Proceedings of the 1997 IEEE
International Conference on Robotics and Automation
Albuquerque, New Mexico - April 1997

# Virtual Model Control of a Bipedal Walking Robot

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

## Abstract

*The transformation from high level task specification to low level motion control is a fundamental issue in sensorimotor control in animals and robots. This paper describes a control scheme called Virtual Model Control that addresses this issue.*

*Virtual Model Control is a motion control language that uses simulations of imagined mechanical components to create forces, which are applied through real joint torques, thereby creating the illusion that the virtual components are connected to the robot. Due to the intuitive nature of this technique, designing a Virtual Model Controller requires the same skills as designing the mechanism itself. A high level control system can be cascaded with the low level Virtual Model Controller to modulate the parameters of the virtual mechanisms. Discrete commands from the high level controller would then result in fluid motion.*

*Virtual Model Control has been applied to a physical bipedal walking robot. A simple algorithm utilizing a simple set of virtual components has successfully compelled the robot to walk continuously over level terrain.*

## 1 Introduction

Dynamic legged robots suffer from lack of powerful control techniques. These robots are extremely difficult to control since they are nonlinear and operate throughout the range of their state space; act in a gravity field; interact with a semi-structured, complex environment; are nominally unstable; are Multi Input, Multi Output (MIMO); exhibit time variant and intermittent dynamics; and require both continuous control and discrete control (for step-to-step transitions).

In addition, the performance measures of such robots are much different from typical notions of performance such as command following and disturbance rejection. Performance for these robots is usually defined in terms of biological similarity, efficiency, locomotion smoothness, top speed, and robustness to rough terrain.

Because of these difficulties, the only acceptable tools for analyzing such systems are often simulation or experimentation and the only good design tools are often physical intuition, parameter iteration, and "hand tweaking".

Anyone who wishes to expand the toolbox of analysis and design techniques for such a class of robots will typically either make an advancement in control system design and analysis mathematics, develop automatic techniques, or exploit physical intuition. In this paper we present a control technique, called Virtual Model Control, which is based on the latter approach.

### 1.1 Virtual Model Controllers

Virtual Model Control is a language for describing interactive force behaviors. 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 simple springs, dampers, dashpots, masses, latches, bearings, non-linear potential and dissipative fields, or any other imaginable component. Virtual components can even contain adaptive and learning elements [12]. Virtual Model Control borrows ideas from Virtual Reality, Hybrid Position-Force Control [14], Stiffness Control [16], Impeadance Control [2], and the Operational Space Formulation [5].

Many complex tasks that are difficult to describe using traditional techniques can be readily characterized with a simple set of virtual components. For example, consider a robot wishing to impart an impact onto an unknown surface (e.g. knocking on a door). Ordinarily, this would be a very difficult task to specify. However, with Virtual Model Control, we need merely attach a virtual mass with a given kinetic energy to the robot's hand via a virtual spring and damper. The robot's hand will now move to strike out, and, after imparting the desired impact to the environment, bounce back due to mass resonating with the virtual spring-damper.

Some benefits of Virtual Model Control are that it is compact, requires relatively small amounts of computation, and can be implemented in a distributed manner (see [13, 11] for information on how to implement virtual components). Furthermore, a high level controller could be implemented as a state machine which simply changes virtual component connections or parameters at the state transitions. Even though a discrete high level controller would be used, the overall motion would be fluid since the virtual components are continuous.

Note that Virtual Model Control does not use dynamic inversion to alter the behavior of the robot. We like to call dynamic inversion the "virtual robot" approach. We believe that the virtual robot approach should only be used when high performance requirements or other extreme situations dictate. This is be-

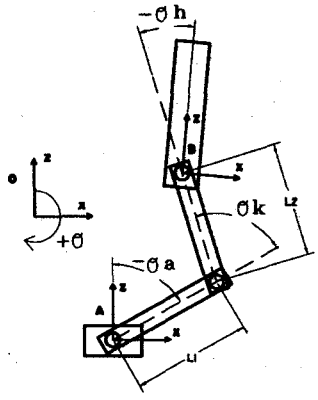


Figure 1: Single Leg Implementation. Reaction frame {A} is assumed to be in the same orientation as reference frame {O} so that ${}^{O}_{A}R = I$.

cause plant inversion adds computational complexity; fighting the natural dynamics of the robot can be inefficient; undesirable natural dynamics is an indication that the real robot was designed improperly; and overcompensation can lead to instability.

