①

AD-A231 315

Generalizing the Hop: Object-Level Programming for Legged Motion

by

J. K. Kearney and S. Hansen

DTIC
ELECTE
JAN 23 1991
S B D

DEPARTMENT OF COMPUTER SCIENCE

THE UNIVERSITY OF IOWA • IOWA CITY

47

THE UNIVERSITY OF IOWA
DEPARTMENT OF COMPUTER SCIENCE

List of Technical Reports 1975-

* no longer available.

Generalizing the Hop: Object-level programming for legged motion

J. K. Kearney

S. Hansen

Abstract

This paper presents a model-independent method for controlling a hopping robot. The approach focuses on the interaction between the hopper and its surroundings at points of contact. It is only through reactions at contact points that a hopper can alter its momenta. A generalization of the virtual leg abstraction allows control programs to be expressed as constraints on the external forces and torques acting on the hopper. The relationship between the nature external contacts and motion control strategies is investigated. The results of simulations for a variety of hoppers are presented.

1. Introduction

This paper examines methods for programming locomotion of legged robots. Walking, running, and hopping are distinguished from most conventional robot tasks in several respects. A stationary robot manipulator has a base that provides a fixed reference from which motions of the end-effector can be directed to grab, push, paint, or in some other way interact with an object. Legged robots have no fixed element and intermittent contact with the surrounding environment. Because component masses, velocities, and hence momenta are substantial and critically determine behavior, issues of stability and the generation, transformation, and dissipation of energy are of pre-eminent importance in locomotion. Although speed is important for economical implementation of assembly and finishing tasks performed by stationary robot manipulators, velocities are usually small enough that stability and energy transformation need not be a central focus.

The role of the robot in legged movement tasks is also different from more conventional robot problems. In typical applications, the robot is treated as an agent acting on some object. The focus of the action is the resulting change in the state of the object. Considerable attention has been directed toward eliminating the need for specifying robot motion in robot programs. *Object-level* languages allow the programmer to concentrate on the desired consequences of operations to be performed on the object.[1, 6, 7, 9, 10, 15] In these languages, instructions specify the desired positions of objects or tasks to be performed rather than the motions of the robot needed to achieve these results. In walking, running, and hopping, the robot is both the agent and the object of the action. An object-level program for these tasks should express the desired state of the robot. The details of the action not crucial to the outcome should be suppressed.

This paper investigates methods to derive model-independent control programs for hopping. Formulation of control strategies as device-independent constraints allows control algorithms to be applied to a wide variety of hopper designs. We focus on the interaction between the hopper and its surroundings at points of contact. It is only through reaction forces at contact points that external forces can be applied to alter the motion of the hopper. The nature of contact relations places strict constraints on the dynamics of control problems. Control is organized as a two-step process. Within the bounds of contact constraints, a plan for applying forces and torques to the whole hopper is devised. The use of a generalized, virtual leg permits programs to express the motion of hopping bodies independent of their structure. Secondly, the controllable degrees of freedom are used to achieve the required contact forces.

The approach was tested in simulation using the model-driven dynamic simulator Newton. The experiments demonstrate the utility of simulation for studying physical control problems. Through simulation, a broad range of design parameters can be easily tested to determine the limits of the control strategy.

2. The CMU hopper

In an elegant series of experiments, Marc Raibert and colleagues at Carnegie Mellon University (CMU) designed, built, and tested a variety of legged robots.[11] The basic hopper is illustrated in Figure 1. It consisted of a large chassis on which sensors, valves, actuators, and electronics were mounted. A revolute joint connected the chassis to a telescoping leg. An air cylinder in the leg gave it springiness along its longitudinal axis and provided a means of actuation. A second actuator could pivot the leg with respect to the trunk at a hip joint located near the center of mass of the trunk*. In early designs, a tether linking the hopper to a fixed pivot point restricted motion to a circular path in a horizontal plane, centered at the pivot point. Later versions hopped freely in three-space.

The control algorithms for the hopper were remarkably simple and intuitive. The hopper cycled through two phases. During **stance** phase the foot was in contact with the ground; during **flight** the hopper traveled ballistically through the air. The friction between the foot and floor was sufficient to prevent sliding of the foot during stance. The control of hopping was decomposed into three subproblems:

  (1) Posture Maintenance,

  (2) Hop Height Adjustment, and

  (3) Forward Speed Regulation.

It is immediately apparent that the three kinematic parameters to be controlled outnumbered the two actuators. This problem was overcome by allocating actuators to control variables for particular time periods. A stable posture was maintained by adjusting the hip angle during stance to align the trunk relative to the horizontal. Following first contact with the ground, the leg acted as a passive spring absorbing energy until fully compressed, and then returning energy to the trunk in the remaining portion of the stance interval. The leg actuator injected energy into the system by changing the effective spring length in the latter portion of the stance phase. The height of the next hop was controlled by adjusting the magnitude of the change in spring length.

Forward running speed was not explicitly controlled during stance. To achieve a constant average forward speed, foot position relative to the trunk was adjusted during flight so that there would be no net acceleration over the upcoming stance period. Raibert observed that if the foot is placed such that the forward motion of the hopper will leave its center of mass directly over the foot when the leg spring is maximally compressed, then there will be an odd

---

*The large chassis that is the upper member of the hopper is commonly called the body in the hopping literature. We will reserve the term body to refer to the composite linkage and call the chassis the trunk.

Hip

Trunk

Leg

Our Simulated Hopper

CMU Implemented Hopper

Figure 1. The CMU one-legged hopper and the idealized model used in our simulations.

Trunk

Upper Leg

Lower Leg
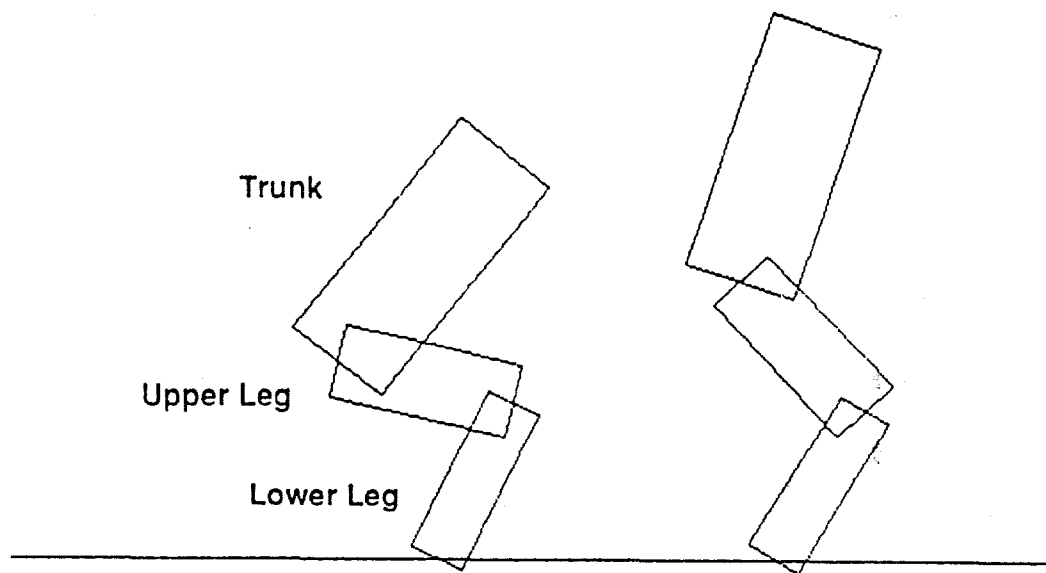
Figure 2. An anthropomorphic hopper. Two frames from a simulated hop are shown.

symmetry in the horizontal forces acting on the body over the stance interval. Backward acceleration in the first half of the stance period will be counterbalanced by forward accelera- tion during the second half period. The position of the foot, relative to the trunk at first contact, which led to this symmetric motion was called the neutral point. The hip actuator was devoted to control of leg orientation during the flight phase. Controlled acceleration and deceleration was achieved by displacing the foot position from the neutral point.

The assignment of actuators to control functions was determined by a state machine. Transitions between states were based upon events triggered by sensory input conditions. The stance phase began when contact with the ground was first sensed. The stance phase ended and the flight phase began when the foot left the ground.

