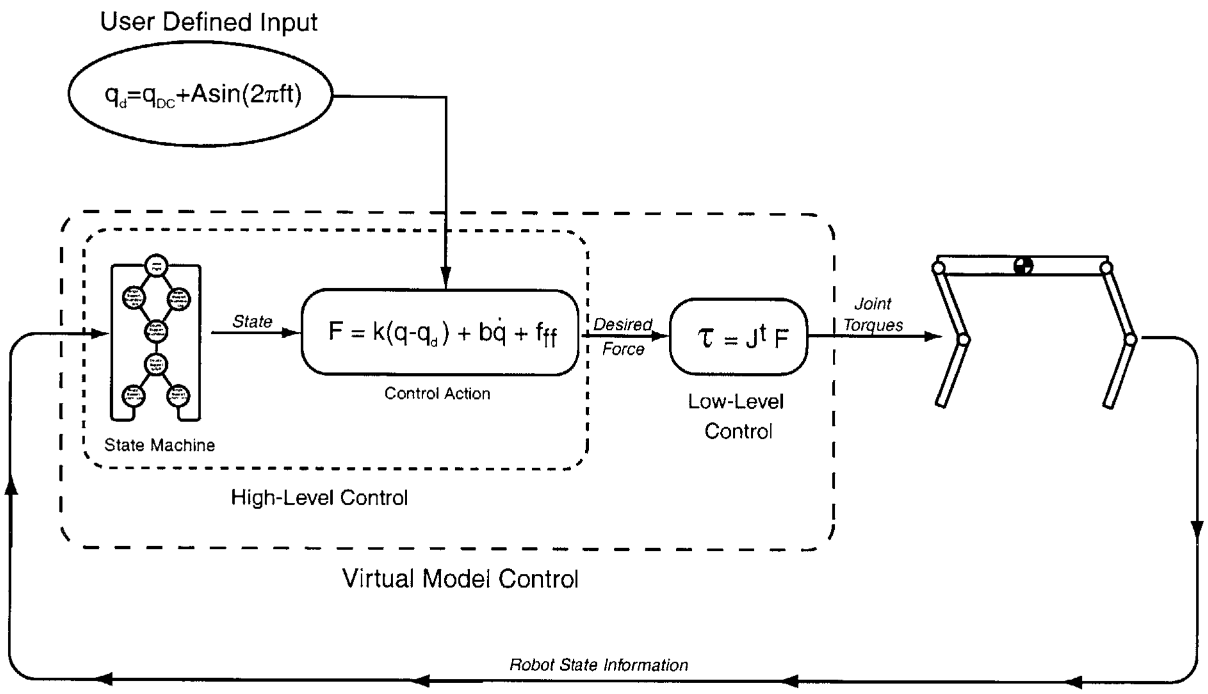

Figure 7-2: Control flow of standing algorithm. *Robot State Information* enters the High-Level Controller where the state machine determines the *state* of the robot. Based on the *state*, some control action calculates the *desired force*. Next, the *desired force* enters the Low-Level Controller, where the *joint torques* are calculated using the Jacobian. Finally, the *joint torques* are applied to the robot and new *robot state information* is gathered.

Table 7.1: Summary of Torque and Power Requirements for Simulated Step Responses

	Figure Number	Max Torque (knee/elbow)	Max Power (knee/elbow)	Max Torque (hip/shoulder)	Max Power (hip/shoulder)
X step	7-3 (left)	25 N/m	10 Watts	30 N/m	10 Watts
Z step	7-3 (middle)	20 N/m	30 Watts	5 N/m	5 Watts
ϕ step	7-3 (right)	20 N/m	20 Watts	2 N/m	3 Watts

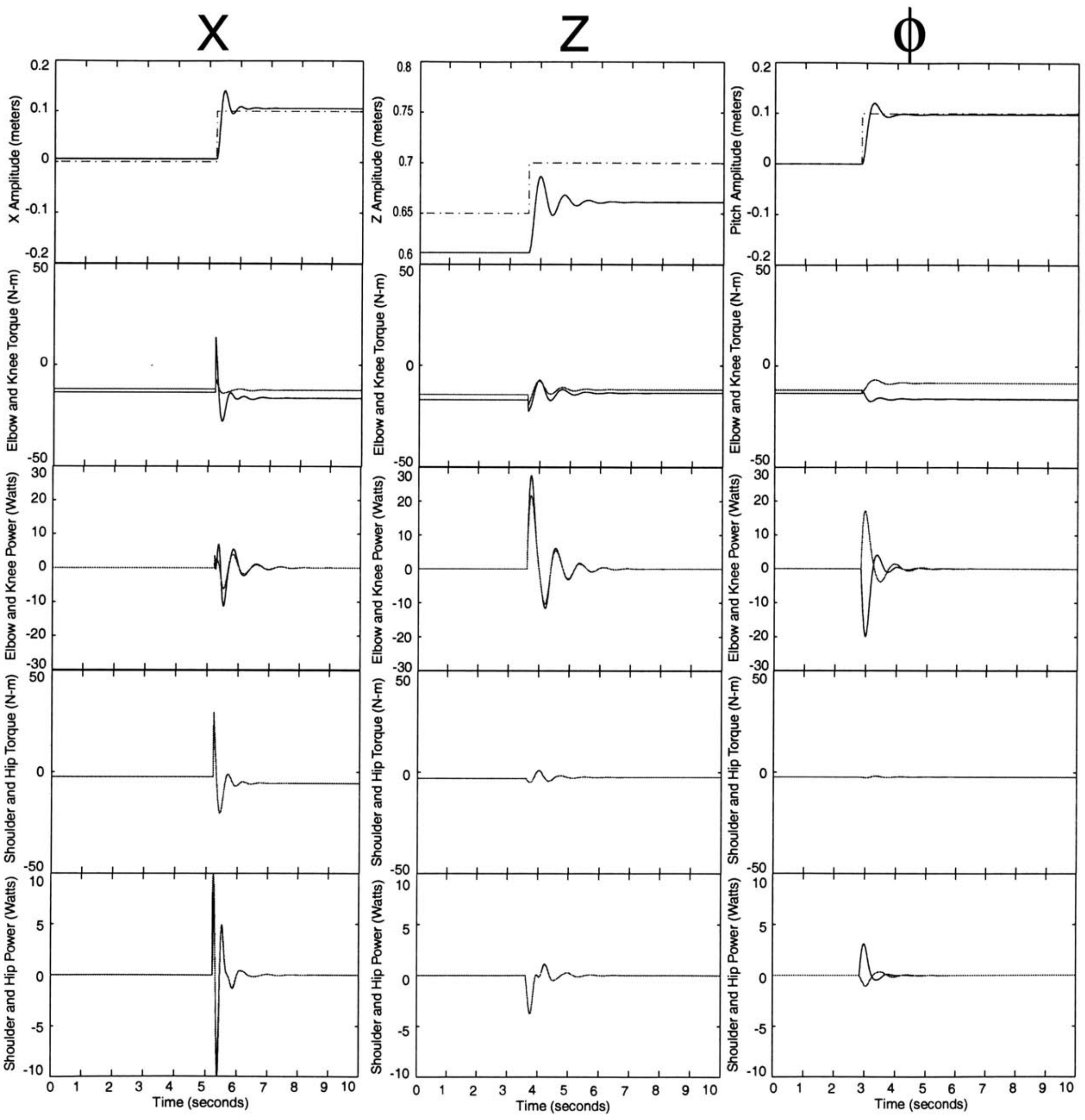

Figure 7-3: Left column shows simulated X step response of 0.05 meters with $k_x = 1000$, $b_x = 100$. Middle column shows simulated Z step response of 0.05 meters with $k_z = 1000$, $b_z = 50$. Right column shows simulated pitch step response of 0.1 radians with $k_phi = 100$, $b_phi = 10$. Joint torques and associated joint power are also shown for the knee, elbow, hip and shoulder in each column.

Table 7.2: Summary of Torque and Power Requirements for Simulated Frequency Responses

	Figure Number	Max Torque (knee/elbow)	Max Power (knee/elbow)	Max Torque (hip/shoulder)	Max Power (hip/shoulder)
X frequency	7-4 (left)	20 N/m	1 Watt	5 N/m	2 Watts
Z frequency	7-4 (middle)	20 N/m	15 Watts	2 N/m	3 Watts
ϕ frequency	7-4 (right)	15 N/m	15 Watts	2 N/m	3 Watts

Table 7.3: Summary of Torque and Power Requirements for Experimental Step Responses

	Figure Number	Max Torque (knee/elbow)	Max Power (knee/elbow)	Max Torque (hip/shoulder)	Max Power (hip/shoulder)
X step	7-5 (left)	20 N/m	20 Watts	10 N/m	10 Watts
Z step	7-5 (middle)	15 N/m	20 Watts	5 N/m	5 Watts
ϕ step	7-5 (right)	25 N/m	30 Watts	2 N/m	5 Watts

7.2, the robot was commanded to follow a 0.5 Hz sine wave of 0.05 meter amplitude in the x and z directions and 0.1 radian amplitude in the ϕ direction. The desired set point versus actual set point can be seen in the first row of figure 7-4. From these figures we can see that the simulated robot has little difficulty following the desired trajectories using Virtual Model Control.

Joint torques, and associated joint power, were recorded for each simulation experiment and can also be seen in figure 7-6. A summary of the maximum joint torque and maximum joint power for the simulated frequency response can be seen in table 7.2. We find that significant knee and elbow torque, but little power, is required to follow the desired trajectory. Hip and shoulder torques are nearly insignificant for all three cases.

7.1.3 Experimental Standing Control

After the standing algorithm was verified in simulation, it was implemented on the experimental robot. To test the abililty of the experimental hardware to execute the standing algorithms two experiments were conducted. First, the experimental robot was commanded to execute a position step of 0.05 meters in the x and z directions and an angular step of 0.1 radians in the ϕ direction. The spring constant (k), damping term (b), and feed forward force (f_{ff}) were tuned to get the appropriate second-order step response. Row one of figure 7-5 shows the experimental step responses in the x, z, and ϕ direction. Each of the figures closely resembles a typical second-order step response. From this data, we can infer that the Virtual Model Control and experimental hardware is working as expected.