Also note that with Virtual Model Control, we usually talk in terms of spring set points, for example, and not *commanded* positions. Except for actuator and computation non-idealities, we can perfectly implement virtual components whereas very few control algorithms can perfectly track a commanded trajectory. In this light we in the Leg Laboratory believe that robots cannot be *commanded* to perform a task; they can only be given *hints* and *suggestions.*

Virtual Model Control has been used to control a dynamic walking bipedal robot (described below) and an agile 3D hexapod in simulation [17].

## 2 Virtual Model Implementation for a Biped

In this Section we present the mathematics to implement virtual components on our bipedal robot for the support leg in single support or both legs in double support. This follows the procedure described in [11].

### 2.1 Single Leg Implementation

Figure 1 shows a simple 2-D, four link, three joint, serial robot model that we use to represent a single leg of our walking robot. We wish to connect a virtual component between frame {A}, which is attached to the foot, and frame {B}, which is attached to the body. The angles $\theta_a$, $\theta_k$, and $\theta_h$ are those of the ankle, knee, and hip. The lower link (tibia) is of length $L_1$, while the upper link (femur) is of length $L_2$. In this example we assume that the foot is flat on the ground, so that ${}^{O}_{A}R = I$.

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

$$ {}^{A}_{B}\vec{X} = \begin{bmatrix} x \\ z \\ \theta \end{bmatrix} = \begin{bmatrix} -L_1 s_a - L_2 s_{a+k} \\ L_1 c_a + L_2 c_{a+k} \\ -\theta_h - \theta_k - \theta_a \end{bmatrix} \quad (1) $$

Partial differentiation produces the Jacobian,

$$ {}^{A}_{B}J = \begin{bmatrix} -L_1 c_a - L_2 c_{a+k} & -L_2 c_{a+k} & 0 \\ -L_1 s_a - L_2 s_{a+k} & -L_2 s_{a+k} & 0 \\ -1 & -1 & -1 \end{bmatrix} \quad (2) $$

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

$$ {}^{A}_{B}\vec{X} = {}^{A}_{B}J\,\dot{\vec{\Theta}} \quad (3) $$

and the virtual force to joint torque,

$$ \vec{\tau} = ({}^{A}_{B}J)^T\,({}^{A}_{B}\vec{F}) \quad (4) $$

The Jacobian is of full rank, indicating that all virtual force directions are admissible. We add the constraint of an unactuated ankle, $\tau_a = 0$, since the real robot has a point foot and no ankle. This will constrain the direction in which virtual forces can be applied. With a limp ankle, Equation 4 is constrained,

$$ \begin{bmatrix} 0 \\ \tau_k \\ \tau_h \end{bmatrix} = \begin{bmatrix} -L_1 c_a - L_2 c_{a+k} & -L_1 s_a - L_2 s_{a+k} & -1 \\ -L_2 c_{a+k} & -L_2 s_{a+k} & -1 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} f_x \\ f_z \\ f_\theta \end{bmatrix} \quad (5) $$

For our walking robot we are more concerned about applying forces in the vertical direction and torques about the body then we are concerned about applying horizontal forces. Therefore, we specify $f_z$ and $f_\theta$ and solve for $f_x$

$$ f_x = \frac{-1}{L_1 c_a + L_2 c_{a+k}} \begin{bmatrix} L_1 s_a + L_2 s_{a+k} & 1 \end{bmatrix} \begin{bmatrix} f_z \\ f_\theta \end{bmatrix} \quad (6) $$

