Proceedings of the 1998 International Conference on Robotics and Automation (ICRA '98), Belgium

Intuitive Control of a Planar Bipedal Walking Robot

Jerry Pratt, Gill Pratt
MIT Leg Laboratory, Cambridge, MA 02139
http://www.leglab.ai.mit.edu/

Abstract

Bipedal robots are difficult to analyze mathematically. However, successful control strategies can be discovered using simple physical intuition and can be described in simple terms.

Five things have to happen for a planar bipedal robot to walk. Height has to be stabilized. Pitch has to be stabilized. Speed has to be stabilized. The swing leg has to move so that the feet are in locations which allow for the stability of height, pitch, and speed. Finally, transitions from support leg to support leg must occur at appropriate times. If these five objectives are achieved, the robot will walk.

A number of different intuitive control strategies can be used to achieve each of these five objectives. Further, each strategy can be implemented in a variety of ways. We present several strategies for each objective which we have implemented on a bipedal walking robot.

Using these simple intuitive strategies, we have compelled a seven link planar bipedal robot, called Spring Flamingo, to walk. The robot walks both slowly and quickly, walks over moderate obstacles, starts, and stops. Video, photographs, and more information on Spring Flamingo can be found at http://www.leglab.ai.mit.edu/

1 Introduction

Walking is a moderately easy task, and complex control techniques are not necessary to compel bipedal robots to walk. Instead, simple control strategies which can be explained in simple terms can be used.

Five things have to happen for a planar bipedal robot to walk. Height has to be stabilized. Pitch has to be stabilized. Speed has to be stabilized. The swing leg has to move so that the feet are in locations which allow for the stability of height, pitch, and speed. Finally, transitions from support leg to support leg must occur at appropriate instances. If these five objectives are achieved, then the robot will walk.

Any controller which results in stable walking must meet these five objectives. No matter what control technique is used, something must be happening which stabilizes height, pitch, and speed, swings the swing leg, and transitions from double to single and single to double support.

In this paper we describe intuitive control strategies which can be used to meet these five objectives and hence compel a robot to walk. We describe how these control strategies can be implemented with Virtual Model Control [12], a robot control language which itself appeals to intuition.

We introduce a simple technique, called the "virtual toe point" constraint for dealing with feet and actuated ankles. The virtual toe point is the point along the foot at which zero torques are commanded. This is similar to a static version of the zero moment point [16]. The virtual toe point constraint prevents the foot from over-rotating due to high ankle torques, while allowing the robot to go up on its toes in order to get an extended range of motion from the rear leg. The ability of the robot to go up on its toes allows it to walk with a straight leg which in turn increases energy efficiency.

We also present a simple scheme for dealing with force distribution when in the double support phase of walking. Instead of solving the force distribution problem exactly, we use an approximate force distribution parameter which can then be modified for speed control.

Using simple intuitive control strategies, the virtual toe point constraint, and the simple force distribution scheme, we have compelled a seven link planar bipedal robot to walk. The robot walks both slowly and quickly, walks over moderate obstacles, starts, and stops.

1.1 Intuitive Control

An intuitive controller is one which is based on human intuition of the system and an idea of what is going on. Descriptions of such controllers sound like something a coach would tell a player, like "Square up to the bucket, feet shoulder width apart, bend the knees, pretend you're shooting out of a telephone booth, and follow through." A robotic example might be to put a peg in a hole, "Push the peg and block together, slide the peg until part of it falls in the hole, line it up better, and wiggle it around until it is in."

Intuitive control is nothing new. A PD (Proportional-Derivative) controller is often described as a controller which pushes in the direction of the error and pushes back with increasing velocity to take some energy out and prevent the system from going too fast. Add the I (Integral) term and the controller keeps pushing harder and harder in the direction of the error until it finally goes away.

Raibert's hoppers use a simple 3-part intuitive controller [14]. The height, balance, and speed are stabilized using simple intuitive control laws which can be described in simple terms: to control height, energy is pumped into the leg spring when the leg is fully compressed; to control pitch, the body is servoed to be level to the ground when the stance leg is compressed; to control speed over a stride, the foot is placed further forward (to slow down) or further back (to speed up) from the neutral point in which speed is neither increased or decreased.

Intuitive controllers can be powerful, they are easy to apply, and by default they provide a high level of insight as to what is going on with the system and what is really important in the control of a robot. Unfortunately, intuitive controllers are often called "ad-hoc" or a "hack" because they are not mathematically based, and seldom rigorously proven to work. They are often considered to be "cheating" because parameters are usually hand tuned until desired performance is achieved.

However, using an intuitive controller does not preclude the use of mathematical analysis; if any controller for a system can be analyzed for stability, an intuitive one may be. Unfortunately, legged robots are complicated to analyze for several reasons which are independent of getting a legged robot to perform a task. Therefore, we believe that the limited capability of modern control theory and analysis should not be used as an excuse for the limited abilities of modern robots.