Joint torque and power were also measured for each step response. Graphically, the data from the experiments can also be seen in figure 7-5. For convenience, a summary of the experimental results, showing maximum joint torque and maximum joint power can be seen in table 7.3. Close agreement can be seen between the simulated data in table 7.1 and the experimental data in table 7.3. From this data, it can be seen that the experimental robot uses significant torque at the knee and elbow for positional steps. Still, the power consumption at the knee and elbow remain low. Shoulder and hip torque and power are nearly insignificant for the z and ϕ directions, but much larger for the positional steps in x.

To further investigate controllability of standing and power consumption, an additional experiment was conducted. The robot was also commanded to follow a 0.5 Hz sine wave of 0.05 meter

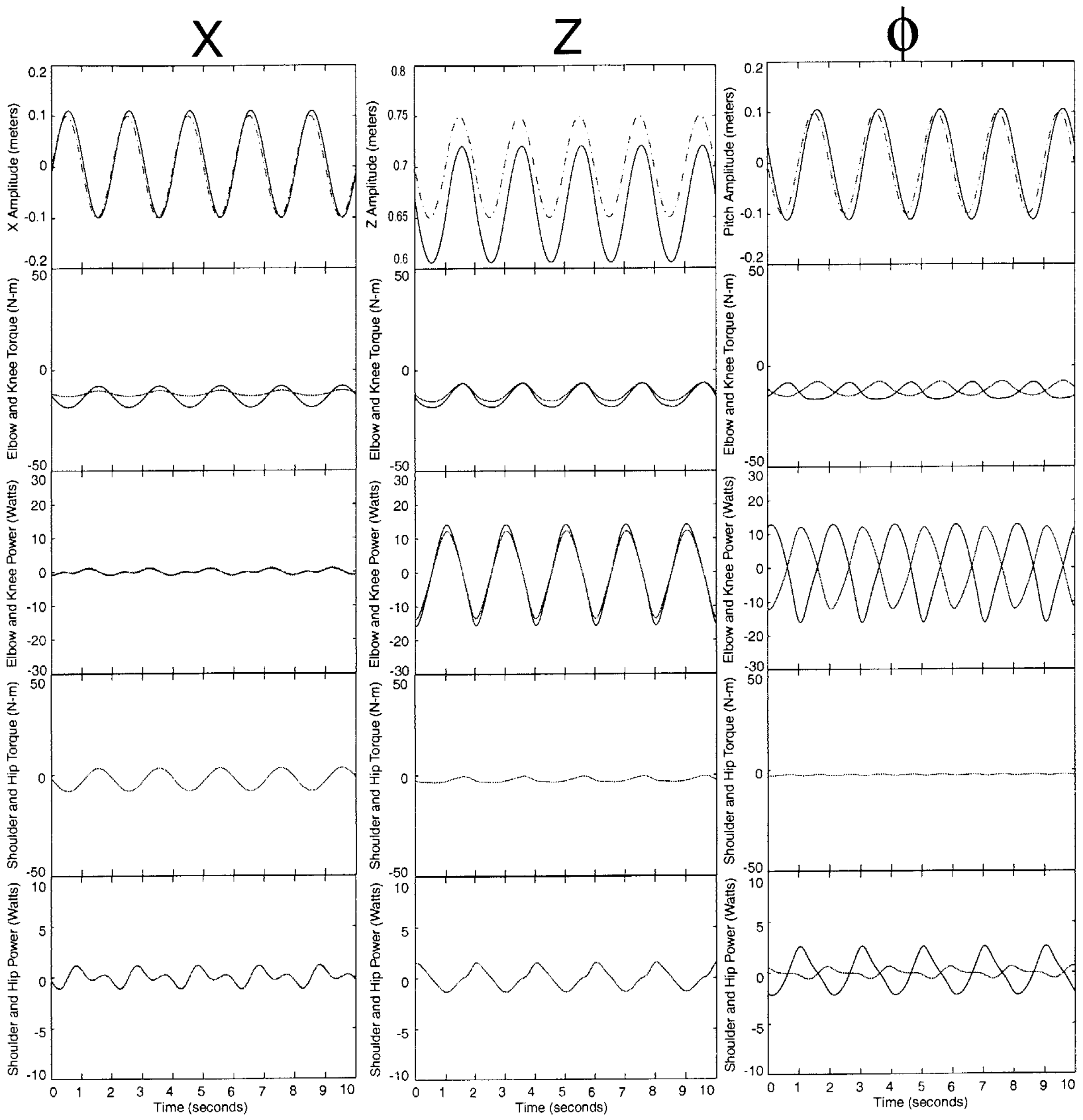

Figure 7-4: Left column shows simulated X frequency response at an amplitude of 0.05 meters and frequency of 0.5 hertz with $k_x = 100$, $b_x = 10$. Middle column shows simulated Z frequency response at an amplitude of 0.05 meters and frequency of 0.5 hertz with $k_z = 1000$, $b_z = 50$ Right column shows simulated pitch frequency response at an amplitude of 0.1 radians and frequency of 0.5 hertz with k_phi=100, $b_phi = 10$. Joint torques and associated joint power are also shown for the knee, elbow, hip and shoulder in each column.

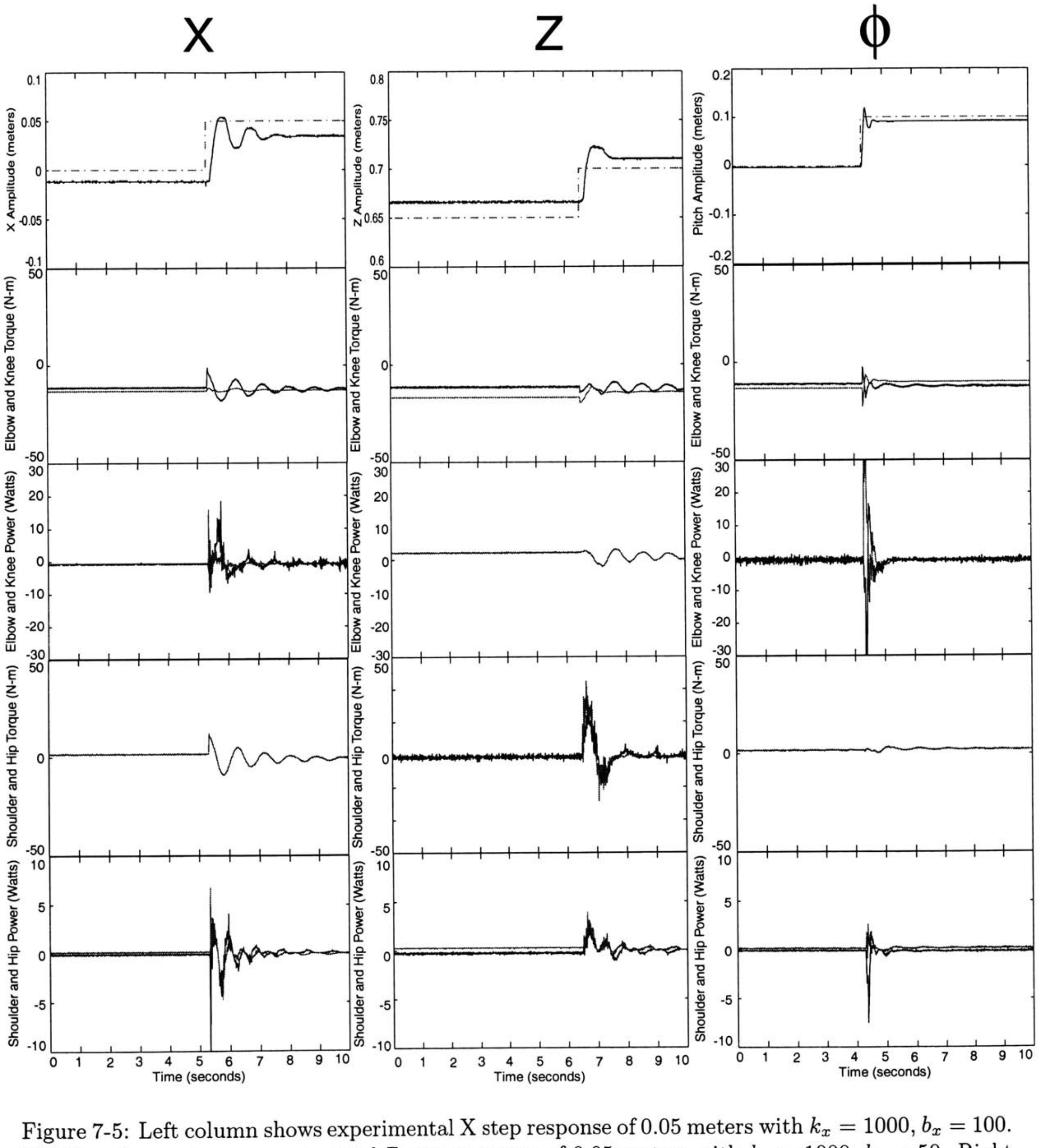

Figure 7-5: Left column shows experimental X step response of 0.05 meters with $k_x = 1000$, $b_x = 100$. Middle column shows experimental Z step response of 0.05 meters with $k_z = 1000$, $b_z = 50$. Right column shows experimental pitch step response of 0.1 radians with $k_phi = 100$, $b_phi = 10$. Joint torques and associated joint power are also shown for the knee, elbow, hip and shoulder in each column.

Table 7.4: Summary of Torque and Power Requirements for Experimental Frequency Responses

	Figure Number	Max Torque (knee/elbow)	Max Power (knee/elbow)	Max Torque (hip/shoulder)	Max Power (hip/shoulder)
X frequency	7-6 (left)	20 N/m	1 Watt	5 N/m	2 Watts
Z frequency	7-6 (middle)	20 N/m	20 Watts	10 N/m	5 Watts
ϕ frequency	7-6 (right)	25 N/m	20 Watts	2 N/m	3 Watts

amplitude in the x and z directions and 0.1 radian amplitude in the ϕ direction. The desired position versus actual position can be seen in row one of figure 7-6. From these figures we can see that the experimental robot has little difficulty following the desired trajectory using Virtual Model Control.