Plugging this back into 5, we get

$$ \begin{bmatrix} \tau_k \\ \tau_h \end{bmatrix} = \begin{bmatrix} \frac{-L_1 L_2 s_k}{L_1 c_a + L_2 c_{a+k}} & \frac{-L_1 c_a}{L_1 c_a + L_2 c_{a+k}} \\ 0 & -1 \end{bmatrix} \begin{bmatrix} f_z \\ f_\theta \end{bmatrix} \quad (7) $$

We now have a simple set of equations for determining joint torques given virtual forces. Note that the matrix in Equation 7 is of full rank for all values of $\vec{\Theta}$ except for $\theta_k = 0$. This corresponds to a fully extended knee, for which no virtual forces can be applied in the $z$ direction. If the knee is not fully extended, all virtual forces are admissible. Throughout this example we have assumed that the feet are flat on the ground and that we can measure all angles. In actuality, we use point feet and measure the body angle via a potentiometer on a boom or a gyroscope, rather than the ankle. Therefore, we must make the substitution

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

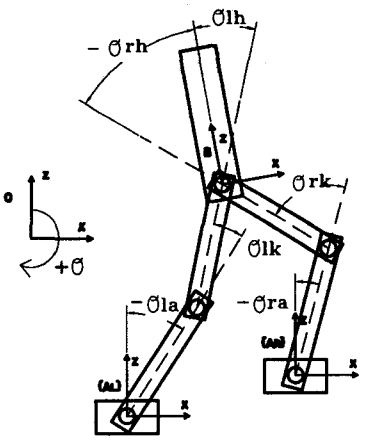


Figure 2: Dual Leg example. Reaction frames $\{A_l\}$ and $\{A_r\}$ are assumed to be in the same orientation as reference frame {O} so that ${}^{O}_{A_l}R = {}^{O}_{A_r}R = I$.

These equations will be used in the next Section in the control of a bipedal walking robot during the single support phase.

## 2.2 Dual Leg Implementation

The previous example discussed a serial chain manipulator. Here we examine a parallel mechanism representing a simple, 2-D, bipedal robot (See Figure 2).

Our model consists of the previous single leg example plus another leg. We wish to connect a multi-frame virtual component between the reaction frames $\{A_l\}$ and $\{A_r\}$ which are connected to the feet, and the action frame {B} which is connected to the body. The individual leg parameters and joint angles are identical to those of the single leg example with the $l$ subscript denoting the left leg and the $r$ subscript denoting the right leg. Again, we assume that the feet are flat on the ground so that ${}^{O}_{A_l}R = {}^{O}_{A_r}R = I$.

We already have the kinematics for each leg from the previous example. To calculate the body kinematics, we choose to use the average value of the kinematics from the two legs.

We have computed the Jacobian for each serial link of this parallel mechanism in the previous example. We now combine them in the following manner,

$$\begin{bmatrix} \vec{\tau_l} \\ \vec{\tau_r} \end{bmatrix} = \begin{bmatrix} {}^{A_l}_{B}J^T & 0 \\ 0 & {}^{A_r}_{B}J^T \end{bmatrix} \begin{bmatrix} \vec{F_l} \\ \vec{F_r} \end{bmatrix} \tag{8}$$

This expands to

$$\begin{bmatrix} \tau_{la} \\ \tau_{lk} \\ \tau_{lh} \\ \tau_{ra} \\ \tau_{rk} \\ \tau_{rh} \end{bmatrix} = \begin{bmatrix} A & B & -1 & 0 & 0 & 0 \\ Q & R & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & C & D & -1 \\ 0 & 0 & 0 & S & T & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} f_{xl} \\ f_{zl} \\ f_{\theta l} \\ f_{xr} \\ f_{zr} \\ f_{\theta r} \end{bmatrix} \tag{9}$$

where

$$A = -L_1\cos(\theta_{la}) - L_2\cos(\theta_{la} + \theta_l k)$$
$$B = -L_1\sin(\theta_{la}) - L_2\sin(\theta_{la} + \theta_l k)$$
$$C = -L_1\cos(\theta_{ra}) - L_2\cos(\theta_{ra} + \theta_{rk})$$
$$D = -L_1\sin(\theta_{ra}) - L_2\sin(\theta_{ra} + \theta_{rk})$$
$$Q = -L_2\cos(\theta_{la} + \theta_{lk}),\quad R = -L_2\sin(\theta_{la} + \theta_{lk})$$
$$S = -L_2\cos(\theta_{ra} + \theta_{rk}),\quad T = -L_2\sin(\theta_{ra} + \theta_{rk})$$

Equation 9 maps the virtual forces for each leg to the required joint torques, whereas we wish to specify a single virtual force. We therefore need to solve the individual leg forces in terms of the combined virtual force subject to several constraints.