It is possible to use adaptive or learning techniques to automatically tune intuitive control parameters. When the adaptation or learning is complete it may then be possible to understand what was learned or why by looking at parameters whose effects can be understood. For example, suppose the single support to double support transition distance parameter is changed by a learning algorithm that is attempting to maximize efficiency. One may then draw insight into how that parameter affects efficiency.

In the following section we describe intuitive strategies that can be used to control bipedal walking robots. These strategies are easy to understand and easy to apply. Some of the strategies have been used to successfully compel Spring Flamingo, a planar bipedal robot, to walk.

2 Simple Intuitive Control Strategies for Bipedal Walking

We now describe simple intuitive control strategies which can be used to achieve the five objectives required for planar bipedal walking. These objectives are height stabilization, pitch stabilization, speed stabilization, swing leg placement, and support transitions.

There are a number of methods which can be used to implement the intuitive control strategies. These include inverse kinematics, high gain servos, feedforward control, impedance control, etc.

We use Virtual Model Control, a method which itself relies on intuition, to implement the control strategies. This control technique uses simulations of virtual mechanical components to generate real actuator torques (or forces). These joint torques create the same effect that the virtual components would have created, had they existed, thereby creating the illusion that the simulated components are connected to the real robot. Such components can include linear or non-linear springs, dampers, dashpots, masses, latches, bearings, potential and dissipative fields, or any other imaginable component.

In section 4 we discuss how the following control strategies are used to compel Spring Flamingo, a bipedal walking robot, to walk.

2.1 Height Stabilization

Stabilizing height is straightforward as long as we have a support leg firmly on the ground. This will be the case if the swing leg strategy and support leg strategies are working. There are many ways height can be stabilized. Two strategies are listed here.

1. Maintain a constant height above the ground.
2. Maintain a constant stance leg length.

Maintaining a constant height can be implemented with a virtual spring-damper mechanism attached between the ground and the robot's body. The spring set point will determine the height above ground that the robot maintains. The damper will cause oscillations about that height to decay. A virtual vertical force of the weight of the body can be used to allow for lower virtual spring constants and decrease the DC offset.

Maintaining a constant stance leg length can be done in a number of different ways. A simple method which does not require high gain feedback is to push upwardly on the body a little harder than gravity. This will cause the robot to increase its height until the knee hits its joint limit (knee cap).

The first strategy was used with Spring Turkey [12], our previous walking robot. We successfully applied both strategies on Spring Flamingo but embraced the second one because it is much more efficient and more similar to biological walking. By walking with fully extended legs, there is a low torque requirement on the knee. The robot will walk with somewhat of a compass gait, in which potential energy and kinetic energy are out of phase and hence total mechanical energy is nearly constant.

2.2 Pitch Stabilization

As in height stabilization, stabilizing pitch is straightforward as long as we have a support leg firmly on the ground. Two strategies for stabilizing pitch are

1. Maintain a level pitch.
2. Follow a pitch trajectory.

Both strategies require feedback to implement since the center of gravity is above the hip. If it were below, we could just rely on the natural dynamics which would already be stable. To control the pitch, we use a virtual torsional spring damper mechanism. If a level pitch trajectory is desired, then the virtual spring set point is held constant. If a pitch trajectory is desired (perhaps to help control speed, as described below), then the set point is changed to match the desired pitch position. We used the level pitch strategy in the control of Spring Flamingo.

2.3 Speed Stabilization

Most speed stabilization techniques for bipedal walking are discrete events and thus control the speed from stride to stride rather than throughout a given stride. In fact, for dynamic bipedal walking, it is impossible to arbitrarily control the forward speed during a stride since the center of mass projection lies outside the support foot polygon during much of the stride.

Five strategies for stabilizing speed are

1. Change stride length with speed.
2. Change transition events with speed.
3. Servo speed when the center of mass is over the support foot or when the robot is in double support.
4. Change the location of the body center of mass by pitching forward or backward.
5. Apply energy at strategic times.

The first two strategies (stride length and transitions) are discrete events which will stabilize the speed over a number of strides. By changing stride lengths and transition events one can change the percent of the stride in which the support leg is in front of the body and hence is slowing down the robot and the percent of the stride in which the support leg is behind the body and hence is speeding up the robot.

The third strategy (servo speed when possible) can be used when the center of mass is over the support foot polygon or when the robot is in double support. The placement of the virtual toe point can be used in single support by placing it forward, closer to the front of the foot, to slow the robot down and placing it backward, closer to the heel, to speed it up. Force distribution can be used during double support. More force can be applied by the rear leg to speed the robot up and more force can be applied by the front leg to slow the robot down.