Joint torques and associated joint power were recorded for each experiment and can also be seen in figure 7-6. A summary of the experimental results can be seen in table 7.4. Good agreement can be seen with the simulated results shown in table 7.2. We find that significant knee and elbow torque, but little power, is required to follow the desired trajectory. Hip and shoulder torques are nearly insignificant for all three cases.

7.1.4 Summary of Standing Control

A high-level algorithm to control standing has been described. The algorithm uses a singular state machine. Since coupling of the arm and the leg is not a problem, the low-level Virtual Model Control in section 6.2 of chapter 6 is used. Virtual springs and dampers are used to control x, z, and ϕ. Simulation experiments were conducted to verify the high-level and low-level components of Virtual Model Control. The simulations allowed us to quickly tune the spring constant and damper coefficient for each virtual element. Torque and power requirements were measured in simulation for positional stepping and sinusoidal tracking. The maximum values were found to be well within the limits of the Series Elastic Actuators. Hip torque is underutilized during control of the z and ϕ directions.

The same algorithm was implemented on the experimental robot. The spring and damping values had to be reduced due to electrical noise on the signal lines. Again, the torque and power requirements were measured for positional stepping and sinusoidal tracking in each degree of freedom. The maximum values were similar to those measured in simulation. The Series Elastic Actuators had no difficulty producing the necessary torque and power for standing. Hip torque is again under utilized during control of the z and ϕ directions.

7.2 Pronking Control

7.2.1 Pronking Control Algorithm

Like the high-level standing control, pronking control consists of two elements, a *state machine* and *control actions*. The state machine for pronking is much more complex than the single state needed to describe the standing algorithm. In fact, seven distinct states are used to describe the pronking algorithm. These include, *Aerial Fight, Single Support Touchdown - Leg, Single Support Touchdown - Arm, Double Support Touchdown, Double Support Liftoff, Single Support Liftoff - Leg, Single Support Liftoff - Arm.* A block diagram showing the flow of the state machine can be seen in figure 7-7. Notice that only certain states are accessible from other states. For example, If the robot is in Double Support Touchdown, it cannot get to Aerial Flight without passing through Double Support Liftoff and Single Support Liftoff. This prevents false state switching.

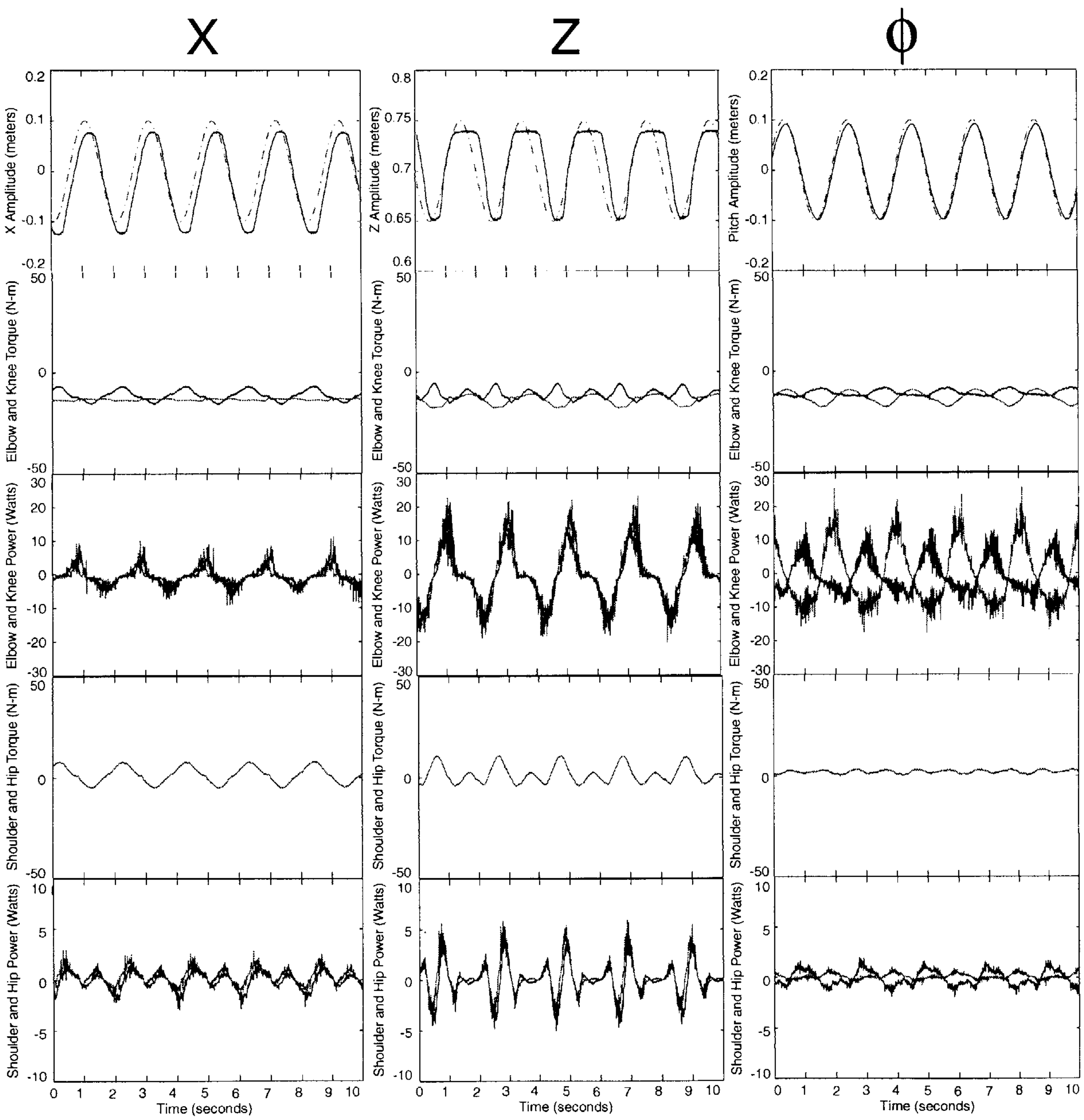

Figure 7-6: Left column shows experimental X frequency response at an amplitude of 0.05 meters and frequency of 0.5 hertz with $k_x = 100$, $b_x = 10$. Middle column shows experimental Z frequency response at an amplitude of 0.05 meters and frequency of 0.5 hertz with $k_z = 1000$, $b_z = 50$. Right column shows experimental pitch frequency response at an amplitude of 0.1 radians and frequency of 0.5 hertz with k_phi=100, $b_phi = 10$. Joint torques and associated joint power are also shown for the knee, elbow, hip and shoulder in each column.

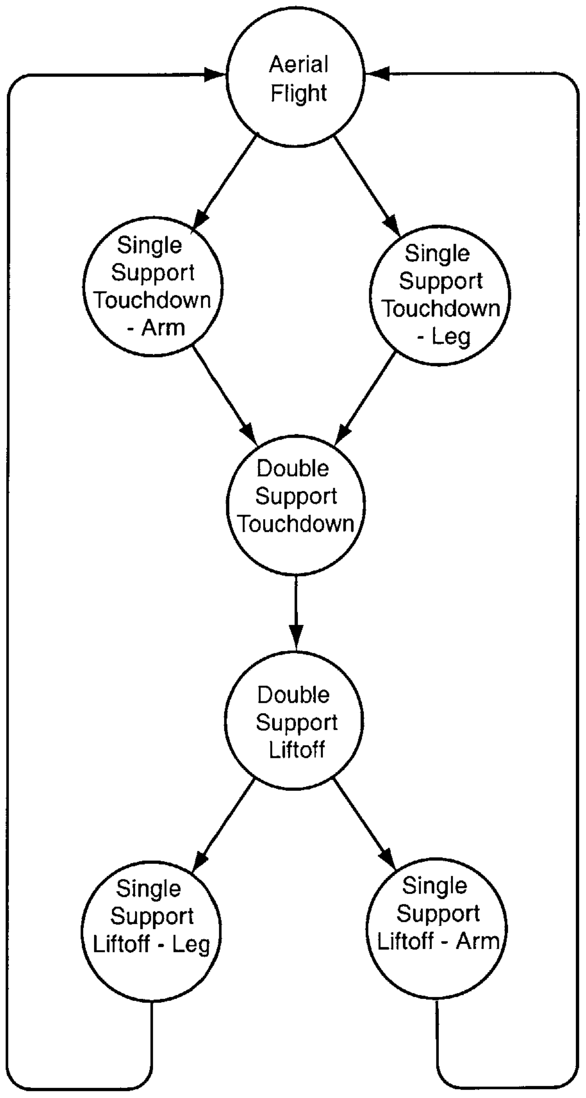
Aerial Flight
Single Support Touchdown - Arm
Single Support Touchdown - Leg
Double Support Touchdown
Double Support Liftoff
Single Support Liftoff - Leg
Single Support Liftoff - Arm

Figure 7-7: State machine used in pronking algorithm.

Table 7.5: State Machine with Sensor Values Indicated

State	Vertical Velocity	Foot Switch	Hand Switch	Previous State
Aerial Flight	+ or -	off	off	Single Support Liftoff - Arm Single Support Liftoff - Leg
Single Support Touchdown - Arm	-	off	on	Aerial Flight
Single Support Touchdown - Leg	-	on	off	Aerial Flight
Double Support Touchdown	-	on	on	Single Support Touchdown - Arm Single Support Touchdown - Leg
Double Support Liftoff	+	on	on	Double Support Touchdown
Single Support Liftoff - Arm	+	off	on	Double Support Liftoff
Single Support Liftoff - Leg	+	on	off	Double Support Liftoff

To determine the state of the robot, three inputs are required: the sign of the z velocity of the robot, V_z and ground contact information from the the foot and hand (referred to as foot and hand switches). If the switch is *on*, then the foot or hand is in contact with the ground. Table 7.5 shows each of the seven possible states and the condition of the corresponding inputs. Thus, by reading input from only three sensors, and knowing the previous state, we can determine the state of the robot.