Since the action frame {B} is coincidental, we have the compatibility relation that the force vector must equal the vector sum of the forces produced by each serial chain,

$$\begin{bmatrix} f_x \\ f_z \\ f_\theta \end{bmatrix} = \begin{bmatrix} f_{xl} \\ f_{zl} \\ f_{\theta l} \end{bmatrix} + \begin{bmatrix} f_{xr} \\ f_{zr} \\ f_{\theta r} \end{bmatrix} \tag{10}$$

Since we have six joints and wish to control three force directions, we require three constraints. Unactuated ankles provide two constraints,

$$\tau_{la} = 0,\quad \tau_{ra} = 0 \tag{11}$$

The third constraint provides us with a design degree of freedom. We could choose it to maximize some performance criterion, etc. Here we simply choose to match the hip torques,

$$\tau_{lh} = \tau_{rh} \Longrightarrow f_{\theta l} = f_{\theta r} \tag{12}$$

Putting the above constraints in vector form we have,

$$\begin{bmatrix} f_x \\ f_z \\ f_\theta \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ A & B & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & C & D & -1 \\ 0 & 0 & 1 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} f_{xl} \\ f_{zl} \\ f_{\theta l} \\ f_{xr} \\ f_{zr} \\ f_{\theta r} \end{bmatrix} \tag{13}$$

We must now perform a 6 by 6 matrix inversion to solve for the individual leg forces. We drop the terms which are multiplied by zero. The result is a 6 by 3 matrix relating the single vector of virtual forces to the individual leg virtual forces. This matrix can then be plugged into Equation 9 and simplifying, we get the virtual force to joint torque relation

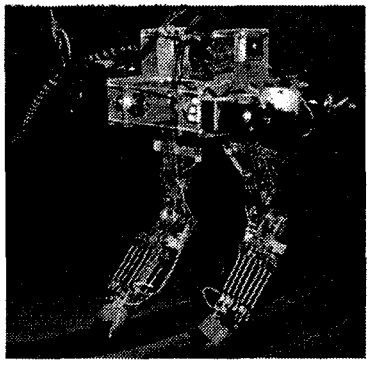

Figure 3: Spring Turkey, our bipedal walking robot. There are four actuators attached to the body. Power is transmitted to the hips and knees via cables. The unactuated feet consist of a U-shaped strip of rubber. A boom is used to prevent motion in the lateral, roll, and yaw directions. Note the spring packs used to implement series elastic actuation.

$$\begin{bmatrix} \tau_{lk} \\ \tau_{lh} \\ \tau_{rk} \\ \tau_{rh} \end{bmatrix} = \begin{bmatrix} \frac{CV}{E} & \frac{DV}{E} & \frac{-V-QD+RC}{2E} - \frac{1}{2} \\ 0 & 0 & -1/2 \\ \frac{-AW}{E} & \frac{-BW}{E} & \frac{W+SB-TA}{2E} - \frac{1}{2} \\ 0 & 0 & -1/2 \end{bmatrix} \begin{bmatrix} f_x \\ f_z \\ f_\theta \end{bmatrix} \quad (14)$$

where

$$E = CB - AD$$

$$V = QB - RA = -L_1 L_2 \sin(\theta_{lk})$$

$$W = SD - TC = -L_1 L_2 \sin(\theta_{rk})$$

Once again, we have a simple set of equations for relating virtual forces to joint torques. Intuitively, the matrix in Equation 14 should be of full rank for all $\Theta$ except when a knee is fully extended or the two feet and hip are colinear. In all other configurations, all virtual forces are admissible. Again, we will use point feet and measure the body angle via a potentiometer on a boom or a gyroscope, rather than the ankle angles. Therefore, we must make the substitutions

$$\theta_{la} = -\theta - \theta_{lh} - \theta_{lk}, \quad \theta_{ra} = -\theta - \theta_{rh} - \theta_{rk}$$

These equations will be used in the next Section in the control of a bipedal walking robot during double support phase.