The fourth strategy (pitching the body) can be used to speed the robot up by leaning forward or slow it down by leaning backward. This strategy is typically seen when a person leans into a hill when walking up it or leans back to brake when going down it.

The fifth strategy (applying energy at strategic times) is a bit trickier to apply. It may be possible to put energy into the system in one mode during part of the stride and later slosh it into forward speed. For example, it may be possible to increase the potential energy of the system by having the robot go up onto its toes earlier and then convert that energy into forward kinetic energy during a later portion of the stride.

The first three strategies were used in stabilizing Spring Flamingo's speed. The last two strategies are currently being investigated.

2.4 Swing Leg Placement

In order to walk successfully, the swing leg must swing quickly to its next support location. Fortunately, the exact placement is not important when walking on smooth ground.

Two possible strategies for swinging the swing leg are

1. Servo the swing leg, either as a function of time, or as a function of the other leg.
2. Let the swing leg swing passively, making sure it does not hit the ground.

Following a swing trajectory can be implemented with a virtual spring-damper mechanism attached between the body and the ankle of the robot. The spring set point can move along the trajectory, pulling the leg along to follow it.

The natural pendulum dynamics of the swing leg are exploited in the second strategy. The leg will naturally swing forward, as long as the foot clears the ground.

Both strategies were successfully applied to Spring Flamingo. However, at high speeds we had some trouble with the second strategy as one cannot rely completely on the natural dynamics of the swing leg but must get it started and stop it at the end. This proved challenging at high speeds and is currently being explored further.

2.5 Support Transitions

To continuously walk forward, the support legs must be alternated since a given leg can only support the body over a small range. For bipedal walking, double support to single support and single support to double support transitions must occur at appropriate times.

Three strategies for transitioning from double support to single support are

1. Transition to single support if the body is within a certain distance to the next support leg.
2. Transition to single support if the body is over a certain distance away from the previous support leg.
3. Transition to the single support leg after being in double support for a certain amount of time.

These strategies can be implemented by measuring joint angles and transitioning based on joint angle threshholds or by computing the kinematics of the robot and transitioning on center of mass to foot distance threshholds. A state machine can be used to keep track of what support state the robot is in.

The first strategy will ensure that the next support leg will have a long enough support time that the other leg will be able to swing through in time. The second strategy will ensure that the rear leg has enough range of motion that the robot does not have to drag its rear leg. The third strategy simply transitions to single support after a given time.

We used the first two strategies together in the control of Spring Flamingo. When either event happens, the robot transitions to single support.

Two strategies for transitioning from single support to double support are,

1. Transition to double support if the body is over a certain distance away from the support leg.
2. Transition to double support if the swing leg has swung beyond a certain position or has slowed below a certain speed.

The first strategy ensures that the robot will transition onto a new support leg before the body becomes too far from the current support leg. This will guarantee that the current support leg can safely support the body and hence stabilize height and pitch.

The second strategy will ensure that the next support leg has swung far enough that it is in a position to support the body when the time comes. This strategy also maximizes double support time as the robot will put its swing foot down as soon as it has swung through rather than wait for the transition distance event to occur.

We used the first strategy in the control of Spring Flamingo.

3 Virtual Actuator Implementation for a Planar Biped With Feet and Ankles

We use Virtual Actuators [13] and Virtual Model Control [12], two techniques based on intuition, to implement some of the strategies of the previous section. In this section we present the mathematics to implement virtual components on Spring Flamingo for the support leg in single support or both legs in double support, following the procedure described in [13].

3.1 Single Leg Implementation

Figure 1 shows a simple planar, five link, four joint, serial robot model that we use to represent a single leg of our walking robot. The toe joint and link do not exist on the real robot (Figure 2). They are used to represent the point on the foot in which no torque is applied. We refer to this as the "virtual toe point". It is similar to the center of pressure on the foot or the zero moment point [16] except that it is a commanded quantity, not a measured one, and is based on static, not dynamic, considerations.

The virtual toe point can be used for control in the following intuitive sense. If it is desired to accelerate the robot backward (or reduce the forward acceleration) one can move the virtual toe point forward on the foot. Similarly, if it is desired to accelerate the robot forward (or reduce the backward acceleration) one can move the virtual toe point backward toward the heel.

We wish to connect a virtual component between the virtual toe point frame, {A}, and the body frame, {B}. The angles θ_t, θ_a, θ_k, and θ_h are those of the virtual toe, ankle, knee, and hip. The upper link (femur) is of length L_2, the lower link (tibia) is of length L_1, and the height of the foot is L_h. We assume that the virtual toe is flat on the ground, so that ${}^O_A R = I$. The commanded distance to the virtual toe point from the ankle is L_{vtp}.