With the state of the robot determined, the next step is to determine a control action to take while in that state. Below, we will examine each state and the control action to be taken.

Aerial Flight

Aerial Flight state is achieved when V_z is positive or negative and both the foot and the hand are *not* in contact with the ground. During Aerial Flight little can be done to control the x, z, or ϕ orientation of the robot. We simply position the feet for touchdown. First we set the desired length and angle for the arm and leg.

```
R_arm_d=0.750;
Alpha_arm_d=0.0;
R_leg_d=0.750;
Alpha_leg_d=0.0;
```

Where R is the distance between the foot and the hip or hand and knee and $Alpha$ is the angle R makes with vertical (see figure 6-5 of chapter 6). With the geometry of the robot and the desired R and $Alpha$ for the arm and leg, desired positions can be calculated for each joint.

```
q_d.shoulder=(Alpha_arm_d-acos((R_arm_d/2.0)/upper_arm_length)-q.phi);
q_d.elbow=2.0*acos((R_arm_d/2.0)/upper_arm_length);
q_d.hip=(Alpha_leg_d-acos((R_leg_d/2.0)/thigh_length)-q.phi);
q_d.knee=2.0*acos((R_leg_d/2.0)/thigh_length);
```

Finally, PD controllers are used at each joint to drive the joint to the desired position.

```
tau_shoulder=k_shoulder*(q.shoulder-q_d.shoulder)-b_shoulder*qd.shoulder;
tau_elbow=k_elbow*(q.elbow-q_d.elbow)-b_elbow*qd.elbow;
tau_hip=k_hip*(q.hip-q_d.hip)-b_hip*qd.hip;
tau_knee=k_knee*(q.knee-q_d.knee)-b_knee*qd.knee;
```

Note that the low-level Virtual Model Control was not used in this state. It is impossible to apply virtual forces when the foot and hand are *not* in contact with the ground.

Single Support Touchdown - Arm

Single Support Touchdown - Arm describes the state when the robot lands on its arm. V_z is negative, the foot switch is off, and the hand switch is on. During this state, the leg is controlled as it was during Aerial Flight. Again, a desired R and *Alpha* are used to calculate the desired hip and knee positions. PD control is used to servo the leg to the desired position.

```
R_leg_d=0.750;
Alpha_leg_d=0.0;
q_d.elbow=2.0*acos((R_arm_d/2.0)/upper_arm_length);
q_d.hip=(Alpha_leg_d-acos((R_leg_d/2.0)/thigh_length)-q.phi);
tau_hip=k_hip*(q.hip-q_d.hip)-b_hip*qd.hip;
tau_knee=k_knee*(q.knee-q_d.knee)-b_knee*qd.knee;
```

Since the arm is in contact with the ground, Virtual Model Control can be used. The Virtual Model Controller in section 6.2 of chapter 6 is used in this case, allowing for independent control of the arm and the leg. We want the arm to behave like a spring and damper. This will give the robot smooth support during touchdown, much the same way the suspension of a car provides the driver with a smooth ride over speed bumps. In this way, the equation for virtual force between the hand and the shoulder is formulated.

```
Fr_arm=k_arm*(R_arm_d-R_arm)-b_arm*dRdt_arm;
```

The length of the arm is set according to R desired.

```
R_arm_d=0.750;
```

The virtual force about the shoulder ($N_{\alpha_{arm}}$) is set to zero.

```
Nalpha_arm=0.0;
```

Finally, these virtual forces are mapped to actual joint torques using equation 6.26 of chapter 6.

```
tau_shoulder=(Nalpha_arm*A_arm+Fr_arm*C_arm);
tau_elbow=(Nalpha_arm*B_arm+Fr_arm*D_arm);
```

Single Support Touchdown - Leg

Single Support Touchdown - Leg describes the state when the robot lands on its leg. V_z is negative, the foot switch is on, and the hand switch is off. Thus, this state is precisely the same as Single Support Touchdown - Arm with the role of the arm and leg reversed.

Double Support Touchdown

Double Support Touchdown is the state when both feet are in contact with the ground and V_z is negative. Since both the foot and hand are on the ground Virtual Model Control can be used to position the robot. Again, the Virtual Model Controller developed in section 6.3 of chapter 6 gives a more intuitive look at the control of the robot. For this low-level controller, we must provide a virtual force F_R and N_α for the leg and arm.

It makes sense that F_R should behave like a spring and damper. Again, this will give the robot a virtual suspension, providing smooth support during touchdown.

Writing the virtual force equations for the arm and the leg yields

```
Fr_arm=k_arm*(R_arm_d-R_arm)-b_arm*dRdt_arm;
Fr_leg=k_leg*(R_leg_d-R_leg)-b_leg*dRdt_leg;
```

N_α is used position the robot horizontally. Instead of placing a virtual spring and damper at the hip and shoulder, we will solve for N_α such that the forces in the x direction sum to zero.

Recall, in chapter 6 we mapped the virtual forces F_R and N_α to forces on the robot along the x, z, and ϕ directions. Since we want the robot to pronk in place, that would suggest that f_x should be zero. If f_x is none zero there would be acceleration in the x direction. Setting $f_{x_{arm}}$ and $f_{x_{leg}}$ (equations 6.27 and 6.31, respectively) to zero and solving for $N_{\alpha_{arm}}$ and $N_{\alpha_{leg}}$ yields the balancing torques.

```
Nalpha_arm=-Fr_arm*R_arm*sin(Alpha_arm)/cos(Alpha_arm);
Nalpha_leg=-Fr_leg*R_leg*sin(Alpha_leg)/cos(Alpha_leg);
```

Finally, the desired virtual forces F_R and N_α are mapped to joint torques using equation 6.26 of chapter 6.

```
tau_hip=(Nalpha_leg*A_leg+Fr_leg*C_leg);
tau_knee=(Nalpha_leg*B_leg+Fr_leg*D_leg);
tau_shoulder=(Nalpha_arm*A_arm+Fr_arm*B_arm);
tau_elbow=(Nalpha_arm*C_arm+Fr_arm*D_arm);
```

Double Support Liftoff

Double Support Liftoff is very similar to Double Support Touchdown. Again, both feet are in contact with the ground the but V_z is positive. There are two primary differences.

1. A vertical thrust force must be applied to launch the robot into Aerial Flight.
2. A torque must be applied about the center of gravity to stablize pitch.

Both the thrust force and the pitch torque will be applied using F_R of the arm and leg. Thrust force is simple, just add a feed forward term large enough to the existing spring and damper to get the robot into Aerial Flight.

Pitch torque requires a differential change in $F_{R_{arm}}$ and $F_{R_{leg}}$ to create a couple to produce the desired torque. First, let us calculate the desired torque about the center of mass using a virtual spring and damper on pitch.

```
tau_phi=k_phi*(q_d_phi-q_phi)-b_phi*qd_phi;
```

Converting the desired torque to a force acting a distance of L_3 from the center of mass

```
F_tau_phi=tau_phi/(body_length/2);
```

Now, simply add half of this force to $F_{R_{arm}}$ and subtract half of the force from $F_{R_{leg}}$. In this way, a positive τ_ϕ will will decrease $F_{R_{arm}}$ and increase $F_{R_{leg}}$, resulting in a positive torque about the center of gravity. Adding the feed forward term and the torque term to F_R yields

```
Fr_arm=k_arm*(R_arm_d-R_arm)-b_arm*dRdt_arm+Fz_ff-F_tau_phi;
Fr_leg=k_leg*(R_leg_d-R_leg)-b_leg*dRdt_leg+Fz_ff+F_tau_phi;
```

N_α is used position the robot horizontally. Again, setting $f_{x_{arm}}$ and $f_{x_{leg}}$ (equations 6.27 and 6.31, respectively) to zero and solving for $N_{\alpha_{arm}}$ and $N_{\alpha_{leg}}$ yields the balancing torques.

```
Nalpha_arm=-Fr_arm*R_arm*sin(Alpha_arm)/cos(Alpha_arm);
Nalpha_leg=-Fr_leg*R_leg*sin(Alpha_leg)/cos(Alpha_leg);
```

Finally, the desired virtual forces F_R and N_α are mapped to joint torques using equation 6.26 of chapter 6.

```
tau_hip=(Nalpha_leg*A_leg+Fr_leg*C_leg);
tau_knee=(Nalpha_leg*B_leg+Fr_leg*D_leg);
tau_shoulder=(Nalpha_arm*A_arm+Fr_arm*C_arm);
tau_elbow=(Nalpha_arm*B_arm+Fr_arm*D_arm);
```

Single Support Liftoff - Arm

Single Support Liftoff - Arm is a variation of the Single Support Touchdown - Arm. The difference is that a feed forward thrusting must occur during Liftoff but not during Touchdown. During this state, the leg is controlled as it was during Aerial Flight. Again, a desired R and $Alpha$ are used to calculate the desired hip and knee positions. PD control is used to servo the leg to the desired position.

```
R_leg_d=0.750;
Alpha_leg_d=0.0;
q_d.elbow=2.0*acos((R_arm_d/2.0)/upper_arm_length);
q_d.hip=(Alpha_leg_d-acos((R_leg_d/2.0)/thigh_length)-q.phi);
tau_hip=k_hip*(q.hip-q_d.hip)-b_hip*qd.hip;
tau_knee=k_knee*(q.knee-q_d.knee)-b_knee*qd.knee;
```

The arm, which is still in contact with the ground continues to thrust vertically with a feed forward term on top of a spring and damper. Differential F_R can no longer provide a couple to the body to control pitch, thus, τ_ϕ is dropped from $F_{R_{arm}}$.

```
Fr_arm=k_arm*(R_arm_d-R_arm)-b_arm*dRdt_arm+Fz_ff;
```

N_α is used position the robot horizontally. Again, setting $f_{x_{arm}}$ to zero and solving for $N_{\alpha_{arm}}$ yields the balancing torque.

```
Nalpha_arm=-Fr_arm*R_arm*sin(Alpha_arm)/cos(Alpha_arm);
Nalpha_leg=-Fr_leg*R_leg*sin(Alpha_leg)/cos(Alpha_leg);
```

Finally, the desired virtual forces F_R and N_α are mapped to joint torques using equation 6.26 of chapter 6.

```
tau_shoulder=(Nalpha_arm*dAdH_arm+Fr_arm*dRdH_arm);
tau_elbow=(Nalpha_arm*dAdK_arm+Fr_arm*dRdK_arm);
```

Single Support Liftoff - Leg

Single Support Liftoff - Leg describes the state when the robot's hand loses contact with the ground and the leg thrusts vertically. Thus, this state is precisely the same as Single Support Touchdown - Arm with the role of the arm and leg reversed.