The utility and robustness of the three-part control algorithm were demonstrated by the ease with which the method was extended from two- to three-dimensions and applied to more complex hopping devices. The same basic algorithm was used to control two- and four-legged running machines with a variety of gaits.[4, 12]

3. Form and Function

The design of the CMU hopper simplified control by nearly decoupling important control variables. The mass of the trunk was substantially larger than that of the leg. As a conse- quence, the leg could be rotated relative to the trunk while in flight with minimal counter rota- tion of the trunk. This permitted the leg to be placed for landing with little disturbance of trunk orientation. The design also facilitated control of trunk orientation during stance. The actuator force in the prismatic leg was always directed along the leg axis. The center of mass of the trunk was located at the hip, where the leg axis intersected the trunk. Because of line of action of the leg actuator force passed through the trunk center of mass, leg actuation applied no moment about the mass center of the trunk. Thus, pushing of the leg against the ground caused by the leg spring during stance generated no angular impulse about the trunk mass center and, hence, had no influence on the angular motion of the trunk.

Generalization of the three-part control strategy to a more human-like figure is compli- cated. Consider a three-link body hinged with two revolute joints and moving in a plane. The linkage, pictured in Figure 2, emulates the torso, upper leg, and lower leg of an anthropoid. Upward linear momentum of the anthropomorphic hopper must be generated in a coordinated push by the knee and hip actuators. To jump straight up from a crouched position, the body must unfold. The orientation of the torso cannot be kept constant during this maneuver. There are important advantages to keeping the hopper approximately aligned with the gravity vector. If, during flight, the body rotates too far in either the forward or backward direction, limitations on the range of joint angles can prevent placement of the foot underneath the body and lead to

an uncomfortable collision between the torso and ground. The knee and hip joints dissipate and absorb kinetic energy when the falling body lands. Consequently, it is critical to keep the leg underneath the torso. However, a range of torso orientations can be tolerated.

4. Generalizing the Virtual Leg

The notion of a virtual leg was conceived by Ivan Sutherland to explain how multiple legs can be coordinated as an equivalent single leg. While working with a six-legged walker, Sutherland observed that a set of legs operating in parallel to balance a load behaved as a single leg located midway between them.[13] Raibert exploited the virtual leg abstraction to control multilegged hoppers with the three-part strategy used to control the one-legged hopper. By treating coupled legs as a single, virtual leg, the motion analysis is simplified and control algorithms can be expressed independent of the number of legs and the manner in which they are coordinated. The leg groupings can be changed to generate different foot placement patterns or gaits.

The virtual leg can also be used to generalize the three-part control strategy to hoppers with a variety of structures. We first need to formalize our view of the virtual leg. Consider an arbitrary articulated body with contact on only one external surface at a single point C. Let the hopper's center of mass be positioned at G, and let the vector from G to the contact point be r. The free-body diagram for the hopper is shown in Figure 3. Two external forces act on this body. The weight, mg, acts through the center of mass and is directed downward along the $-\hat{Y}$ axis of the inertial coordinate system. The other force, f_c, represents the reaction force of the ground and acts at the point of contact.

The overall instantaneous motion of the composite body is determined by the sum of external forces and moments acting on the body about G. The resultant force on the body, f_t is the sum the force caused by gravity and the reaction force:

$$m a_G = f_t = mg + f_c.$$ (1)

where a_G is the inertial acceleration of the center of mass. Since the force of gravity passes through the center of mass, the only moment on the body about the mass center is that caused by the reaction force at the point of contact:

$$\dot{H}_G = M_G = r \times f_c.$$ (2)

where,

M_G is the resultant external moment about the center of mass of the composite body and

$\dot{H}_G$ is the time-derivative of the angular momentum of the body about the center of mass, H_G.

$$\mathbf{M}_G = \mathbf{r} \times \mathbf{f}_c$$
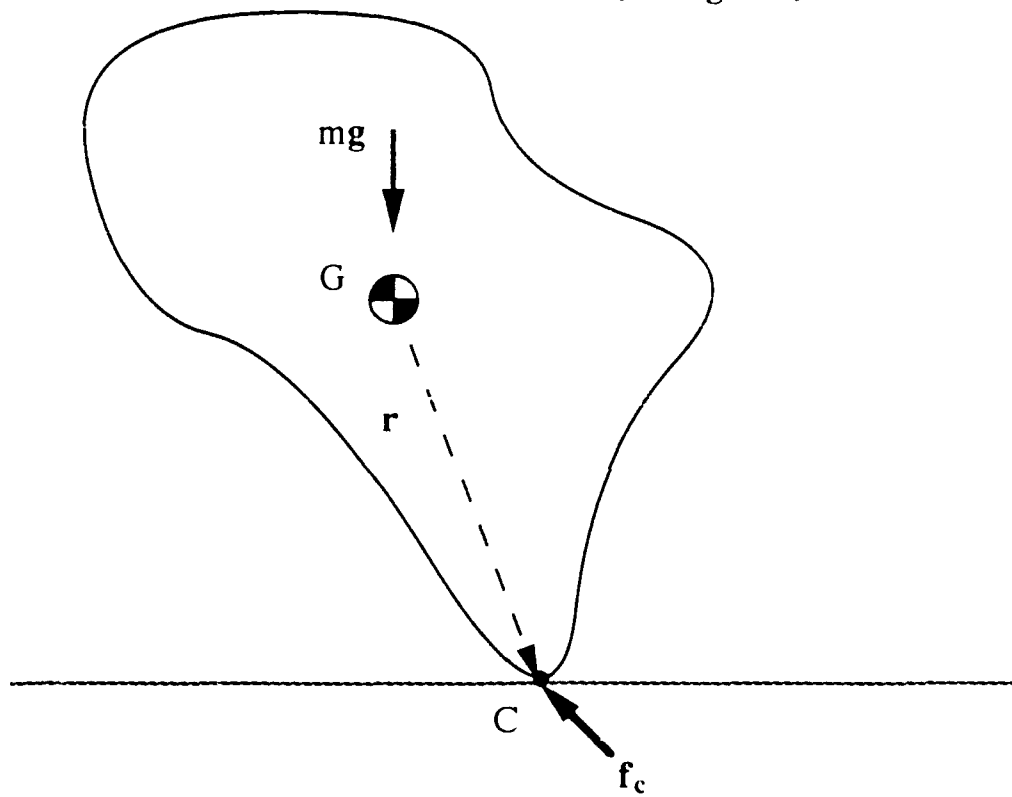
$$\mathbf{f}_t = mg + \mathbf{f}_c$$

Figure 3. The hopper free-body diagram.

The overall instantaneous motion of the composite body during stance is completely determined by (1) and (2).

We will call the vector **r** the virtual leg. The virtual leg determines the relationship between the contact force and moment on the body about the mass center. Note that this relationship is independent of the internal structure of the composite body. The virtual leg allows us to treat an arbitrary body as a lumped mass connected to the ground by a massless leg. By expressing the equations of motion in terms of the contact force, we can ignore the actuator forces and torques that contribute to the reaction force. This permits us to focus on the interaction between the body and its environment and will allow us to express control processes independent of device actuation or body structure.

5. Degrees of Freedom

Consideration of the virtual leg demonstrates intrinsic limitations in controlling a single contact hopper. Irrespective of the actual hopper structure, number of legs, and number of actuators, a single point of contact permits control of only a single, external force on the body. The contact force and gravity determine both the resultant external force and the resultant external moment on the body. Thus, for a planar hopper, only two of the three degrees of freedom of instantaneous motion can be independently controlled. A similar analysis applies to the three-dimensional case.

The coupling between the resultant external force and moment is a function of the contact relationship between the body and its environment, and this relationship holds for all single contact situations. For example, an astronaut pushing with a single finger against a wall in zero gravity faces the same problem.

When considering the overall motion of the body, a planar hopping device with three or more actuators is redundant. For all non-singular configurations, there will be an infinite number of actuator joint torque/force combinations that lead to the same contact force.

It is interesting to consider how body motion might be controlled when there are more complex contact relations between the body and the environment. We will defer consideration of higher order contacts and multiple points of contact until after a discussion of control strategies directed at the single point contact problem.

6. Programming Contact Forces for the Stance Period of a Hop

The generalized virtual leg introduced above allows us to separate the control algorithm for the stance period of a hop from the form of the hopper. We propose a model-based strategy that partitions control into a model independent specification of desired contact forces,

followed by a model dependent determination of joint torques and forces. This two-step decomposition is analogous to the computed-torque method used to control the motion of robot arms.[2] In the first step, the ground reaction force required to achieve the desired motion of the body is determined. This calculation depends only on the state of the composite body. The structure and actuation of the device is irrelevant. In the second step, the actuator values necessary to produce the required reaction force are calculated.