The forward kinematic map from frame {A} to frame {B} is as follows,

$$ {}^A_B\vec{X} = \begin{bmatrix} x \\ z \\ \theta \end{bmatrix} = \begin{bmatrix} -L_{vtp}\, c_t - L_h\, s_t - L_1\, s_{t+a} - L_2\, s_{t+a+k} \\ -L_{vtp}\, s_t + L_h\, c_t + L_1\, c_{t+a} + L_2\, c_{t+a+k} \\ -\theta_h - \theta_k - \theta_a - \theta_t \end{bmatrix} \tag{1} $$

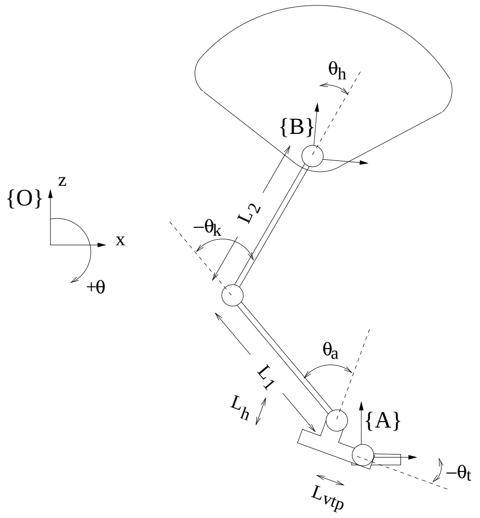

Figure 1: Single leg implementation. Reaction frame {A} is assumed to be in the same orientation as reference frame {O} so that ${}^O_A R = I$.

where $c_a = \cos(\theta_a)$, $s_a = \sin(\theta_a)$, etc. Partial differentiation produces the Jacobian,

$$ {}^A_B J = \begin{bmatrix} J_{1,1} & J_{1,2} & J_{1,3} & 0 \\ J_{2,1} & J_{2,2} & J_{2,3} & 0 \\ -1 & -1 & -1 & -1 \end{bmatrix} \tag{2} $$

where,

$$ \begin{aligned} J_{1,3} &= -L_2\, c_{t+a+k} \\ J_{1,2} &= J_{1,3} - L_1\, c_{t+a} \\ J_{1,1} &= J_{1,2} + L_{vtp}\, s_t - L_h\, c_t \\ J_{2,3} &= -L_2\, s_{t+a+k} \\ J_{2,2} &= J_{2,3} - L_1\, s_{t+a} \\ J_{2,1} &= J_{2,2} - L_{vtp}\, c_t - L_h\, s_t \end{aligned} $$

The Jacobian relates the virtual velocity between frames A and B with the joint velocities,

$$ {}^A_B\dot{\vec{X}} = {}^A_B J\, \dot{\vec{\Theta}} \tag{3} $$

and the virtual force to joint torque,

$$ \vec{\tau} = ({}^A_B J)^T\, ({}^A_B\vec{F}) \tag{4} $$

where $\vec{\tau}$ is the joint torque vector and $\vec{F}$ is the virtual force vector.

Next we add the constraint of an unactuated toe, $\tau_t = 0$, since we desire zero actuated torque about the virtual toe point. This will constrain the direction in which virtual forces can be applied. With the virtual toe point constraint, Equation 4 is,

$$ \begin{bmatrix} 0 \\ \tau_a \\ \tau_k \\ \tau_h \end{bmatrix} = \begin{bmatrix} J_{1,1} & J_{2,1} & -1 \\ J_{1,2} & J_{2,2} & -1 \\ J_{1,3} & J_{2,3} & -1 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} f_x \\ f_z \\ f_\theta \end{bmatrix} \tag{5} $$

For our walking robot we are more concerned about applying forces in the vertical direction and torques about the body than we are concerned about applying horizontal forces. Therefore, we specify f_z and f_θ and solve for f_x

$$f_x = \begin{bmatrix} \frac{-J_{2,1}}{J_{1,1}} & \frac{1}{J_{1,1}} \end{bmatrix} \begin{bmatrix} f_z \\ f_\theta \end{bmatrix} \tag{6}$$

Plugging equation 6 back into equation 5, we get

$$\begin{bmatrix} \tau_a \\ \tau_k \\ \tau_h \end{bmatrix} = \begin{bmatrix} \frac{-J_{1,2}J_{2,1}}{J_{1,1}} + J_{2,2} & \frac{J_{1,2}}{J_{1,1}} - 1 \\ \frac{-J_{1,3}J_{2,1}}{J_{1,1}} + J_{2,3} & \frac{J_{1,3}}{J_{1,1}} - 1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} f_z \\ f_\theta \end{bmatrix} \tag{7}$$

Throughout the above derivation we have assumed that the toe is flat on the ground and that we can measure all angles. Because there is no toe (it is virtual) we cannot measure its angle with the ground. Instead we measure the body angle via a potentiometer on a boom or a gyroscope and compute what the toe angle would be if the toe was flat on the ground,

$$\theta_t = -\theta - \theta_h - \theta_k - \theta_a$$

We now have a simple set of equations for determining joint torques given virtual forces. These equations will be used in the next section in the control of a bipedal walking robot during the single support phase.