The pronking control algorithm is summarized in table 7.6.

7.2.2 Simulated Pronking Control

Using this algorithm, and the associated low-level controller, the robot was made to pronk in simulation. The simulated robot possess realistic mass and inertia but has unlimited power capabilities. A torque limit of 50 N-m is placed on each of the joints. The simulated robot can maintain stable pronking for an indefinite period of time. Figure 7-8 shows the x, z, and ϕ position of the robot over five pronking cycles. It can be seen that the robot pronks at a height of approximately 0.1 meters, while maintaining reasonably good horizontal position and body pitch.

Before examining torque and power requirements, let us first look at a single pronking cycle in detail. Figure 7-9 shows the virtual forces exerted on the robot during a single pronking cycle. During Aerial Flight no virtual forces are applied to the robot. Recall, the joints are simply positioning the leg and arm for touchdown. The next state is Single Support Touchdown - Arm. $F_{R_{arm}}$ begins to apply forces to the robot in this state, though barely detectable. The remaining virtual forces are still zero. After a few milliseconds, both the foot and the arm come into contact with the ground and the robot enters into Double Support Touchdown. During Double Support Touchdown, the arm and leg apply a virtual spring and damper forces to the body. These forces grow larger as the

Table 7.6: State Machine and Associated Control Actions

State	Control Actions
Aerial Flight	Position Control Arm to Desired R and *Alpha* Position Control Leg to Desired R and *Alpha*
Single Support Touchdown - Arm	Apply Virtual Force to Arm to Simulate Spring and Damper Position Control Leg to Desired R and *Alpha*
Single Support Touchdown - Leg	Position Control Arm to Desired R and *Alpha* Apply Virtual Force to Leg to Simulate Spring and Damper
Double Support Touchdown	Apply Virtual Force to Arm to Simulate Spring and Damper Apply Virtual Force to Leg to Simulate Spring and Damper Balance Horizontal Forces with $N_{\alpha_{arm}}$ and $N_{\alpha_{leg}}$
Double Support Liftoff	Apply Virtual Force to Arm to Simulate Spring and Damper - add a feed forward term for vertical thrust - subtract a term to provide differential thrust to control pitch Apply Virtual Force to Leg to Simulate Spring and Damper - add a feed forward term for vertical thrust - add a term to provide differential thrust to control pitch Balance Horizontal Forces with $N_{\alpha_{arm}}$ and $N_{\alpha_{leg}}$
Single Support Liftoff - Arm	Apply Virtual Force to Arm to Simulate Spring and Damper - add a feed forward term for vertical thrust Balance Horizontal Forces with $N_{\alpha_{arm}}$ Position Control Leg to Desired R and *Alpha*
Single Support Liftoff - Leg	Position Control Arm to Desired R and *Alpha* Apply Virtual Force to Leg to Simulate Spring and Damper - add a feed forward term for vertical thrust Balance Horizontal Forces with $N_{\alpha_{leg}}$

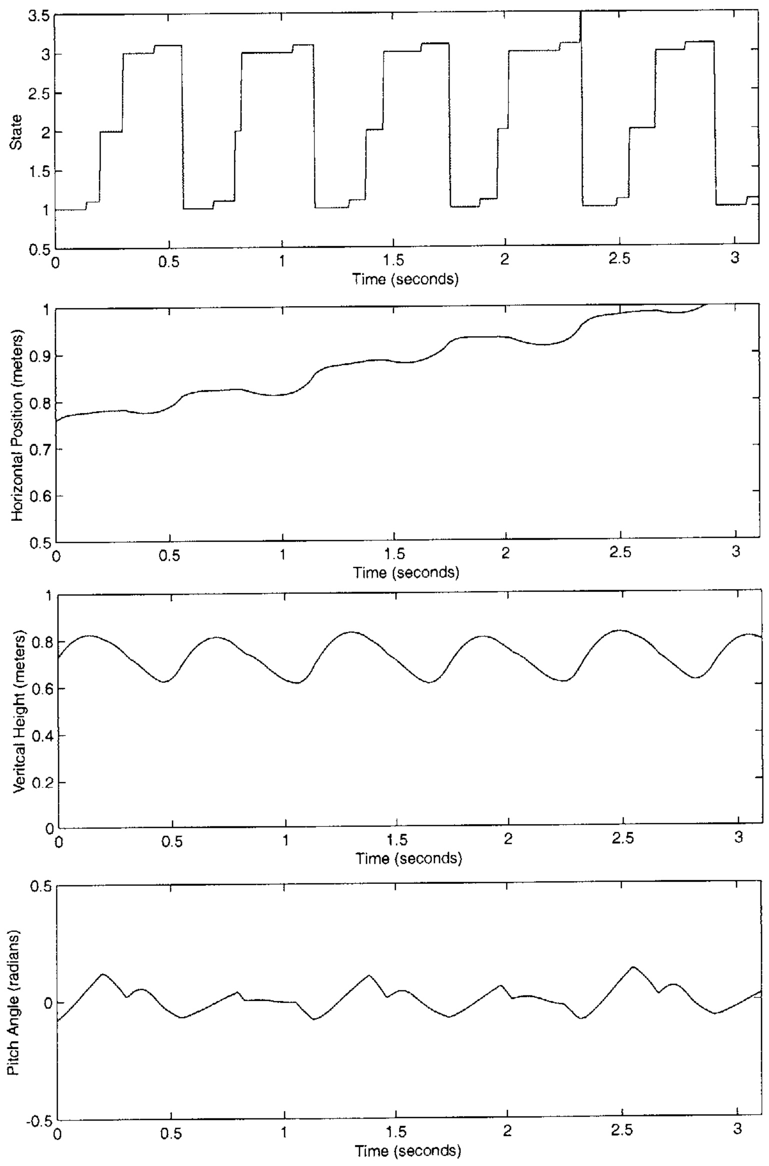

Figure 7-8: x, z, and ϕ position of simulated robot over five pronking cycles.

time in Double Support lengthens, because the virtual springs are compressing. At the same time, $N_{\alpha_{arm}}$ and $N_{\alpha_{leg}}$ are applied such that the net horizontal force on the robot is zero. ($N_{\alpha_{arm}}$ and $N_{\alpha_{leg}}$ are very small because $F_{R_{arm}}$ and $F_{R_{leg}}$ are also small.) Eventually, the virtual spring and damper reverse the vertical velocity of the robot. At the moment the velocity becomes positive, the robot enters Double Support Liftoff. Two adjustments are made to the controller at this time. First, a feed forward force is applied on top of the virtual spring and damper to give the robot lift. This is immediately visible by the sharp spike in $F_{R_{arm}}$ and $F_{R_{leg}}$ at the cross over from Double Support Touchdown to Double Support Liftoff. Second, a torque is applied to the robot to obtain the desired pitch. The desired torque can be seen in the figure labeled τ_{ϕ} directly below the state information. Recall, the torque is applied to the robot by differentially thrusting $F_{R_{arm}}$ and $F_{R_{leg}}$. The differential nature of $F_{R_{arm}}$ and $F_{R_{leg}}$ is noticeable During Double Support Liftoff. Finally, the thrust lifts the foot and hand from the ground and the robot is again in Aerial Flight.

With a single pronking cycle well understood, we can now look at the torque and power requirements. Figure 7-10 shows the torque requirements for each joint during the course of five pronking cycles. From this data, we can see that the maximum torque of 50 N-m is reached often in the elbow. This is because the arm must provide a larger F_R than the leg to control the pitch of the robot. The knee joint also frequently reaches its maximum torque limit. This is expected since most of the vertical thrust comes from the knee and elbow joints. The shoulder and hip joints never reach their maximum torque. In fact, they are significantly below there maximum torque most of the time. This would suggest that the hip and shoulder could be used to provide additional vertical thrust for greater hopping height or horizontal thrust for forward speed.

Power was recorded for the same five hopping cycles and can be seen in figure 7-11. It is not surprising that the elbow consumes the most power of any joint. Again, it provides vertical thrust and accounts for most of the torque needed to position the pitch of the body. The shoulder and knee consume moderate power, while the hip appears to be under utilized. In the experimental robot, power is limited to approximately 500 watts by the actuators. Still, the simulation provides a good starting point for implementation on the experimental robot.

7.2.3 Experimental Pronking Control

Using the same algorithm, and the associated low-level controller, the experimental robot was made to pronk in the laboratory. A physical torque limit of 50 N-m exists on each of the joints. Still, the experimental robot can maintain stable pronking for several minutes, after which, the motors overheat. Figure 7-12 shows the x, z, and ϕ position of the robot over five pronking cycles. It can be seen that the robot pronks at a height of approximately 0.1 meters, while maintaining reasonably good horizontal position and body pitch.