The contact force is decomposed into components parallel and orthogonal to the virtual leg. We introduce constraint equations to control the two components of the contact force. Recall that r is the vector from the center of mass of the body to the contact point. The component of the contact force parallel to r is constrained to behave as a linear spring along the leg axis. Hence, the spring force may be expressed as

$$\mathbf{f}_c \cdot \frac{\mathbf{r}}{\|\mathbf{r}\|} = k_l(\|\mathbf{r}\| - r_s) \tag{3}$$

where $\|\cdot\|$ is the vector norm operator, k_l denotes the spring stiffness, and the scalar r_s is the resting length of the spring.

The component of the contact force perpendicular to the virtual leg is used to control the resultant moment about G acting on the body. The value of this moment is constrained to reduce the difference between the angular momentum of the body about its mass center, $\mathbf{H}_G$, and the desired angular momentum of the body about its mass center, $\bar{\mathbf{H}}_G$:

$$\mathbf{M}_G = \mathbf{r} \times \mathbf{f}_c = k_h(\bar{\mathbf{H}}_G - \mathbf{H}_G). \tag{4}$$

The constant of proportionality, k_h, determines the rate at which corrections are made. The angular momentum about the mass center of an articulated body consisting of n rigid links is defined by

$$\mathbf{H}_o = \sum_{i=1}^{n} \left[\mathbf{I}_i \omega_i + (\mathbf{c}_i \times m_i \mathbf{v}_i) \right] \tag{5}$$

where

$\mathbf{I}_i$ is the inertia tensor of link i about its center of mass,

ω_i is the angular velocity of link i,

m_i is the mass of link i,

v_i is the velocity of the center of mass of link i, and

c_i is the vector from the center of mass of the composite body to the center of mass of link i.

The two constraint equations (3) and (4) have a simple geometric interpretation. The component of force along the virtual leg acts as a spring pushing the body away from the ground. The component orthogonal to the virtual leg moderates the direction of the push and applies a torque to the body about the mass center. If the pushing is directed to the left of the center of mass in Figure 3, a clockwise torque is applied to the body. In normal hopping, it is desirable to have little or no angular momentum about the center of mass at take-off. For diving or flipping, a controlled amount of angular momentum can be introduced by specifying a nonzero value for $\bar{H}_G$.

7. Actuator Torque and Force

Having determined the desired contact force, the model dependent actuator torques and forces required to generate the reaction force must be computed. Given a model of the object and its contacts, we can formulate a system of linear equations that govern its dynamic behavior. Constraint forces are of primary interest. ˙ ˙ause the Newton-Euler method explicitly includes these terms, it is used. In the usual derivation, constraint forces are treated as unknowns and eliminated from the dynamic equations. For our problem, the motion equations are augmented with the additional control constraints on the contact force expressed in Equations (3) and (4). With these additional constraints, we can solve for joint torques and forces.

In our hopping experiments, the number of controllable joint torques was always equal to the number of control constraints introduced on contact forces. Therefore, a unique solution existed whenever the system of equations was non-singular. For a more complicated hopper with greater articulation, the internal degrees of freedom may outnumber the external constraints. In this case, a redundancy exists and the system will have an infinite number of solutions. There are a number of ways to overcome this problem. Internal constraints can be introduced or a minimization technique such as the pseudo-inverse method can be used to find a solution.

A more serious problem arises if the augmented system of motion equations is inconsistent. When this happens, the combination of mechanical and control constraints cannot be satisfied by any set of actuator forces and torques. Recognition and avoidance of inconsistent sets of constraint equations is a difficult problem for which there is no general solution. The problem is compounded when limits on joint range of motion are incorporated in the model. We did not attempt to devise a general method to avoid, detect, or resolve inconsistencies.

Instead, we employed heuristic rules to prevent inconsistencies from arising. For example, the anthropomorphic hopper can apply no vertical force on the center of mass when standing perfectly erect. This and similar singular configurations were avoided by control routines.

8. Simulation Experiments

We tested the hopping method in simulation using the model-driven, dynamic simulator Newton.[3,5] The primitive objects represented in Newton are rigid solids. Composite objects consist of sets of primitive objects hinged together. A hinge constrains the relative motion of two objects. For example, a point on one object may be constrained to be coincident with a point on another object to simulate a ball-and-socket hinge. A hinge may be temporary. For example, it might only exist while the contact force between two objects is compressive. This permits supporting relations and compliant motion to be modeled.

Newton maintains schema for the equations of motion of each primitive object. When two primitive objects are hinged together, a geometric constraint is placed on their locations and/or relative motion. Consequently, the dynamic equations of each object must be modified to be consistent with the additional kinematic relationship. The system includes a libraiy of predefined hinges for which equation editing is automatically performed. The hinge library includes ball-and-socket, pin, and prismatic hinges. A modeling language provides a convenient interface for specifying object geometry, interconnection, and initial conditions.

Newton also supports equation editing at the level of user defined functions. The set o. dynamic equations derived by the system can be augmented by differential equations constraining instantaneous quantities. In this way, all dynamic constraints on the object are treated in a homogeneous manner and solved simultaneously. There is no distinction between constraints arising from mechanical considerations and constraints due to control. The system of equations can include arbitrary linear constraints on accelerations, forces, and torques, and it may include functions of state variables. Equations may be later retracted. A unique name must be assigned to each equation when it is created to permit reference to it. Constraints may be expressed as vector or scalar equations. While it is beyond the scope of this paper, it should be mentioned that when defining new constraint equations, some care must be taken to avoid violation of continuity assumptions in the integration method.

A simulation proceeds by iteratively integrating the augmented set of motion equations. The basic simulation loop is outlined in Figure 4.

In preparation for the first integration cycle, an initial state is established. Integration proceeds from the initial conditions by iteratively solving the motion equations and integrating to establish a new state. After each integration step, the resulting state is tested for the

```
(initialize-state)
(until (stop.simulation-p) do
          (integrate.from.current.state)
          (handle.exceptional.events)
          (establish.complete.state)
          (report)
          )
```

Figure 4. The basic simulation loop.

occurrence of exceptional events. A list of active events is maintained. Associated with each event is a condition which signals the event and a method of resolution. For example, the hopper landing is detected by penetration of the foot into the ground. The landing is resolved by solving the impact equations for the collision between the hopper and ground, changing positions and velocities in accordance with the result, and creating a temporary hinge between the foot and ground at the point of first contact. An event is also created to monitor the contact force between the foot and ground. If, on some later iteration, the contact force becomes tensile, then the hinge is broken, the motion equations are re-solved, and the hopper lifts off the ground.

The last step in the loop is to generate a report of the current state. The report can be presented in a number of forms including a pictorial rendering of the geometric model, graphs, or printed values of state variables.

8.1. Simulating the CMU Hopper

To provide a basis of comparison, we programmed a planar model based on the CMU one-legged hopper using the control scheme described by Raibert. The model is pictured in Figure 1. It consists of a relatively massive trunk connected to an upper leg by a revolute joint. The upper leg is connected to the lower leg by a prismatic joint. The CMU hopper has a mechanical stop that limits the extension of the lower leg relative to the upper leg. The action of the stop is modeled by an inelastic collision between surfaces on the top of the lower leg and bottom of the upper leg when the maximal extension is reached. Values of model parameters are presented in Table 1.

To simulate the actuators, we assigned values to the leg joint force and hip joint torque. In place of sensory dependent triggers, event monitors inspected the values of state variables.

Property	symbol	Value
Trunk Mass	M_t	$10\ kg$
Upper Leg Mass	M_u	$0.5\ kg$
Lower Leg Mass	M_l	$0.7\ kg$
Trunk Moment of Inertia	J_t	$3.335\ kgm^2$
Upper Leg Moment of Inertia	J_u	$0.0093\ kgm^2$
Lower Leg Moment of Inertia	J_l	$0.0863\ kgm^2$
Spring Stiffness	k_l	5000

Table 1.   Model parameters for the CMU-like hopper.

Trunk posture was maintained by applying a torque at the hip joint during stance. We found that the timing of torque application was critical. Once the hopper passes vertical, the leg is behind the center of mass. During this period, hip torques tend to raise the leg. Large torques can cause the leg to lift off the ground prematurely. Because lift-off results in the termination of other control processes, it can have serious consequences.