3.2 Dual Leg Implementation

In the previous subsection we discussed virtual actuator implementation for a single leg (when the robot is in single support). Here we examine the double support case.

As in [13] we could construct a constraint matrix and exactly solve for the force distribution between the two legs such that any arbitrary F_x, F_z, F_θ force vector could be commanded. However, we decide to use a simpler method because

- There is no solution when the feet are together as the constraint matrix is not invertible in such a configuration.
- It is unlikely that biological creatures exactly solve the force distribution problem.
- Solving the force distribution problem exactly is unnecessary.
- The method presented below appeals more to intuition.

Instead of solving the force distribution problem exactly we simply distribute the force between the two legs with force distribution parameter α such that,

$$\begin{bmatrix} f_z \\ f_\theta \end{bmatrix}_l = \alpha \begin{bmatrix} F_z \\ F_\theta \end{bmatrix}, \quad \begin{bmatrix} f_z \\ f_\theta \end{bmatrix}_r = (1-\alpha) \begin{bmatrix} F_z \\ F_\theta \end{bmatrix} \tag{8}$$

where $0 \leq \alpha \leq 1$

As in the single support case, we do not command forces in the x direction. Instead we command forces in the z and θ directions, decide the force distribution between the two legs, and solve for the joint torques using Equation 7 for each leg.

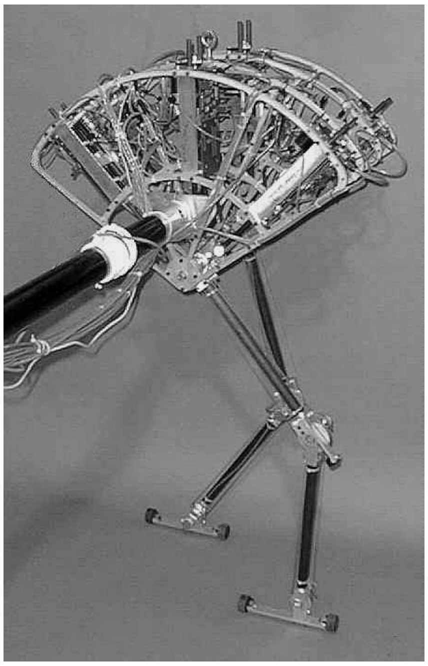

Figure 2: Spring Flamingo, a planar bipedal walking robot. There are six force controlled actuators attached to the body. Power is transmitted to the hips, knees, and ankles via cables. A boom prevents motion in the lateral, roll, and yaw directions.

We choose α such that the forces are divided between the legs in a natural way. If the robot's body is directly above the left leg, all forces are provided by the left leg ($\alpha = 1$). Similarly, if the robot's body is directly above the right leg, all forces will be provided by the right leg ($\alpha = 0$). If the robot's body is between the left and right legs, the forces will be divided with a linear relationship

$$\alpha = \frac{x_{right}}{x_{left} + x_{right}} \tag{9}$$

where $x_{left} \geq 0$ is the horizontal distance from the left leg to the body and $x_{right} \geq 0$ is the horizontal distance from the right leg to the body. If the legs are both close together, it is very much like the single support case and we simply set $\alpha = 0.5$, dividing the forces evenly between the two legs.

We can modify the force distribution parameter α for control in the following way. To accelerate forward, put more of the force distribution on the rear leg. To accelerate backward, put more of the force distribution on the front leg. For example we can use the simple control law $\alpha = \alpha_0 + b(\dot{x}_d - \dot{x})$ to help regulate velocity when the left leg is the rear leg.

In the next section we will use this control strategy to help regulate forward velocity during double support.

4 Intuitive Control Strategies Applied to a Bipedal Walking Robot

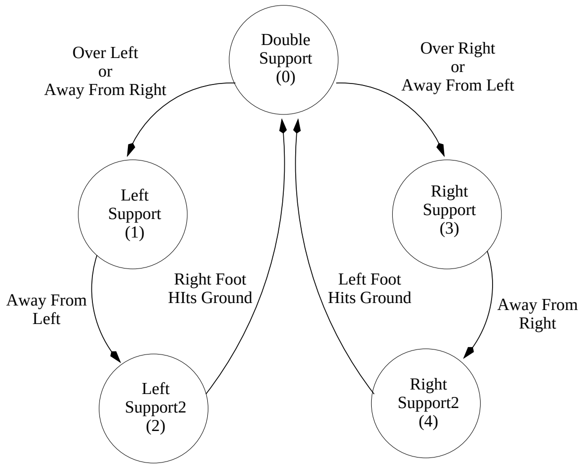

Figure 3: State machine used in Spring Flamingo's walking algorithm.