The torque and power requirements for pronking were recorded over the same five hopping cycles and can be seen in figures 7-13 and 7-14, respectively. Again, it appears that the elbow and knee are at or near their maximum torque much of the time. The shoulder and hip appear to play secondary roles in the pronking algorithm. Figure 7-14 shows that most of the power is consumed in the elbow and knee of the robot while the shoulder and hip operate well below their maximum power capabilities. This suggest that the hip and shoulder could be incorporated in the algorithm to assist vertical or horizontal thrust.

7.2.4 Summary of Pronking Control

A state machine and control algorithm for pronking has been developed. The algorithm was simulated on a robot with physical properties similar to the experimental robot. The robot successfully pronked in simulation but required power above the capabilities of the actual robot. This left some question as to whether the experimental robot would successfully pronk without counter balancing or other extraordinary measures.

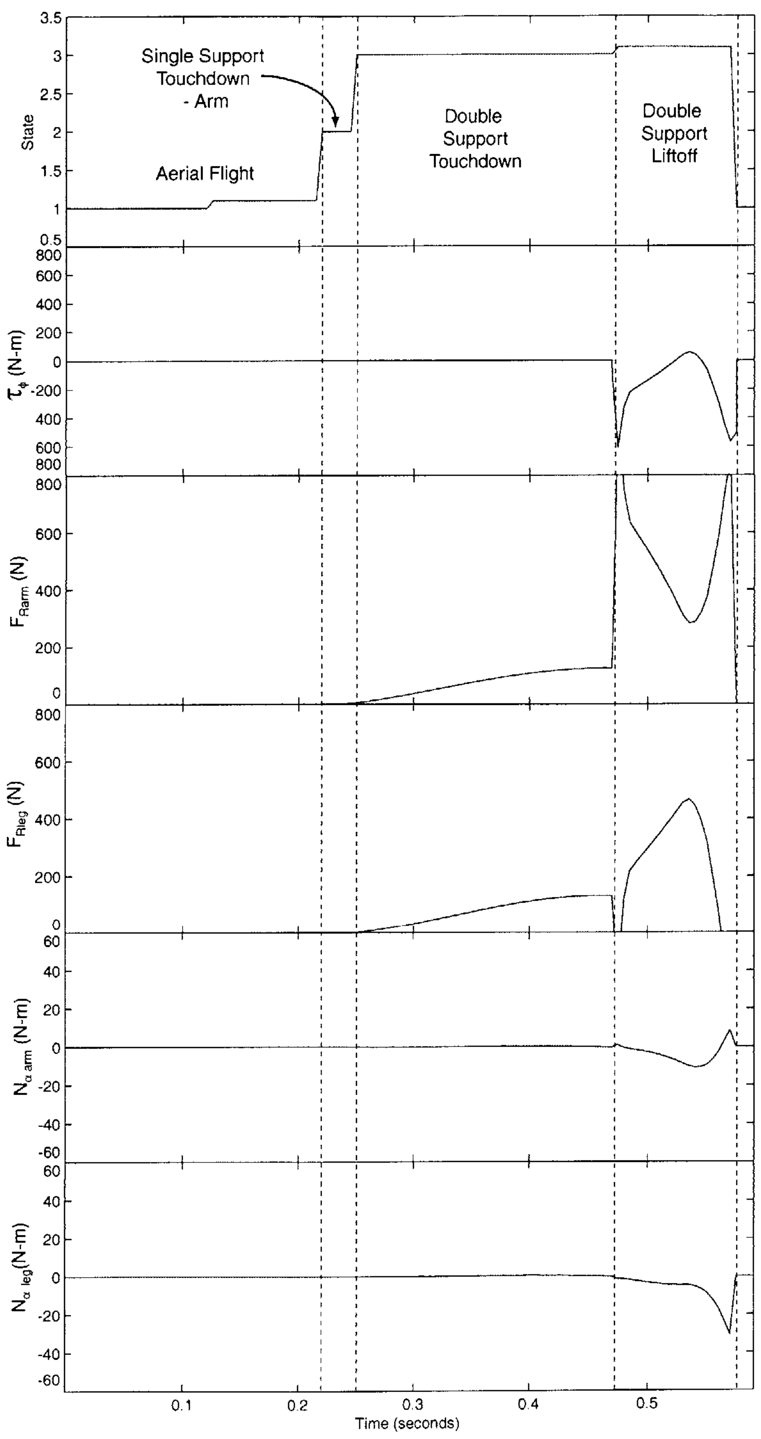

Figure 7-9: Data from single pronking cycle for simulated robot.

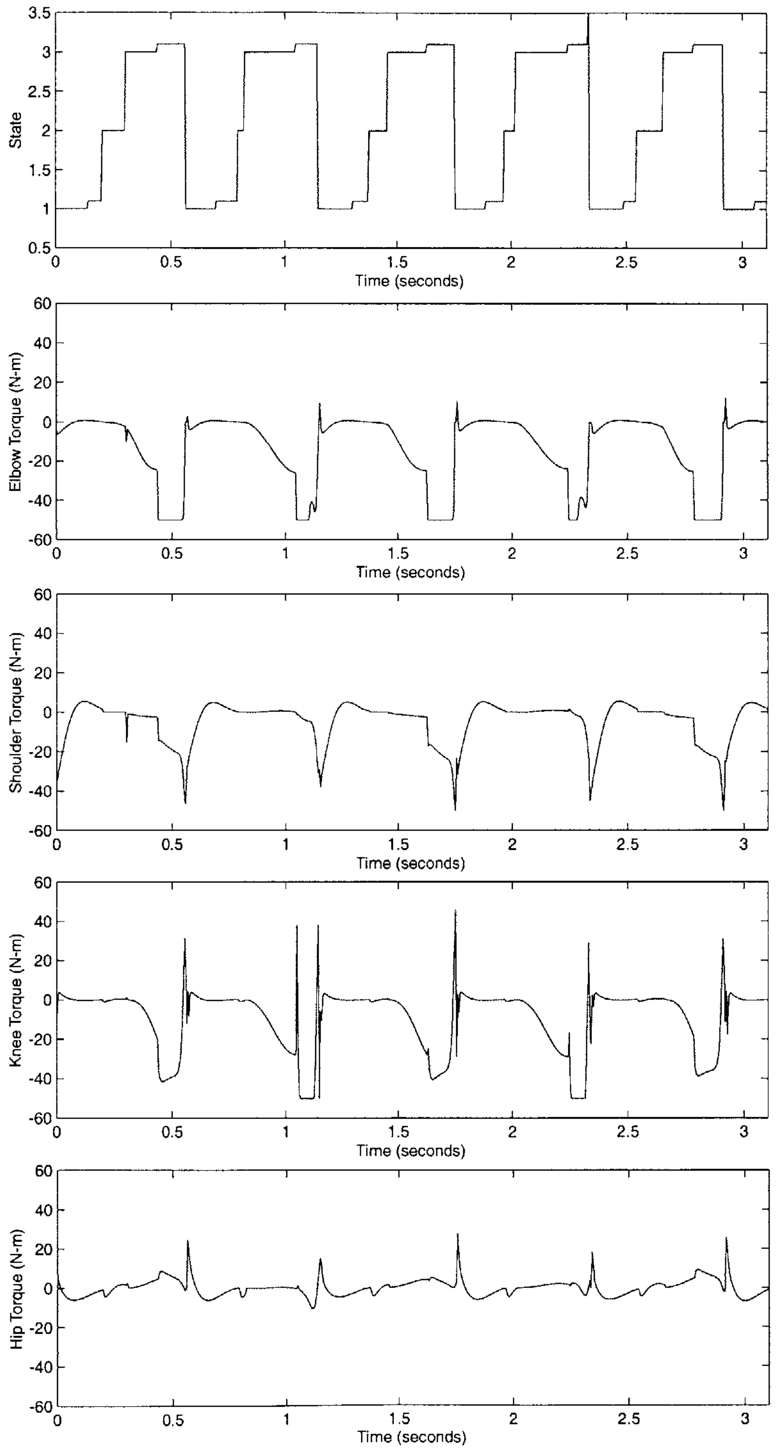

Figure 7-10: Torque data over five pronking cycles for simulated robot.