Hopping height was controlled by lengthening the leg spring. Energy must be inserted into the system to compensate for the energy absorbed by the ground on the foot's impact. In the CMU simulations, spring length was changed discontinuously in the latter stages of the stance period. Recently, Sznaier and Damborg reported that[14] a more complete decoupling of leg actuation and forward velocity can be achieved if the actuator force is varied linearly over the time of stance. In all the experiments reported here, spring length was increased linearly over the stance period.

Forward velocity was controlled by adjusting the hip joint angle during flight to position the foot for touchdown. The neutral point was calculated using the approximation introduced by Raibert.[11] This method estimates the motion of the body center of mass during stance by assuming that velocity is constant while in contact with ground and equal to the initial velocity at touch down. The time of stance is determined by the period of the leg spring. The neutral point is approximated by the mid-point of the estimated path of the center of mass projected onto the supporting surface.

We encountered some difficulty in establishing and maintaining a desired forward velocity. In our experiments, we found that small variations in foot placement caused sizable
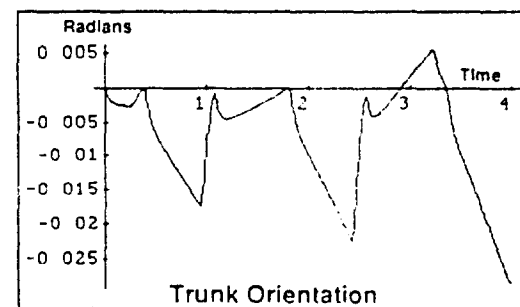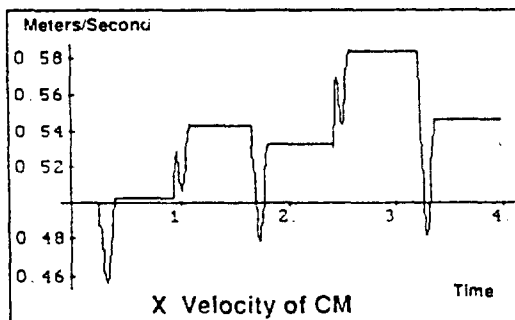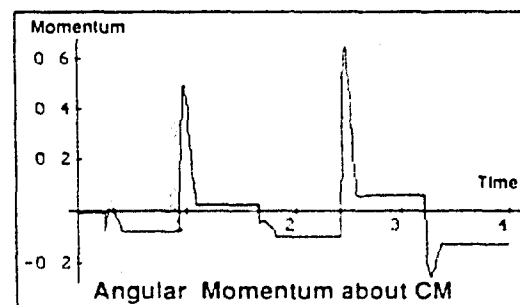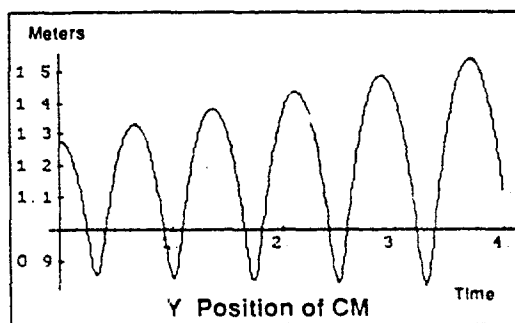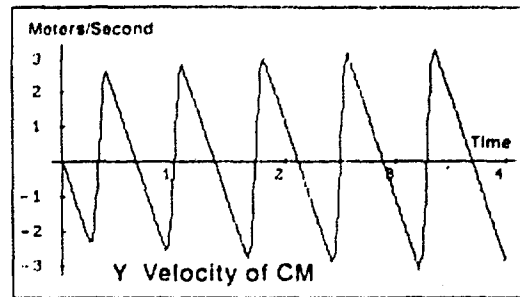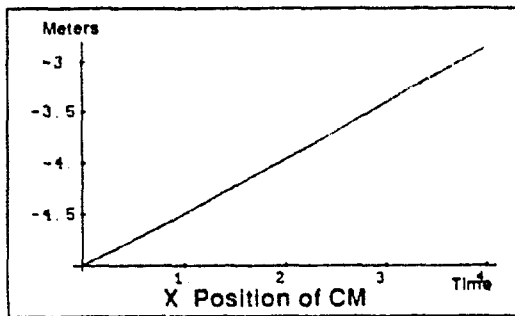
Figure 5. Simulation results for the CMU hopper. In this experiment, the actuator values were specified using the three-part control strategy.

variations in mass center forward velocity. We are currently investigating alternative methods to estimate the neutral point. Since accurate control of forward velocity was not a primary concern for the work presented, the simple mid-point calculation was used.

The time-histories of several important state variables are graphed for a representative series of hops in Figure 5. The data correspond well with results reported in experiments with versions of the CMU hopper.

We also programmed the 2-D model of the CMU hopper using constraints on the contact force. Adapting the model to a control strategy based on contract force constraints proved remarkably easy. As with the CMU approach, we controlled forward velocity by applying a torque at the hip to adjust the foot position during flight. During stance, joint actuator specifications were replaced by constraints on the ground reaction force expressed in Equations (3) and (4).

Graphs of state variables for a typical series of hops using contact force control are shown in Figure 6. The hopper motion is similar under the two control regimes. The principal difference arises from the manner in which angular motion is handled by the two techniques. The CMU method maintains posture by applying a torque at the hip joint to control the angle of the torso. Our method constrains the moment on the composite body about its center of mass to control angular momentum. Because torso posture is not explicitly controlled, the orientation of the torso can drift from the horizontal. The collision of the foot with the ground at touch-down creates an impact force that causes an abrupt change in angular momentum. This momentum is systematically nulled during stance. Since the direction of angular momentum is same the throughout the period, the orientation of the body will change. The magnitude of this change is normally small. However, for a long series of hops in the same direction, it may be necessary to compensate for the accumulated rotation of the body.

8.2. Simulating the Anthropomorphic Hopper

We programmed an anthropomorphic hopper to demonstrate the versatility of the contact force programming strategy. The anthropomorphic model consisted of three links hinged end to end with revolute joints and moving in the plane. The model was intended to approximate the torso, upper leg, and lower leg of an anthropoid. Model parameters are presented in Table 2.