Figure 2 is a photograph of Spring Flamingo, a planar bipedal walking robot. The robot has an actuated hip, knee, and ankle on each leg. An unactuated boom constrains Spring Flamingo's roll, yaw, and lateral motion, thereby reducing it to a planar robot. All of Spring Flamingo's motors are located in its upper body, with power being transmitted to the joints via cable drives. Series Elastic Actuation [11] is employed at each degree of freedom, allowing for accurate application of torques and a high degree of shock tolerance. The maximum torque that can be applied to the hips and ankles is approximately 18 Nm, while approximately 24 Nm can be applied to the knees. The force control bandwidth we achieve is approximately 20 Hz. Spring Flamingo weighs approximately 30 lbs (14 kg) and stands 3 ft (1 m) tall from floor to hip.

Potentiometers at the hips, knees, ankles, and boom measure joint angles and body pitch. Compression springs are used in the joint actuators to implement Series Elastic Actuation. Linear potentiometers measure the stretch in the springs.

4.1 Walking Algorithm

The intuitive control strategies used on Spring Flamingo are:

- Maintain a constant stance leg length by pushing up until hitting the knee cap.
- Maintain a constant level pitch using a virtual spring damper mechanism with constant set point.
- Transition from double support to single support if the body's x position becomes further than a certain distance from the rear foot or closer than a certain distance from the front foot.
- Transition from single support to double support if the body's x position becomes further than a certain distance from the support foot.
- Swing the non-stance leg so that the foot is roughly placed a nominal stride length away from the support foot when transitioning to double support.
- Increase the nominal stride length as the robot walks faster.

Table 1: Important Parameters for Spring Flamingo's Walking Algorithm.

Parameter	Value	Range
Height Control		
Virtual Z Anti-Gravity Force	110 N	90-120
Virtual Z Damper	$200 \frac{N}{m/s}$	100-250
Pitch Control		
Virtual Pitch Spring	$60 \frac{Nm}{rad}$	30-80
Virtual Pitch Damper	$10 \frac{Nm}{rad/s}$	4-15
Forward Speed Control		
Nominal Velocity	$0.4 \frac{m}{s}$	0.0-0.7
Virtual Toe Point Gain	$0.3 \frac{m}{m/s}$	0.0-0.5
Double Support Transfer Ratio Gain	$0.3 \frac{\%}{m/s}$	0.0-1.5
Double to Single Support Transition Distance Gain	$0.3 \frac{m}{m/s}$	0.0-0.5
Single to Double Support Transition Distance Gain	$0.3 \frac{m}{m/s}$	0.0-0.5
Swing Leg Control		
Virtual Swing Leg X Spring	$25 \frac{N}{m}$	10-40
Virtual Swing Leg X Damper	$3 \frac{N}{m/s}$	1-5
Virtual Swing Leg Z Spring	$150 \frac{N}{m}$	100-200
Virtual Swing Leg Z Damper	$8 \frac{N}{m/s}$	2-14
Support Transitions		
Double to Single Support Rear Leg Transition Distance	$0.26m$	0.20-0.30
Double to Single Support Front Leg Transition Distance	$0.05m$	0.01-0.10
Single to Double Support Transition Distance	$0.16m$	0.10-0.24
Nominal Stride Length	$0.36m$	0.24-0.42

- Transition to double support later if the robot is walking too slowly or sooner if the robot is walking too quickly.
- Maintain the virtual toe point of the support foot approximately below the center of mass. Move it forward if walking too quickly or backward if walking too slowly.
- During double support put more of the load on the back leg if walking too slowly and more on the front leg if walking too quickly.

To implement Spring Flamingo's walking algorithm, we use a simple set of virtual components and a state machine shown in Figure 3.

The various virtual spring, damper, and force variables and walking parameters are chosen using physical insight and a manual search. Some of the parame-