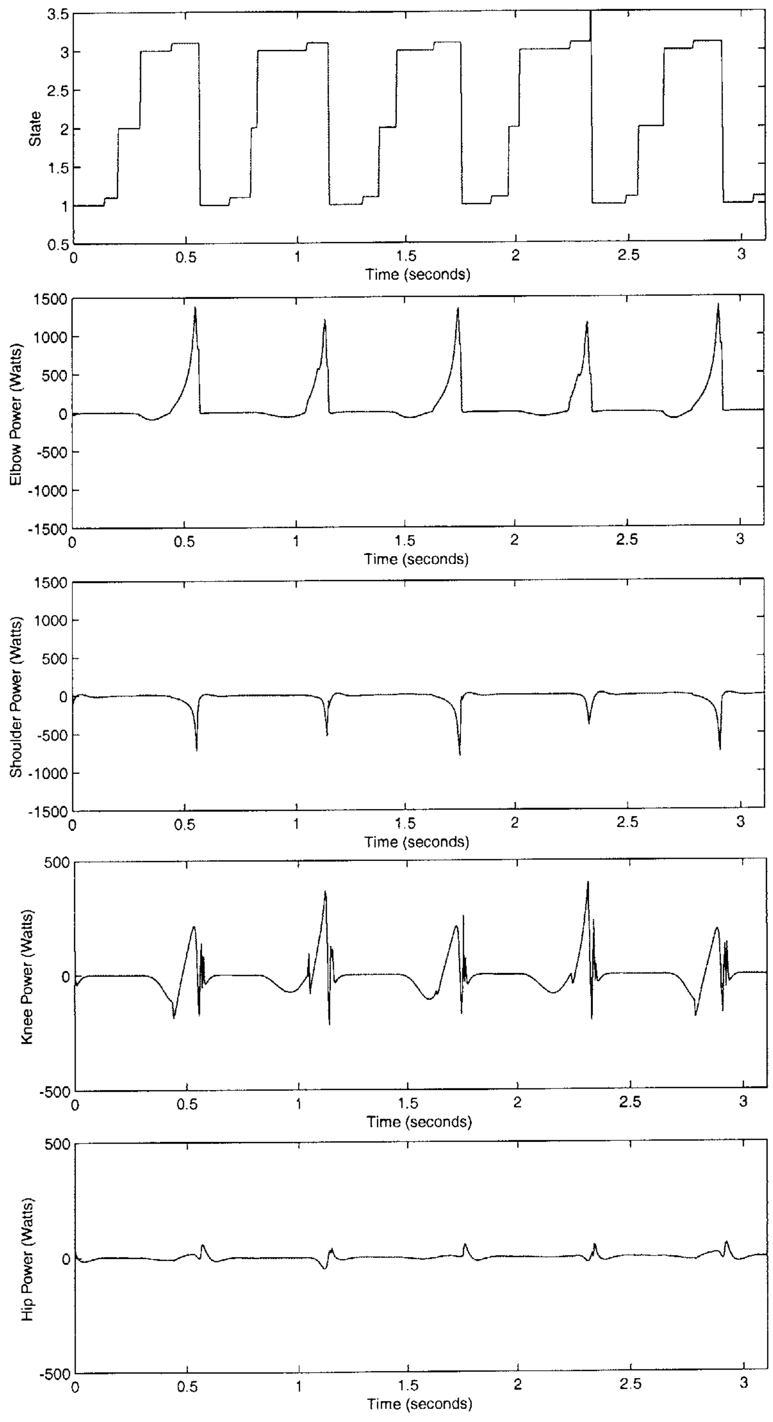

Figure 7-11: Power data over five pronking cycles for simulated robot.

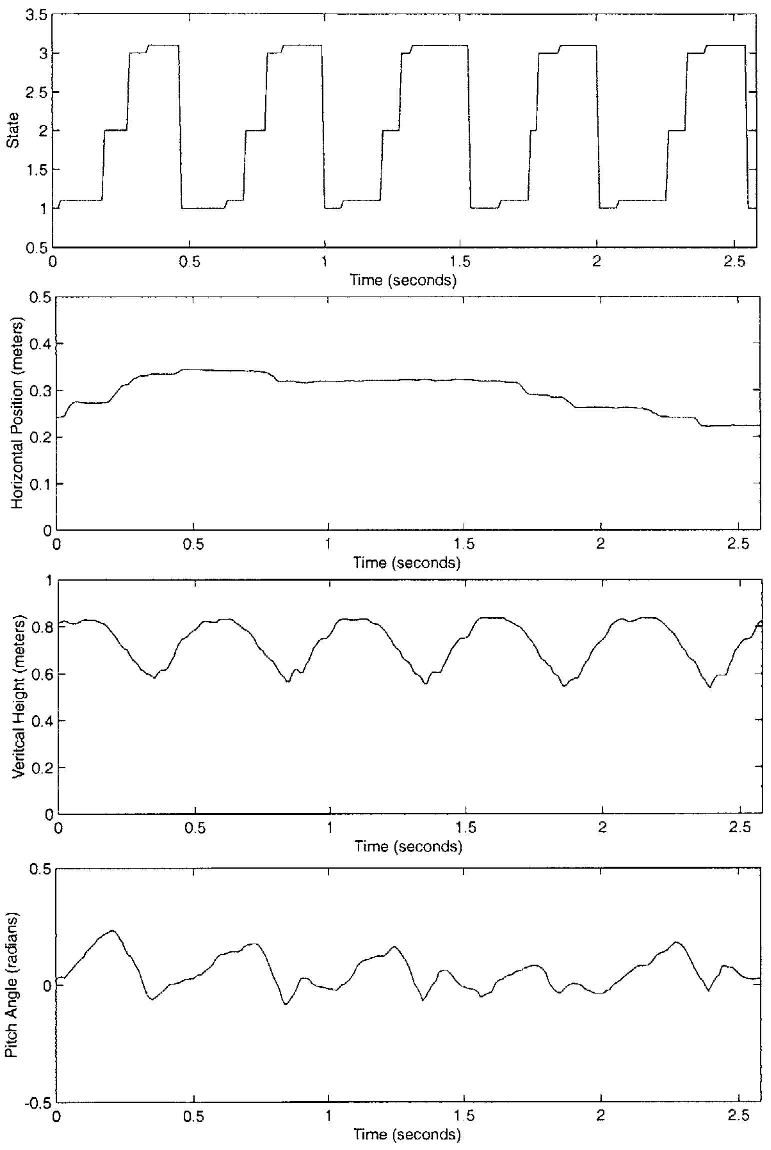

Figure 7-12: x, z, and ϕ position of experimental robot over five pronking cycles.

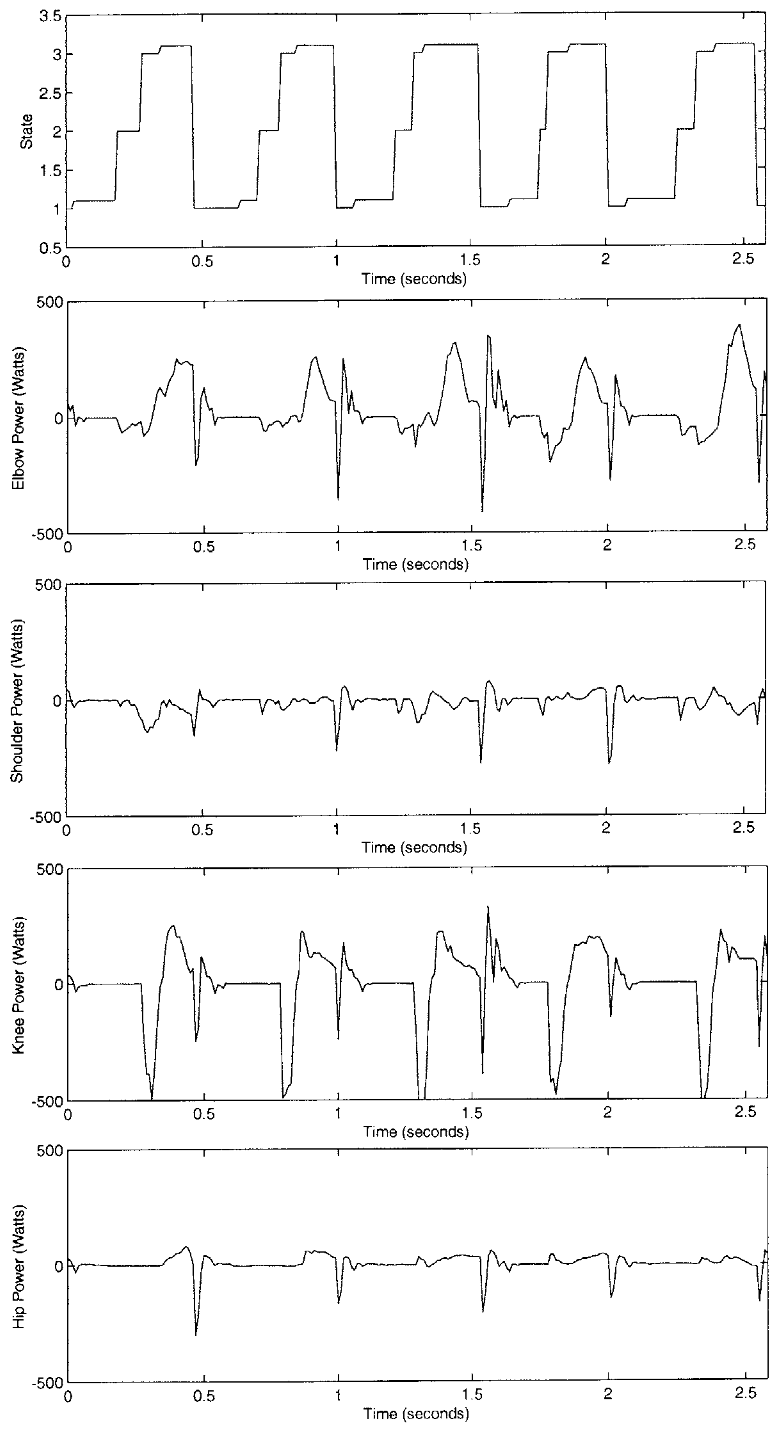

Figure 7-13: Power data over five pronking cycles for experimental robot.