The anthropomorphic hopper was controlled with the same contact force constraints used for the CMU hopper. During stance, leg springiness and angular momentum about the mass center were controlled by constraints on the ground reaction force expressed in Equations (3) and (4). Only the constants of proportionality in Equations (3) and (4) were changed. As in the
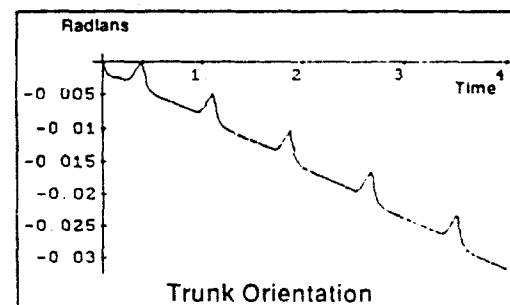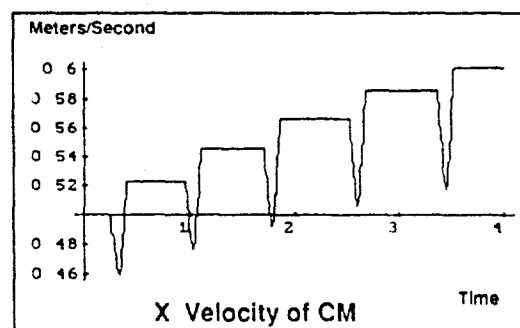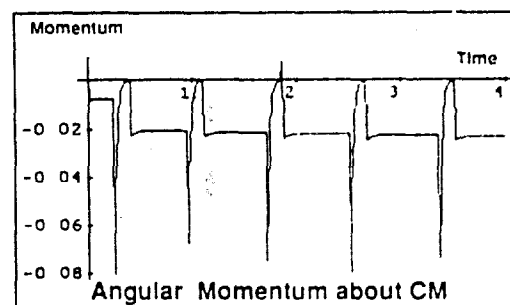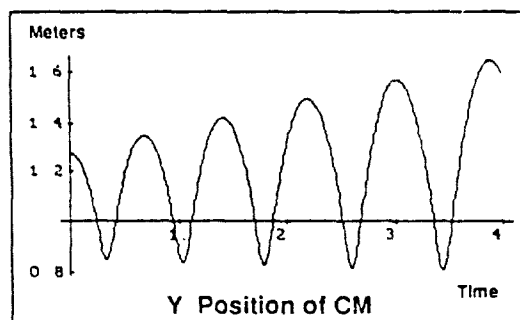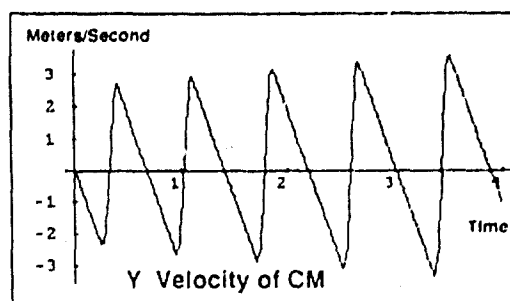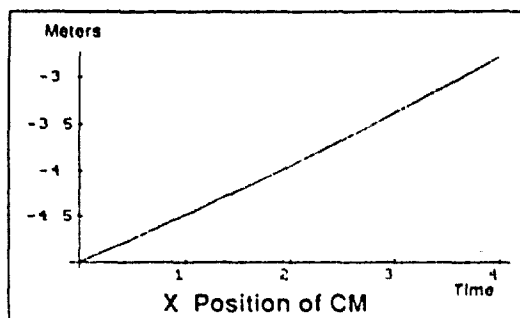
Figure 6. Simulation results for the CMU hopper. In this experiment, the hopper was controlled by contact force constraints during the stance period.
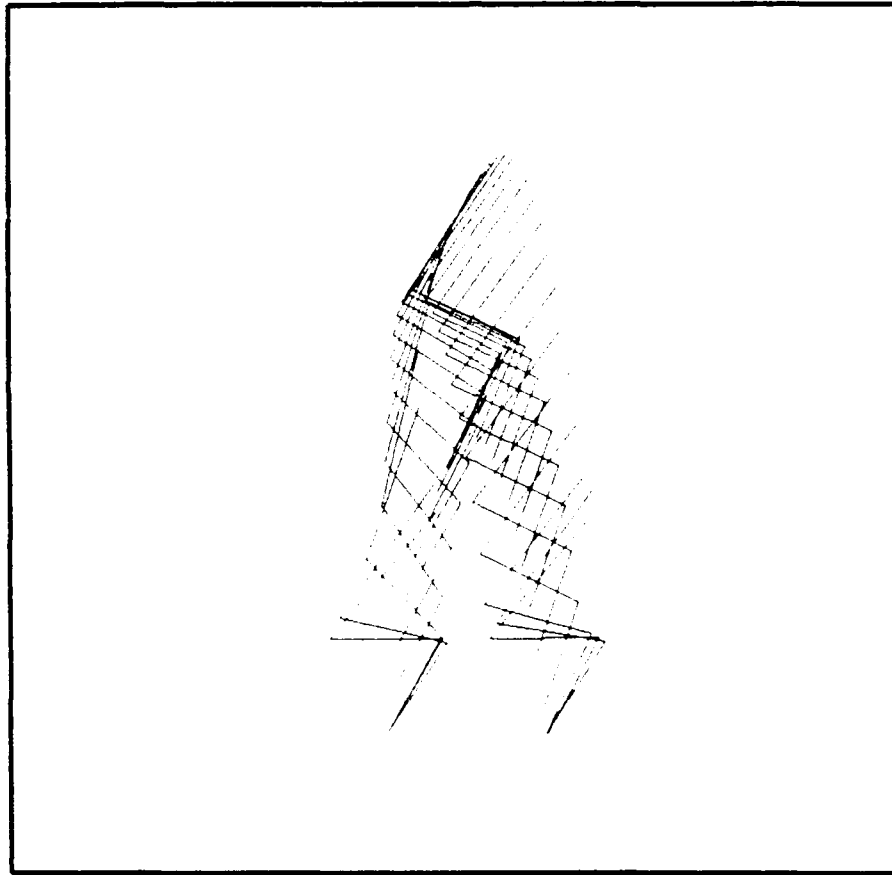
Figure 7. An overlayed sequence of frames from a single hop with the anthropomorphic hopper. The longitudinal axis of each link is drawn.

Property	symbol	Value
Trunk Mass	M_t	45.0 kg
Lower Leg Mass	M_l	13.3 kg
Upper Leg Mass	M_u	10.0 kg
Trunk Moment of Inertia	J_t	1.584 kgm^2
Upper Leg Moment of Inertia	J_u	0.208 kgm^2
Lower Leg Moment of Inertia	J_l	0.146 kgm^2
Spring Stiffness	k_l	35000

Table 2.   Model parameters for the anthropomorphic hopper.

previous experiments, the spring was gradually lengthened over the stance period to inject energy into the system.

During flight, the hip was held at a fixed angle and the knee joint was rotated to position the foot with respect to the center of mass for landing. The neutral point for foot placement was determined by the approximation method used for the CMU hopper.

A sequence of frames from an animated representation of the motion are overlayed in Figure 7. Time-history profiles corresponding to those presented for the CMU hopper are graphed in Figure 8. The graphs bear a marked resemblance to those in Figure 6.

As with the CMU model, impact forces introduced a bias that caused the orientation of the trunk to drift. Because the trunk rotates significantly throughout the hopping motion, the drift appears as a gradual shift in the cyclic bending and unbending of the trunk. If the hopper moved in the same direction for a series of hops, it would soon rotate too far to permit correct foot placement. To correct this bias, we modified the moment constraint equation (4) to include a term for the accumulated error in angular momentum. The modified constraint equation caused both the angular momentum about the center of mass and its integral to be nulled:

$$M_G = r \times f_c = -k_h H_G - k_i \int H_G. \tag{6}$$

State variable graphs from a simulation using the modified angular momentum constraint are shown in Figure 9. The orientation curve is significantly more stable when the correction for angular momentum is introduced.
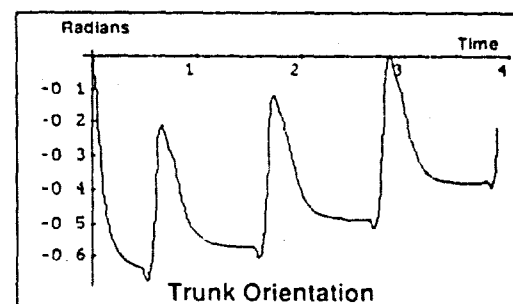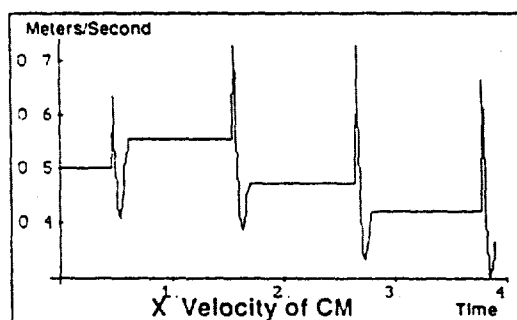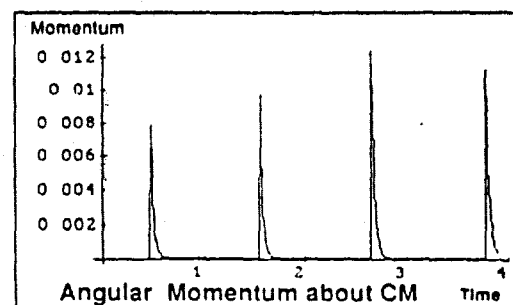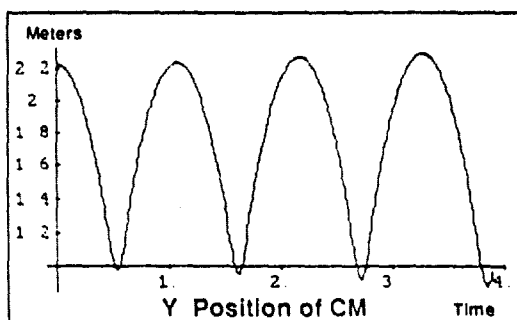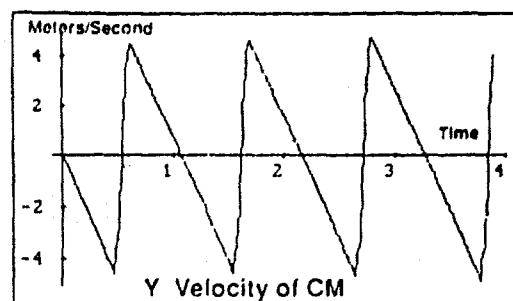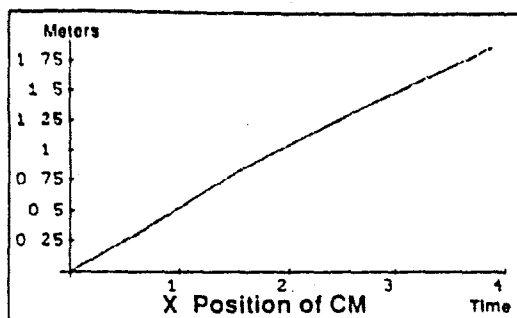
Figure 8. Simulation results for the anthropomorphic hopper. In this experiment, the hopper was controlled by contact force constraints during the stance period.
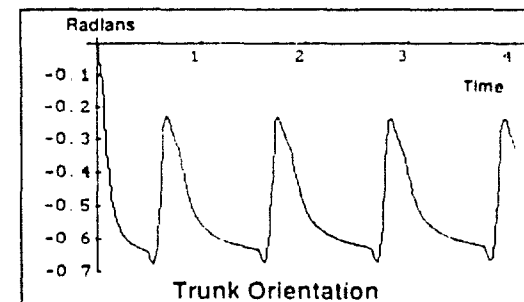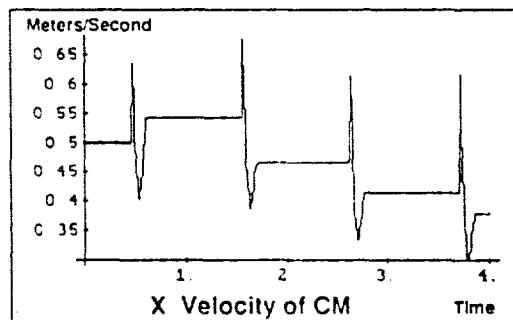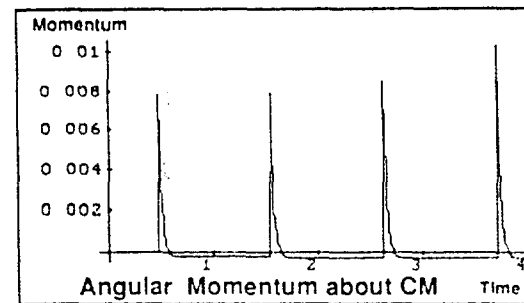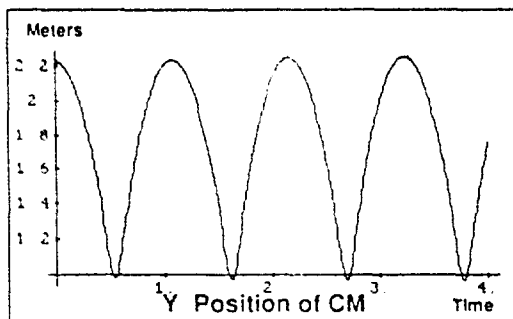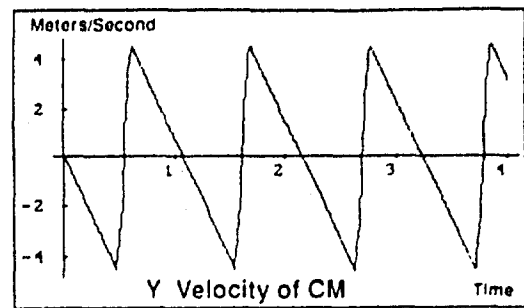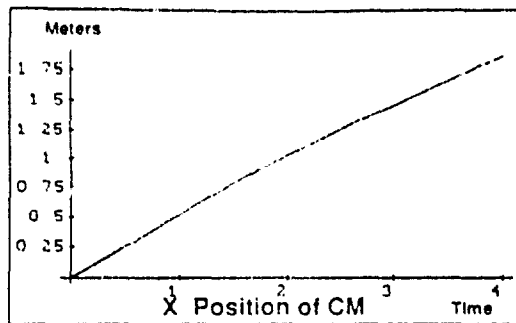
Figure 9. Simulation results for the anthropomorphic hopper. In this experiment, a correction was added to the angular moment constraint equation to eliminate accumulated angular momentum.

9. Contact Programming with Multiple Contacts

More complicated contact relations between an articulated object and the environment can be exploited in a variety of interesting ways. We examine two familiar problems in which, for a period of time, an articulated figure has a richer interaction with the surrounding environment.

9.1. The Flat-footed Jump

The work presented to this point has modeled the hopper as having a single point of contact with the ground during stance. This approximates the problem of hopping on tip-toes or on a pogo stick. For many human activities, the foot provides a surface of contact with the ground. The distribution of forces over the surface of contact may have a strong influence on the motion of the body. The importance of pressure sensing on the foot for balance has been noted in studies of human posture control.[8]

Muscles and tendons crossing the lower extremity joints allow control torques to be exerted on the leg. When the foot is affixed to the floor, these torques cause external torques about the body's mass center to be exerted. Thus, the 2-D, flat-footed hopper has control of three degrees of motion freedom of the body. The additional degree of freedom caused by surface contact between the foot and ground permits a whole new range of control strategies. The force and moment on the body can, in principle, be independently controlled at all times. We are cautious to avoid statements about the biological significance of this observation. We only emphasize that more complex interconnections between the body and the environment allow more flexible control methods. An articulated body in surface contact with the environment is intrinsically more controllable than a body in point contact.

As an example of the increased controllability, we programmed the anthropomorphic hopper to jump straight upward and flip. Three constraint equations governed the behavior of the hopper while in contact with the ground. As in the previous hopping experiments, the component of the contact force parallel to the virtual leg was constrained to act as a spring by Equation (3). The component of the contact force perpendicular to the virtual leg was constrained to be zero:

$$\frac{r}{\| r \|} \times f_c = 0. \tag{5}$$

Thus, the contact force was constrained to always pass through the center of mass. When the hopper was dropped from a point above the ground with the foot directly under the center of mass, no force would be applied in the horizontal direction and the center of mass could only move parallel to the vertical.

The resultant external moment about the center of mass is equal to by the sum the moment caused by a contact force f_c and the contact torque τ_c applied at the contact point C:

$$M_G = r \times f_c + \tau_c \qquad (6)$$

The inclusion of a contact torque τ_c allows us to introduce an independent constraint on the external moment about the center of mass. During the later stages of the upward hopping motion, we define a proportional constraint on the resultant mass center moment to introduce angular momentum for the flip.

$$M_G = k_h (\bar{H}_G - H_G). \qquad (7)$$

In our experiments the amount of rotational momentum required to perform a flip was determined experimentally. Frames from an animation of a flip are shown in Figure 10. Graphs of state variables for a single flip are shown in Figure 11. The graphs show that a significant increase in angular angular momentum can be achieved with no concomitant change in forward velocity.

9.2. Standing from a Sitting Position

The linkage shown in Figure 12 is meant to represent an idealization of a human body sitting on a chair. The body is in contact with the external world at two points; the foot is touching the floor and the hip joint is in contact with the seat of the chair. Both of these hinges are one-directional -- they will support only compressive constraint forces. The hinge will break and the contact points will separate if the constraint force is ever tensile.

We would like the body to rise from the chair and move into an erect configuration. If arbitrary torques could be applied at the hip, knee, and ankle joints, then the body could stand simply by assigning a joint position controller to each joint actuator with the desired angles set for an upright posture. Unfortunately, it is unreasonable to expect that sufficient torque could be exerted at the knee to implement this motion in a person. Interestingly, this motion, when viewed in animation, appears quite unnatural.

As an alternative strategy, we present a method that exploits the nature of the contact relations by first generating clockwise angular momentum in the torso, and then rotating the composite body's center of mass about the ankle to move it to a position above the ankle. The motion is decomposed into two phases. During the sitting phase, the hip is in contact with the chair. The attachment serves to dynamically isolate the torso from the rest of the body. We treat the torso as an independent body while the figure is sitting. A control torque is applied to the torso at the hip joint to generate clockwise angular momentum. Once substantial angular momentum has been generated, the hip joint is locked. The resulting forward linear momentum of the body will cause contact with the chair to be broken and the second phase begins.