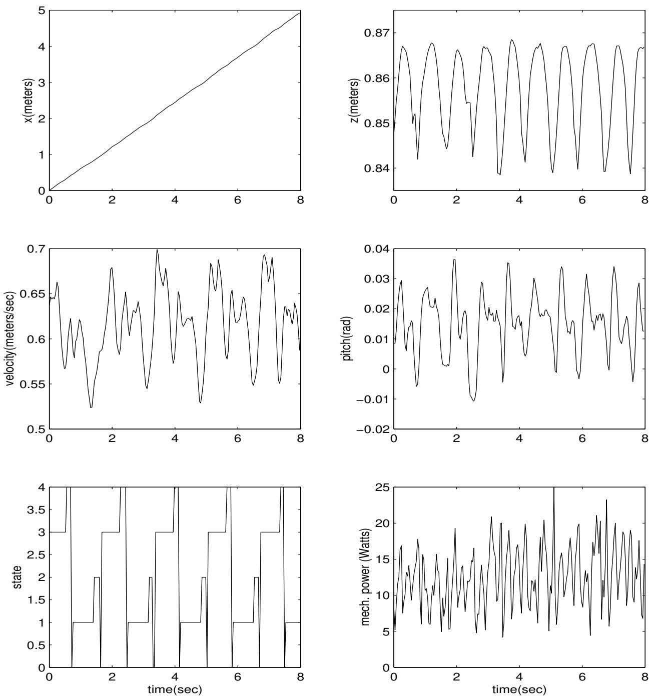

Figure 4: Spring Flamingo walking data. Left graphs display, from top to bottom the horizontal position (x) and horizontal velocity of the body and the stat of the state machine. Right graphs display the vertical position (z) and pitch (θ) of the body and mechanical power being applied to the joints.

ters are listed in Table 1 along with their tuned values and reasonable ranges. This range represents the reasonable amount that the parameter can vary by while the robot remains well-behaved. For some parameters, the robot can continue to walk throughout this range. For others, the robot can not walk if the parameter is at the boundaries of the range. All the parameters can be individually varied by at least 10% from the tuned value without disrupting the stable walking.

The vertical force to control height is calculated to be a little larger than the weight of the robot. Many parameters are tuned by physically examining their effects (resistance to being pushed on, decay rate, etc.) until the desired effects are achieved and the robot walks successfully. These walking parameters consist of nominal stride length, transition distances, swing leg gains, and velocity gains on the transition distances, stride length, virtual toe points, and double support loading ratio. From the time the robot was built until the moment it could continuously walk, approximately 40 iterations were performed over a span of 3 weeks.

Spring Flamingo is initialized balancing with its feet together. It starts walking by lifting up one leg and transitioning into the single support phase. At no time is external intervention required. The robot stops

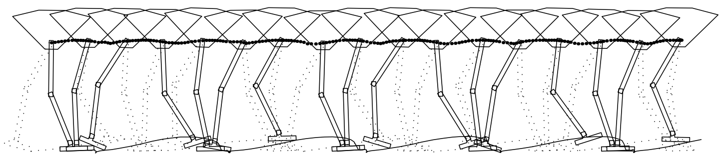

Figure 5: Elapsed time snapshot of the bipedal walking data in Figure 6. The drawings of the robot are spaced approximately 0.5 seconds apart. The left leg is dotted while the right leg is solid. Lines show the path of the tips of the feet and the hip trajectory. The robot walks from left to right.

by setting its desired velocity to zero after walking a given distance.

Figure 4 shows experimental data from Spring Flamingo while walking. The graphs on the left show (from top to bottom) the body's horizontal position (x), horizontal velocity, and state. The graphs on the right show the body's vertical position (z), pitch (θ), and the mechanical power being exerted at the joints. The mechanical power is computed as the sum of the absolute value of the torque times angular velocity at each joint.

The data in Figure 4 is plotted in graphical form in Figure 5. The snapshots in Figure 5 are approximately 0.5 seconds apart. Lines are drawn to show the path of the tips of the feet and the hip trajectory.

Spring Flamingo walked continuously at approximately 0.63 m/s. The data shows approximately 10 steps (left to right or right to left support transitions) in 8 sec, giving a step time of 0.8 seconds. The height fluctuated about 3 cm as the robot walked using a compass-like gait. The pitch was confined to ± 0.04 radians (± 2.1 deg). The mechanical power averaged about 15 watts. However, due to the inefficiencies of the motors, transmissions, and power electronics, the electrical power consumed is probably much higher.

4.2 Robustness of Walking Algorithm

The intuitive walking control algorithm discussed above is somewhat robust to external forces, rough terrain, and parameter variation. Spring Flamingo can be pushed fairly hard in either direction, temporarily changing its speed by about 25%, recovering to the original speed within a few steps. The robot can walk up and down slopes of approximately 5^0 without any change in the algorithm and without being informed of or detecting the presence of the slope. All of the control parameters can be individually changed by 10% or more while still maintaining stable walking.

The most common failure mode occurs when the robot is pushed too quickly and can not recover. The robot will typically take several short, choppy steps further increasing its speed and finally falling. A biological creature in this situation typically recovers by running a few steps and slowing down. Unfortunately, Spring Flamingo can not run and hence has no recourse when its speed increases above its natural walk to run transition speed.

4.3 Self-Stabilizing Speed

The above algorithm used several controllers to stabilize speed. In another algorithm, we successfully compelled Spring Flamingo to walk stably without any feedback on the forward speed. We used the speed control strategy "Take longer strides as the robot walks faster" but we never implicitly programmed it. Instead, we used low gains on the swing leg such that overshoot was significant and the natural dynamics of the system played a large roll in where the leg was placed. As the robot walked faster it naturally took longer strides without explicitly being told to. The speed-dependent stride lengths then self-stabilized the forward walking speed.