# 3 Bipedal Walking Robot

Figure 3 is a photograph of Spring Turkey, our bipedal walking robot. Spring Turkey was designed and built by Peter Dilworth and Jerry Pratt in 1994. It has an actuated hip and knee on each leg. An unactuated boom constrains Spring Turkey's roll, yaw, and lateral motion thereby reducing it to a planar robot.

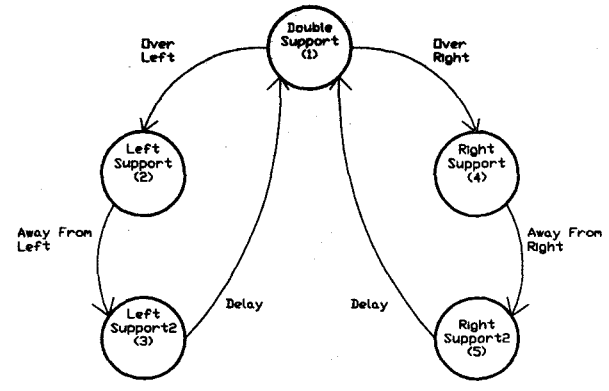


Figure 4: State machine used in the turkey walking algorithm.

All of Spring Turkey's motors are located in its upper body, with power being transmitted to the joints via cable drives. Series Elastic Actuation [10] 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 is approximately 12 Nm while approximately 18 Nm can be applied to the knees. The force control bandwidth we achieve is approximately 20 Hz. Spring Turkey weighs approximately 22 lbs (10 kg) and stands 2 ft (60 cm) tall from toe to hip.

Potentiometers at the hips, knees, and boom measure joint angles and body pitch. Extension springs are used at the hips and knees to implement Series Elastic Actuation [10]. Linear potentiometers measure the stretch in the springs.

## 3.1 Turkey Walking

Virtual Model Control was applied to Spring Turkey to allow it to perform simple walking. The algorithm is as follows:

- Attempt to maintain a constant height and pitch.
- Transition from double support to single support if the body's x position becomes close to the point where a foot contacts the ground.
- Transition from single support to double support if the body's x position becomes far away from the support foot's ground contact point.
- Try to swing the non-stance leg so that the foot is placed a nominal stride length away from the support foot when transitioning to double support.
- During double support attempt to correct for velocity disturbances.

To implement turkey walking, we use a simple set of virtual components and a state machine. Figure 4 shows the state machine and Table 1 lists the trigger and branch events and the virtual components which are utilized in each state. During both double support and single support, a virtual granny walker

Table 1: Details of turkey walking state machine.

| State | Trigger Event | Virtual Components |
|---|---|---|
| 1 Double Support | Delay after left or right support2 | Granny Walker Dogtrack Bunnie |
| 2 Left Support | Body nearly over left foot. | Granny Walker Swing Leg Linkage |
| 3 Left Support2 | Body away from left foot. | Granny Walker |
| 4 Right Support | Body nearly over right foot. | Granny Walker Swing Leg Linkage |
| 5 Right Support2 | Body away from right foot. | Granny Walker |

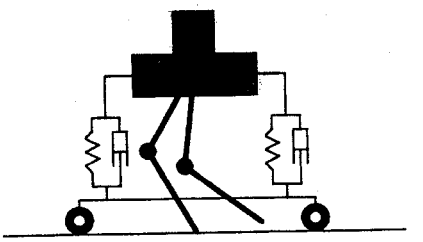

Figure 5: Spring Turkey with virtual granny walker mechanism.