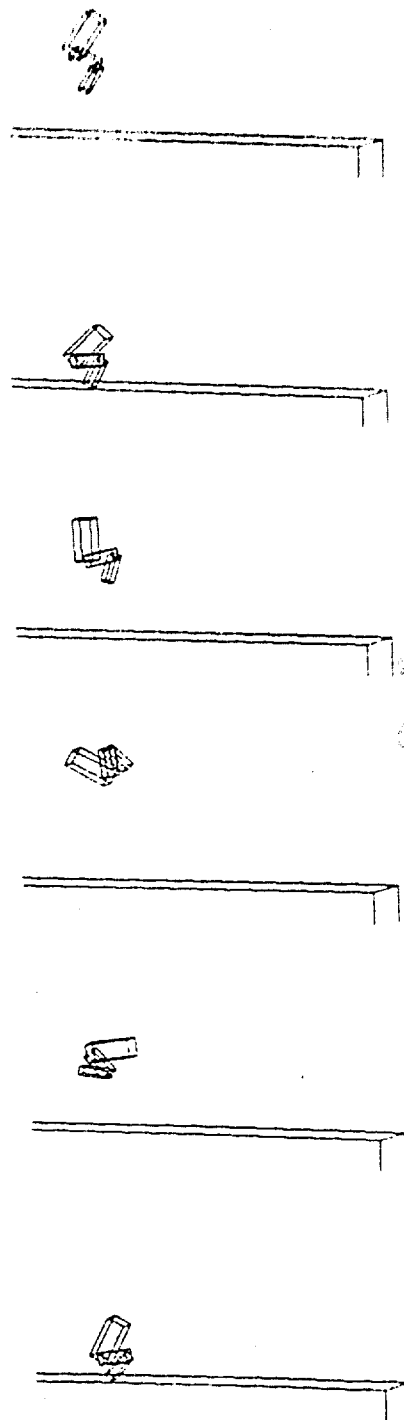**Figure 10.** The anthropomorphic hopper performing a flip. During stance the hopper was in surface contact with the ground.
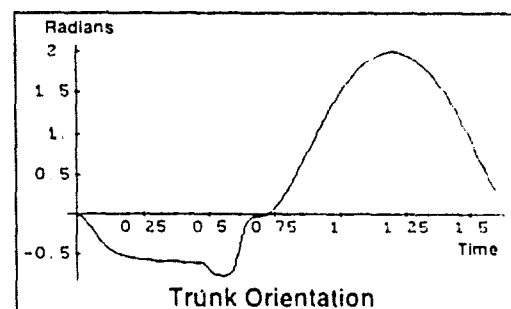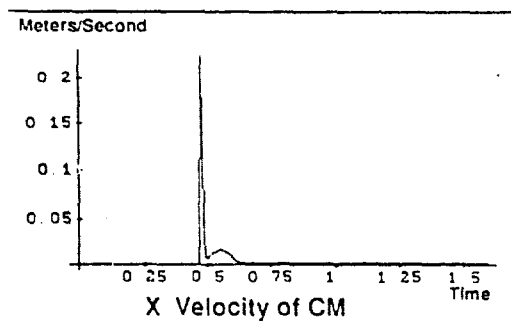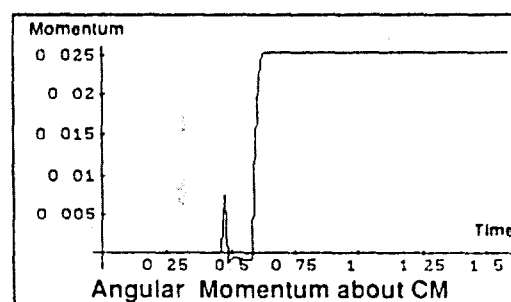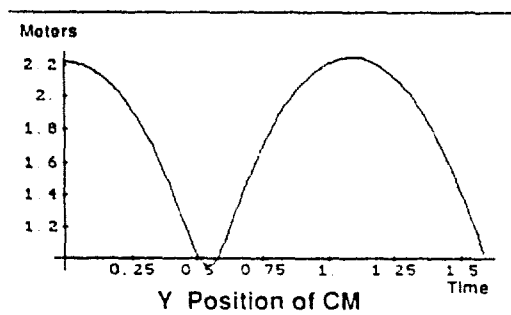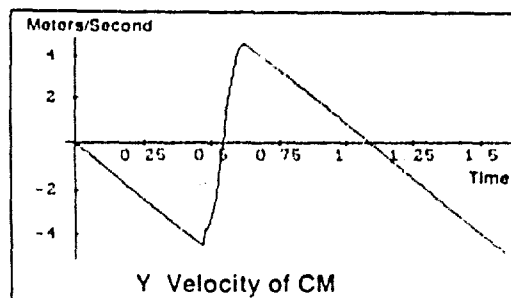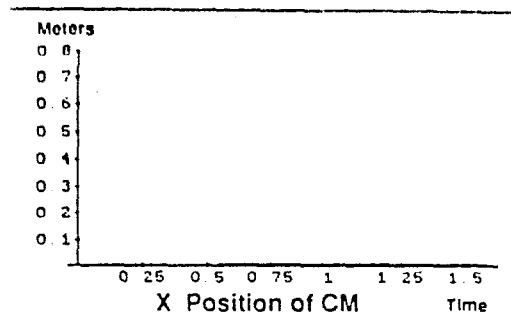
Figure 11. Simulation results for the anthropomorphic hopper in surface contact with the ground. The surface contact enables forward velocity and angular momentum to be decoupled. This allows to hopper to jump straight upward with sufficient angular momentum to flip.

Figure 12. A sequence of frames from a simulation that stands from a sitting position. The sequence begins in the upper left corner and ends in the lower right corner.

In the second phase of standing, the knee and hip joints are used to move the figure into an erect position. A torque applied at the floor contact is used to gradually reduce the rotation of the body, leaving it standing in a stable, upright potion. A sequence of frames from a program for sit-to-stand using this method is shown in Figure 12.

10. Discussion

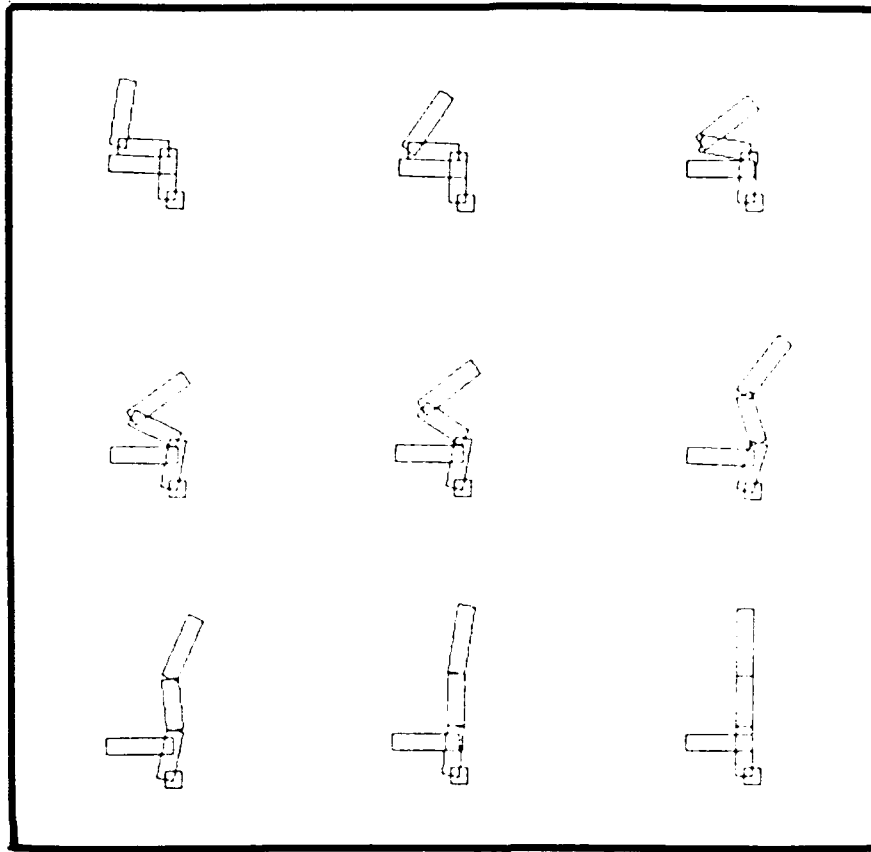The experiments presented illustrate the richness of expression and explanatory power of the approach. By focusing on the interaction between the robot and the environment, general purpose, device independent strategies for controlling movement can be developed. The resulting programs specify dynamic constraints on the external forces and torques applied to the body. The internal structure and actuation of the device are suppressed to lower levels of processing.