With this self-stabilizing speed algorithm, the robot walked continuously while being robust to external pushes. However, we could only get this algorithm to work for slow speed walking. Future work may focus on using a self-stabilizing speed algorithm at higher speeds.

5 Conclusions

Spring Flamingo walked continuously using a simple set of intuitive control strategies. These strategies are easy to develop, are easy to understand, and are easy to implement. In short, planar bipedal walking is easy to achieve despite being difficult to analyze mathematically.

By tuning the intuitive control parameters by hand, one gains insight into how the parameters relate to the resultant walking. This insight in turn helps speed up the tuning process. We are currently investigating the possibility of using learning or adaptive techniques to tune the parameters automatically.

Spring Flamingo gained several advantages over Spring Turkey [12] by the use of feet and actuated ankles. Since Spring Turkey had only point feet, it could not balance on one foot, had to walk with bent knees, and had large velocity fluctuations during each stride. Spring Flamingo exhibits smaller velocity fluctuations as it keeps its virtual toe point directly underneath when the center of mass passes over the support foot. It walks more efficiently as a compass gait with straight legs can be used without the worry of losing range of motion on the rear leg. Also, with feet and ankles, the robot is able to stand and balance on one leg.

Stable, robust, and efficient planar bipedal walking can be achieved using intuitive control strategies and intuitive control techniques. Spring Flamingo walked over moderate slopes with no change to its level ground algorithm. We are currently focusing on a few simple intuitive control strategies for dealing with more formidable slopes.

We are confident that we can develop similar strategies for three dimensional bipedal walking. We are currently developing such strategies and applying them to walking simulations.

References

[1] E. Dunn and R. Howe. Towards smooth bipedal walking. *IEEE Conference on Robotics and Automation*, pages 2489–2494, 1994.

[2] E. Dunn and R. Howe. Foot placement and velocity control in smooth bipedal walking. *IEEE Conference on Robotics and Automation*, pages 578–583, 1996.

[3] L. Jalics, H. Hemami, and B. Clymer. A control strategy for adaptive bipedal locomotion. *IEEE Conference on Robotics and Automation*, pages 563–569, 1996.

[4] S. Kajita and K.Tani. Experimental study of biped dynamic walking in the linear inverted pendulum mode. *IEEE Conference on Robotics and Automation*, pages 2885–2891, 1995.

[5] S. Kajita and K.Tani. Adaptive gait control of a biped robot based on realtime sensing of the ground profile. *IEEE Conference on Robotics and Automation*, pages 570–577, 1996.

[6] A. Kun and W. T. Miller. Adaptive dynamic balance of a biped robot using neural networks. *IEEE Conference on Robotics and Automation*, pages 240–245, 1996.

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

[8] W. T. Miller. Real time neural network control of a biped walking robot. *IEEE Control Systems Magazine*, Feb:41–48, 1994.

[9] H. Miura and I. Shimoyama. Dynamic walk of a biped. *International Journal of Robotics Research*, 3(2):60–74, 1984.

[10] Simon Mochon and Thomas A. McMahon. Ballistic walking: An improved model. *Mathematical Biosciences*, 52:241–260, 1979.

[11] Gill A. Pratt and Matthew M. Williamson. Series elastic actuators. *IEEE International Conference on Intelligent Robots and Systems*, 1:399–406, 1995.

[12] J. Pratt, P. Dilworth, and G. Pratt. Virtual model control of a bipedal walking robot. *IEEE Conference on Robotics and Automation*, pages 193–198, 1997.

[13] J. Pratt, A. Torres, P. Dilworth, and G. Pratt. Virtual actuator control. *IEEE International Conference on Intelligent Robots and Systems*, pages 1219–1226, 1996.

[14] Marc H. Raibert. *Legged Robots That Balance*. MIT Press, Cambridge, MA, 1986.

[15] Ann L. Torres. Implementation of virtual model control on a walking hexapod. May 1996. Undergraduate Thesis, Massachusetts Institute of Technology.

[16] M. Vukobratovic, B. Borovac, D. Surla, and D. Stokic. *Biped Locomotion: Dynamics, Stability, Control, and Applications*. Springer-Verlag, Berlin, 1990.

[17] J. Yamaguchi, A. Takanishi, and I. Kato. Development of a biped walking robot adapting to a horizontally uneven surface. *IEEE International Conference on Intelligent Robots and Systems*, pages 1156–1163, 1994.

[18] K. Yi and Y. Zheng. Biped locomotion by reduced ankle power. *IEEE Conference on Robotics and Automation*, pages 584–589, 1996.