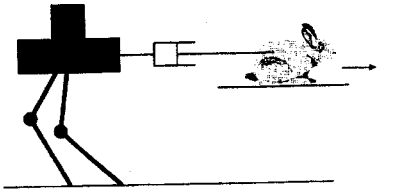

Figure 6: Spring Turkey with virtual dog track bunny mechanism.

with spring-damper mechanisms (Figure 5) maintains a constant height and regulates the pitch angle to zero.

During double support, a virtual dogtrack bunny with a damper mechanism (Figure 6) applies a virtual force in the forward horizontal (x) direction to help maintain a desired velocity. Unlike many speed control algorithms, which operate by modulating foot placement, we chose to leave foot placement a free variable (so that the robot could choose to avoid stepping in certain areas). Instead we modulated "food placement" by use of the virtual dogtrack bunny.

The swing leg is controlled via a virtual linkage with springs and dampers which compel the swing leg to mirror the stance leg while clearing the ground and to set down at the nominal stride length before transitioning back to double support. States Left Support 2 and Right Support 2 are used as buffer states between single and double support. Because Spring Turkey has no foot switches to detect ground contact, in these states the swing leg is simply made limp (zero torque applied to the joints) for a set delay time, allowing for the swing leg to fall to the ground before the large forces, which double support require, are applied.

The various virtual spring, damper, and force variables and walking parameters were chosen using physical insight and a manual search. The virtual granny walker spring-damper constants were experimentally varied while physically examining their effects (resistance to being pushed on, decay rate, etc.) until the desired effects were achieved; the walking parameters and virtual dogtrack bunny damper were changed through trial and error until the robot successfully walked. These walking parameters consisted of nominal stride length and percent of stride length spent in single support.

Walking was initiated in the single support phase. A slight push was applied to the robot to propel it forward. After the push, no external intervention was required.

Figure 8 shows experimental data from Spring Turkey while performing turkey walking. The upper left graphs show the body's horizontal position (x), vertical position (z), and pitch (theta), and the corresponding spring set points (dotted). The upper right graphs show the virtual forces applied to the body due to the virtual components. The horizontal velocity, along with the virtual dogtrack bunny velocity (dotted) is plotted in the lower left graph. The state of the state machine is plotted in the lower right graph.

The data in Figure 8 is plotted in graphical form in Figure 7. The snapshots in Figure 7 are approximately 0.5 seconds apart. Lines are drawn to show the path of the tips of the feet and the center of the body.

Spring Turkey walked continuously at approximately 0.5 m/s (1.125 mph). The data shows approximately 6 steps (left to right or right to left support transitions) in 4 sec, giving a step time of 0.5 seconds. It deviated a maximum of 3 cm from the nominal height of 54 cm and pitch was confined to $\pm 0.10$ radians ($\pm 5.2$ deg).

### 3.2 Stupid Walking

Another algorithm which we tried, called stupid walking, was identical to turkey walking except there was no speed control mechanism (no virtual dogtrack bunny with damper). With this algorithm, the state trajectory converged to a stable limit cycle for an appropriate choice of initial conditions and compelled the robot to walk as many as 14 steps. However, it

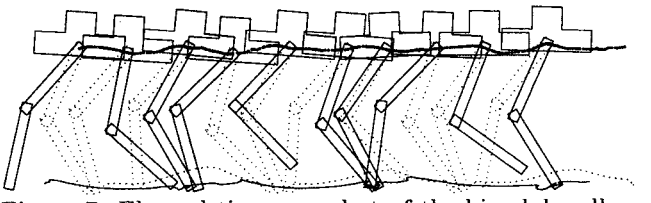

Figure 7: Elapsed time snapshot of the bipedal walking data in Figure 6. The drawings of the robot are spaced approximately 0.5 seconds apart. The left leg is dotted while the right leg is solid. Lines show the path of the tips of the foot and the center of the body.