Major problems remain to be solved. A recurrent experience during early simulation experiments was that it is easy to mistakenly specify inconsistent constraints. Determining the source of the inconsistency and correctly formulating the intended constraint were usually very difficult. Singularities were a persistent nuisance for mechanisms as simple as the planar, three link hoppers. Robust methods to detect and avoid singularities would considerably ease the burden of the programmer and generalize the results. Lastly, we have only begun to investigate redundancy resolution for situations where the number of independent actuators is greater than the constraints on the device. A strength of the approach is that it delineates singularities and redundancy as potent but separate problems from the object-level motion constraints.

References

1. A. Ambler, S. A. Cameron, and D. F. Comer, "Augmenting the RAPT robot language," in *Languages for Sensor-based Control in Robotics*, ed. Hormann, pp. 305-316, Springer-Verlag, New York, NY, 1987.

2. J.J. Craig, *Introduction to Robotics*, Addison-Wesley Publishing Company, Reading, MA, 1986.

3. J. Cremer, "An Architecture for General Purpose Physical Simulation -- Integrating Geometry, Dynamics, and Control," Ph.D. Thesis, TR 89-987, Cornell University, April, 1989.

4. J. Hodgins, J. Koechling, and M. H. Raibert, "Running experiments with a planar biped," in *The 3rd International Symposium on Robotics Research*, ed. M. Ghallab, The MIT Press, Cambridge, MA, 1985.

5. C. M. Hoffmann and J. E. Hopcroft, "Simulation of physical systems from geometric models," *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 3, pp. 194-206, June 1987.

6. J. C. Latombe, C. Laugier, J. M. Lefebvre, E. Mazer, and J. F. Miribel, "The LM robot programming system," in *Robotics Research - The Second International Symposium*, ed. H. Inoue , pp. 377-391, MIT Press, Cambridge, MA, 1985 .

7. T. Lozano-Perez, "Robot Programming," *Proceedings of the IEEE*, vol. 71, no. 7, July, 1983.

8. L. Nashner and G. McCollum, "The organization of human postural movements: A formal basis and experimental synthesis," *The Behavioral and Brain Sciences*, vol. 8, pp. 132-172, 1985.

9. R. Popplestone, A Ambler, and I. Bellos, "An interpreter for a language for describing assemblies," *Artificial Intelligence*, vol. 14, pp. 79-107, 1980.

10. R. Popplestone and A. Ambler, "A language for specifying robot manipulations," in *Robotic Technology*, ed. A. Pugh, pp. 125-141, Peter Peregrinus, London, 1983.

11. M. H. Raibert, *Legged Robots That Balance,* The MIT Press, Cambridge, MA, 1986.

12. M. H. Raibert, M. Chepponis, and H. B. Brown, Jr., "Running on four legs as though they were one," *IEEE Journal of Robotcs Research*, vol. 2, 1986.

13. I. E. Sutherland and M. K. Ullner, "Footprints in the asphalt," *International Journal of Robotics Research*, vol. 3, pp. 29-36, 1984.

14. M. Sznaier and M. J. Damborg, "An Adaptive Conroller for a One-Legged Mobile Robot," *IEEE transactions on Robotics and Automation*, vol. 5, no. 2, pp. 253-259, April, 1989.

15. R. Volz, "Report of the robot programming language working group: NATO workshop on robot programming languages," *IEEE Journal of Robotics and Automation*, vol. 4 , no. 1, pp. 86-90, Feb 1988.

Acknowledgements

83-10* D. A. Eichmann, *A preprocessor approach to separate compilation in Ada.*
83-11* R. K. Shultz, *Simulation of multiprocessor computer architectures using ACL.*
84-01   D. H. Freidel, *Modelling communication and synchronization in parallel programming languages.*
84-02* T. Rus and F. B. Herr, *An algebraic directed compiler generator.*
84-03   M. E. Wagner, *Performance evaluation of abstract data type language implementations.*
84-04* R. Ford and M. Wagner, *Performance evaluation methodologies for abstract data type implementation techniques.*
84-05   K. Brinck, *The expected performance of traversal algorithms in binary trees.*
84-06   K. Brinck, *On deletion in threaded binary trees.*
84-07   K. Siu-ming Yu, *A testbed database generator.*
84-08   D. Sawmiphakdi, *A multiprocess design for an integrated programming environment.*
84-09   D. W. Jones, *Machine independent SMAL: a symbolic macro assembly language.*
84-10* R. K. Shultz, *Comparison of database operations on a multiprocessor computer architecture.*
84-11   J. H. Kingston, *A new proof of the Garsia-Wachs algorithm.*
85-01   T. Rus, *Fast pattern matching in strings.*
85-02   T. Rus, *An inductive approach for program evaluation.*
85-03   G. S. Singer, *Extensions to the Iowa logic specification language.*
85-04   A. C. Fleck, *Babble reference manual.*
85-05   K. Brinck, *Computing parent nodes in threaded binary trees.*
85-06   J. H. Kingston, *The amortized complexity of Henriksen's algorithm.*
85-07* R. Ford, M. J. Jipping, and R. Shultz, *On the performance of an optimistic concurrent tree algorithm.*
85-08* D. W. Jones, *Iowa capability architecture project ICAP programmer's preference manual.*
85-09* S. P. Miller, *Automated instrumentation of communication protocols for testing and evaluation.*
86-01   M. J. Jipping, *An information-based methodology for the design of concurrent systems.*
86-02* H. I. Mathkour, *An extended abstract data type specification mechanism.*
86-03   D. E. Glover, *Experimentation with an adaptive search strategy for solving a keyboard design configuration problem.*
86-04   W. Pan, *Designing an operating system kernel based on concurrent garbage collection.*
86-05   B. C. Wenhardt, *A comparison of concurrency control algorithms for distributed data access.*
86-06   R. Shultz and I. Miller, *An execution cost analysis of multiple processor join methods.*
86-07   R. Shultz, *Controlling testbed database characteristics.*
86-08   R. E. Gantenbein, *Dynamic binding of separately compiled objects under program control.*
86-09   M. Pfreundschuh and R. Ford, *A model for modular system builds based on attribute grammars.*
86-10   R. Shultz and I. Miller, *Memory capacity in multiple processor joins.*
86-11* M. P. Pfreundschuh, *A model for building modular systems based on attribute grammars.*
87-01* M. J. Kean, *A communications architecture for distributed applications comprised of broadcasting sequential processes.*
87-02* B. A. Julstrom, *A model of mental image generation and manipulation.*
87-03* M. J. Jipping and R. Ford, *An information-based model for concurrency control.*
87-04* Ravi Mukkamala, *Design of partially replicated distributed database systems: an integrated methodology.*
87-05* A. C. Fleck, *A case study comparison of four declarative programming languages.*
88-01   Jing Jan, *Data abstraction in the Iowa logic specification language.*
88-02* J. P. Le Peau and T. Rus, *Interactive parser construction.*
88-03* J. Gilles, *A window oriented debugging environment for embedded real time ada systems.*
88-04   S. R. Sataluri and A. C. Fleck, *Incremental development of semantics using relational attribute grammars.*
88-05   S. R. Sataluri, *Generalizing semantic rules of attribute grammars using logic programs.*
88-06   Hantao Zhang, *Reduction, superposition and induction: Automated reasoning in an equational logic.*
89-01   C. M. Gessner, *A case study in post-developmental testing.*
89-02   Ken Slonneger, *Denotional semantics in prolog.*
89-03   Deepak Kapur and Hantao Zhang, *RRL: Rewrite rule laboratory user's manual.*
89-04   Hantao Zhang, *Prove ring commutativity problems by algebraic methods.*
89-05   David Allen Eichmann: *Polymorphic extensions to the relational model.*
89-06   In jeong Chung, *Improved control strategy for parallel logic programming.*
89-07   Po-zung Chen, *New directions on stochastic timed petri nets.*
89-08   Frank William Miller, *A predictive real-time scheduling algorithm.*
90-01   Teodor Rus, *Algebraic construction of a compiler.*
90-02   Sukumar Ghosh, *Understanding self-stabilization in distributed systems, part I.*
90-03   Ching-Ming Chao, *A rapid prototyping methodology for conceptual database design using the executable semantic data model.*
90-04   H. S. Park, *Abstract object types · abstract data types · abstract knowledge types · abstract connector types*
90-05   S. Amin and H. S. Park, *KEED: Knowledge engineering environment for diagnostic problems.*
90-06   H. S. Park, *Abstract knowledge prototyping.*
90-07   J. K. Kearney and S. Hansen, *Generalizing the hop: object-level programming for legged motion.*