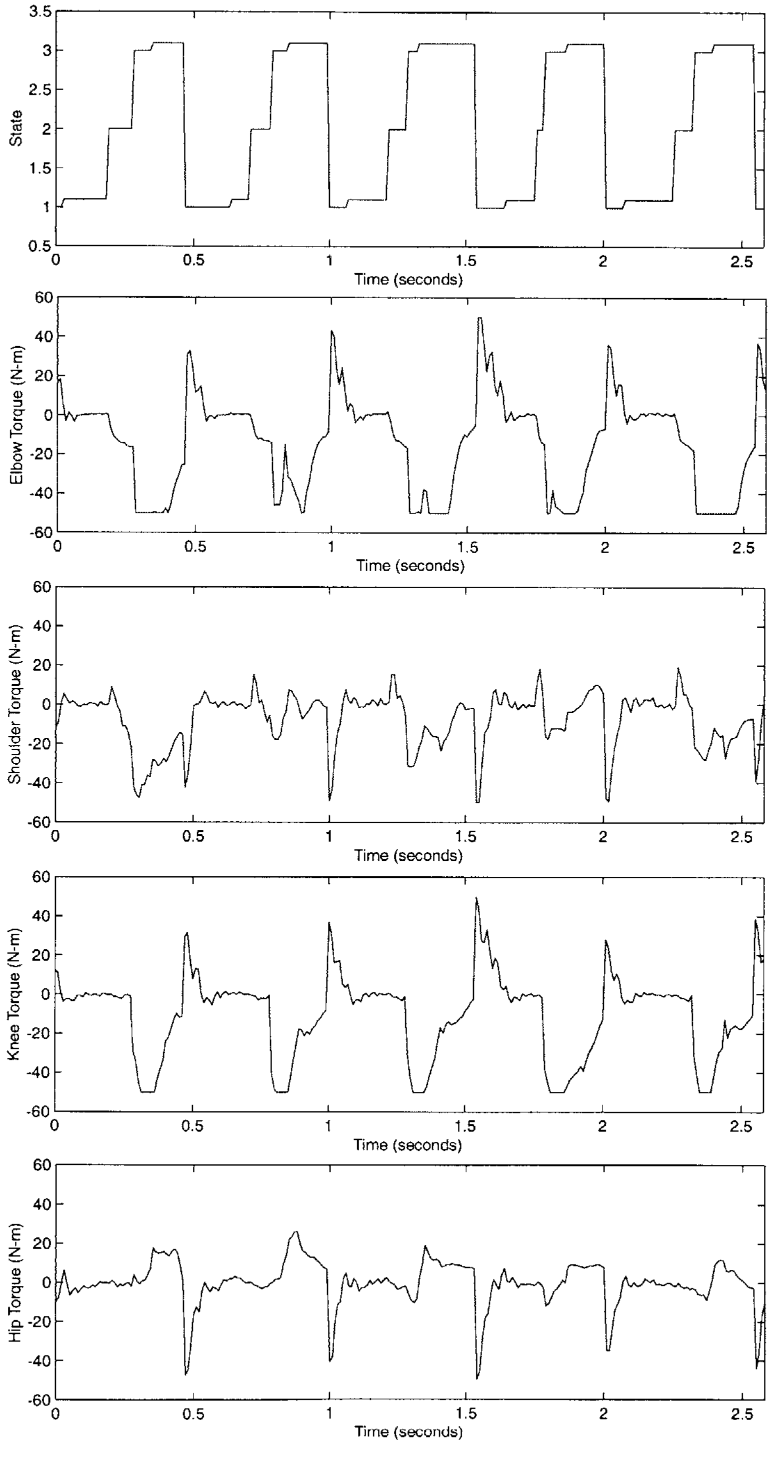

Figure 7-14: Torque data over five pronking cycles for experimental robot.

The same algorithm was implemented on the experimental robot with similar success. Peak power requirements were approximately 500 watts at the knee and elbow joint and 250 watts at the hip and shoulder joints. All joints reached the maximum torque output of the actuator of 50 N-m.. The data suggest that the hip and shoulder joints are under utilized in the pronking algorithm. This power could be put to use making the robot trot.

Chapter 8

Conclusions

In chapter 1 we set forth four goals for this thesis:

1. To investigate the use of springs in the feet for smooth and efficient means of energy transfer.
2. To investigate the use of natural dynamics in the swing leg of the robot to achieve high efficiency.
3. To develop intuitive control algorithms which can easily be scaled to a three dimensional quadrupedal robot.
4. To verify that it is possible to meet the power requirements for a running robot using electric motors.

Each of these has been addressed in previous chapters. Now, we summarize our findings and make recommendations for future work.

8.1 The Use of Springs

Conclusions:

In chapter 3 we showed that springy feet can store and release significant amounts of energy during running. The stiffness of the spring had to be carefully selected based on tradeoffs between efficiency and controllability. It was found that a soft foot spring gives good efficiency and controllability characteristics but limits the maximum speed of the robot. Stiffer springs can also be efficient *if* the virtual spring created by the knee actuator is *significantly* stiffer than the foot spring. Such a situation is difficult to achieve because of the power limitations of electric motors. In conclusion, springs are imperative for high efficiency, but their efficiency is severly limited when placed in series with knee actuators.

Future Work:

An excellent area for future work is the development of a spring which would be capable of storing energy during touchdown, independent of the knee stiffness. This can be achieved by placing the spring in parallel with the knee actuator, rather than the series configuration of chapter 3.

8.2 The Use of Natural Dynamics

Conclusions:

In chapter 4 we suggested that natural dynamics can improve the efficiency of running robots. After conducting experiments on an experimental robot, it is our feeling that the potential cost savings of natural dynamics for running is minimal. We can see how this could be beneficial to a robot at walking speeds, where high speed control is not necessary. However, because of the high frequency of the swing leg, active control must constantly be applied to get the desired behavior. Still, we found the knee and hip stop to be extremely helpful in controlling high speed movements of the thigh and shin with no overshoot.

Future Work:

There is signicant natural dynamic interaction between the body and legs of the robot. It would be interesting to learn if it is possible to control the pitch of the body during aerial flight using the inertia of the legs without interrupting the placement of the feet.

8.3 The Development of Control Algorithms

Conclusions:

In chapters 6 and 7 we developed an intuitive control scheme based on Virtual Model Control. Using Virtual Model Control, virtual forces can be applied to the robot in the same way a puppeteer pulls the strings of a marionette puppet. The control algorithms were successfully implemented in simulation and on the experimental robot.

Future Work:

It is easy to imagine extending the pronking control algorithm into a trotting algorithm by applying a virtual horizontal force while on the ground. The virtual horizontal force could be applied by sweeping the leg and arm rearward during stance. The difficulty here will be stabilization of the pitch in the aerial phase. That is to say, the leg and arm will significantly effect the pitch of the robot as they swing forward during aerial flight. To reduce this effect, the inertia of the body can be increased or the inertia of the arm and leg should be decreased.

8.4 Meeting the Power Requirements

Conclusions:

In Chapter 7 we showed that it is indeed possible to meet the power requirements of pronking with electric motors. The peak power requirement of each joint were less than 500 Watts, while the maximum torque did not exceeded 50 N-m. We can hypothesize that the peak power requirements for running will be no greater than those of pronking because the largest power requirement will come from just getting the robot airborne.

Future Work:

From the power data, it was easy to see that the hip and shoulder joints tended to be underutilized. It would be extremely useful to develop a control algorithm which used the hip and shoulder for vertical thrust. In this way, it might be possible to keep the same average power, but reduce the peak power of the knee and elbow joints.

Bibliography

Adolfsson, J., Dankowicz, H. & Nordmark, A. (1998), '3-d Stable Gait in Passive Bipedal Mechanisms', *Proceedings of 357 Euromech.*

Alexander, R. (1990), 'Three Uses for Springs in Legged Locomotion', *International J. Robotics Research* **9**(2), 53–61.

Buehler, M. (1996), 'Design, Control, and Energetics of an Electrically Actuated Legged Robot', *IEEE Transactions on Systems, Man, and Cybernetics.*

Craig, J. J. (1989), *Introduction to Robotics: Mechanics and Control*, Addison-Wesley.

Farley, C. T., Glasheen, J. & McMahon, T. A. (1993), 'Running Springs: Speed and Animal Size', *ASME Journal of Experimental Biology* **185**, 71–86.

Fowble, J. & Kuo, A. (1996), 'Stability and Control of Passive Locomotion in 3D', *Proceedings of the Conference on Biomechanics and Neural Control of Movement* pp. 28–29.

Garcia, M., Chatterjee, A. & Ruina, A. (1998), 'Speed, efficiency, and stability of small-slope 2D passive dynamic bipedal walking', *IEEE International Conference on Robotics and Automation* pp. 2351–2356.

Heglund, N. C. & Taylor, C. R. (1988), 'Speed, Stride Frequency, and Energy Cost Per Stride: How Do They Change With Body Size and Gait?', *Journal of Experimental Biology* **138**, 301–318.

Herr, H. M. (1998), A Model of a Mammalian Quadrupedal Running, PhD thesis, Harvard University.

Hirai, K., Hirose, M., Haikawa, Y. & Takenaka, T. (1998), 'The Development of Honda Humanoid Robot', *IEEE International Conference on Robotics and Automation.*

Hodgins, J. (1987), 'Biped Gynastics', *International Journal of Robotics Research.*

ichi Yamaguchi, J., Takanishi, A. & Kato, I. (1993), 'Development of a Biped Walking Robot Compensating For Three-Axis Moment By Trunk Motion', *IEEE International Conference on Intelligent Robots and Systems* pp. 561–566.

Leeser, K. F. (1996), Locomotion Experiments on a Planar Quadruped Robot with Articulated Spine, Master's thesis, Massachusetts Institute of Technology.

Margaria, R. (1976), *Biomechanics and Energetics of Muscular Activity*, Claredon Press.

McGeer, T. (1990), 'Passive Dynamic Walking', *International Journal of Robotics Research* **9**(2), 62–82.

McMahon, T. A. (1984), 'Mechanics of Locomotion', *The International Journal of Robotics Reasearch* **3**(2), 4–28.

Miura, H. & Shimoyama, I. (1984), 'Dynamic Walk of a Biped', *International Journal of Robotics Research* **3**(2), 60–74.

Muybridge, E. (1957), *Animals in Motion*, Dover Publications.

Pandy, M. G., Kumar, V., Berme, N. & Waldron, K. J. (1988), 'The Dynamics of Quadrupedal Locomotion', *ASME Journal of Biomechanical Engineering* **110**, 230–237.

Playter, R. (1995), 'Passive Dynamics in the Control of Gymnastic Maneuvers', *A.I. Memo No. 1504, MIT AI Lab.*

Pratt, G. A. & Williamson, M. M. (1995), 'Series Elastic Actuators', *IEEE International Conference on Intelligent Robots and Systems* **1**, 399–406.

Pratt, J. E. (1995), Virtual Model Control of a Biped Walking Robot, Master's thesis, Massachusetts Institute of Technology.

Pratt, J. E. & Pratt, G. A. (1999), 'Exploiting Natural Dynamics in the Control of a 3D Bipedal Walking Simulation', *Proceedings of the International Conference on Climbing and Walking Robots (CLAWAR99).*

Raibert., M. H. (1986), *Legged Robots That Balance.*, MIT Press, Cambridge, MA.

Robinson, D. W., Pratt, J. E., Paluska, D. J. & Pratt, G. A. (1999), 'Series Elastic Actuator Development for a Biomimetic Robot', *IEEE/ASME International Conference on Advanced Intelligent Mechatronics.*

Williamson, M. M. (1995), Series Elastic Actuators, Master's thesis, Massachusetts Institute of Technology.