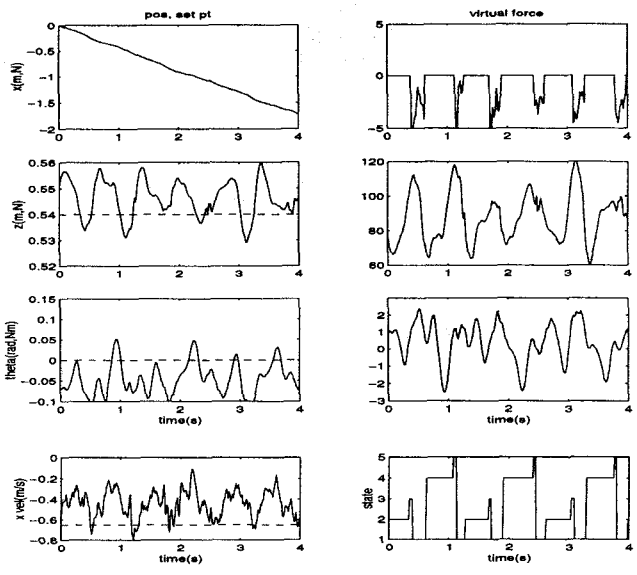


Figure 8: Turkey walking data. Upper left graphs display the $x$, $z$, and $\theta$ positions and virtual spring set points (dashed). Upper right graphs display the resultant forces applied to the body due to the virtual components. The lower left graph shows the body velocity and the dogtrack bunny velocity (dotted). The lower right graph shows the state machine transitions.

was extremely dependent on initial conditions, ground conditions, virtual component parameters, etc. We have not attempted to perform an analysis on why the stupid walking algorithm converges to a limit cycle for the appropriate initial conditions. We only speculate that mechanisms similar to those present in McGeer's passive dynamic walker [6] and Mochon and McMahon's balastic walking model [9] are in force.

## 4 Conclusions

Spring Turkey walked continuously through the use of Virtual Model Control techniques. We stress here that we *augmented* the natural dynamics of the robot with simple virtual components, rather than attempted to *cancel* the natural dynamics. In no case did we assume linear dynamics.

The ease of implementing Virtual Model Control is promising. One of the major incentives of developing Virtual Model Control is to make designing robot control algorithms easier and more intuitive. The algorithm designer is given additional incentive to use a Virtual Model Controller since it requires minimal computational resources and is straightforward to derive. One of our goals is to automate this process.

We are hopeful that Virtual Model Control will be useful in producing more robust walking. Future work will focus on developing such algorithms. We are currently designing Spring Flamingo, a bipedal robot similar to Spring Turkey with the addition of feet and actuated ankles. Preliminary simulations using Virtual Model Control have shown that ankles and feet may allow for improved speed control, higher efficiency, and smoother walking.

## Acknowledgements

This work was supported in part by the Office of Naval Research, Grant #N00014-93-1-0333.

## References

[1] E. Dunn and R. Howe. Towards smooth bipedal walking. *IEEE Conference on Robotics and Automation*, 1996.

[2] N. Hogan. Impeadance control: An approach to manipulation: Part i - theory, part ii - implementation, part iii - applications. *J. of Dynamic Systems, Measurement and Control*, 107:1–24, 1985.

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

[4] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *IEEE Journal of Robotics and Automation*, 5(1):90–98, 1986.

[5] O. Khatib. A unified approach for motion and force control of robot manipulators: the operational space formulation. *IEEE Journal of Robotics and Automation*, 3(1):43–53, 1987.

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

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

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

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

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

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

[12] Jerry E. Pratt. Learning virtual model control of a biped walking robot. Unpublished Project Report for MIT's Machine Learning Class (6.858), 1994.

[13] Jerry E. Pratt. Virtual model control of a biped walking robot. Master's thesis, Massachusetts Institute of Technology, August 1995.

[14] M. H. Raibert and J. J. Craig. Hybrid position/force control of manipulators. *Journal of Dynamic Systems, Measurement, and Control*, 102, 1981.

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

[16] K. Salisbury. Active stiffness control of a manipulator in cartesian coordinates. *19th IEEE Conference on Decision and Control*, pages 83–88, Dec. 1980.

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

[18] 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.