UNIVERSITY OF CALIFORNIA, SAN DIEGO

Multibody Dynamical Algorithms, Numerical Optimal Control, with Detailed Studies in the Control of Jet Engine Compressors and Biped Walking

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy
in Electrical Engineering (Intelligent Systems,
Robotics, and Control)

by

Michael William Hardt

Committee in charge:

Professor Kenneth Kreutz-Delgado, Chairperson
Professor J. William Helton
Professor Mohan Trivedi
Professor David D. Sworder
Professor Robert E. Skelton

1999

TABLE OF CONTENTS

Chapter V

Nonlinear $\mathcal{H}_2$ and $\mathcal{H}_\infty$ Control and Numerical Solution Techniques

V.A Introduction

This chapter discusses various numerical techniques for the solution of nonlinear $\mathcal{H}_2$ and $\mathcal{H}_\infty$ control problems with full state feedback. We present first the definitions of these control formulations and give some indication of their importance in nonlinear control. Their linear analogues already have had much success in the control community. Linear $\mathcal{H}_2$ or LQR control design dates back to Kalman's work in the early 60's and continues to be frequently used. Linear $\mathcal{H}_\infty$ control began to gain favor in the early 80's with the work of Zames [101], Helton [32], and Doyle [20]. State space solutions for the problems were later presented in [19]. The continued popularity of $\mathcal{H}_2$ and $\mathcal{H}_\infty$ control methodologies stems largely from their very good stabilization and robustness properties [102]. There are many types of control problems which fit naturally into the $\mathcal{H}_2$ or $\mathcal{H}_\infty$ framework such as stabilization problems, trajectory tracking, risk-sensitive problems, and disturbance attenuation problems [28, 102].

The extension of the linear methods to nonlinear control problems followed soon afterwards in the late 80's and early 90's with the pioneering work in

[7], followed by the work reported in [8, 37, 91]. In the nonlinear regimes, computational methods for solving these problems are still in their early stages, and there is no generally accepted format for the best approach to solving them. The existence, however, of numerical procedures which can solve these problems is *vastly* underappreciated. In Section V.D, we outline many of the proven numerical methods for solving these problems. We pursue in Section V.C two such approaches for solving nonlinear $\mathcal{H}_2$ and $\mathcal{H}_\infty$ control problems which we feel are well-suited to handling saturation and other constraints of the problem, are efficient, and can accomodate higher dimensional systems to some extent.

After the discussion of the various existing numerical approaches, we illustrate how our proposed numerical scheme is implemented by using the test case of a swinging pendulum. Important subtleties of numerical optimal control become very evident even in this case. For example, there may exist initial conditions of the system for which there exist two completely different, equally optimal solutions. This can lead to situations where a small change in the initial conditions can lead to radically different solution trajectories.

The last section of the chapter, Section V.F, is more in the spirit of the previous chapters on exploiting the recursive algebraic structure of multibody systems. It outlines how one may solve optimal control problems associated with multibody systems via the method of characteristics. Using previous results from Chapter IV on recursive, symbolic equations for the linearized dynamics, a recursive and efficient scheme is presented for the calculation of the adjoint equations for this class of systems.

V.B Nonlinear $\mathcal{H}_2$ and $\mathcal{H}_\infty$ Problems

The class of state equations that will be considered will be nonlinear, smooth, time-invariant having the form,

$$\begin{aligned} \dot{x} &= f(x,u,w) = a(x) + b_1(x)u + b_2(x)w \\ z &= \begin{bmatrix} h(x) \\ u = l(x) \end{bmatrix} \end{aligned} \tag{5.1}$$

where $x(t) \in X$ is the state, $u(t) \in U$ is the controlled input, $w(t) \in \mathcal{L}_2[0,\mathrm{T}]$, $T \geq 0$, is the disturbance input, and z are the to-be-controlled outputs or penalties.

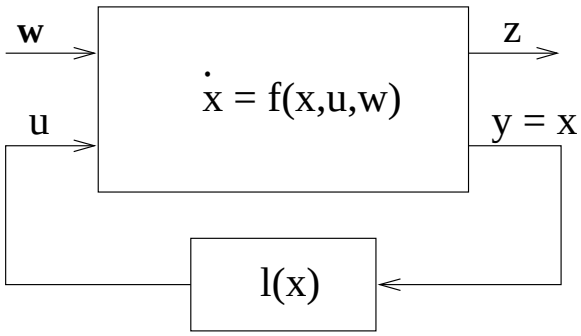

Figure 5.1: System Diagram

V.B.1 Nonlinear $\mathcal{H}_2$ Control and the *HJB* PDE

In the nonlinear $\mathcal{H}_2$ control problem, no disturbances enter into the model (5.1), $w \equiv 0$. The problem lies in minimizing a cost functional of the form,

$$V(x) = \inf_u \left\{ \frac{1}{2} \int_0^T ||z(t)||^2 \, dt + \varphi(x(T)) \right\}, \tag{5.2}$$

which is often called the *value function.*

For numerical purposes, it is possible to incorporate the state equations of the system into the cost functional, whereby traditional nonlinear optimization software may be applied to it. One may also solve the problem by defining a

Hamiltonian function $H(x,p)$,

$$H(x,p) = p^T a(x) - \frac{1}{2} p^T b_1(x) b_1^T(x) p + \frac{1}{2} h^T(x) h(x)\,, \tag{5.3}$$

which in the $\mathcal{H}_2$ case is convex. The solution to the $\mathcal{H}_2$ control problem will satisfy a *Hamilton-Jacobi-Bellman* inequality (*HJB*) defined by $H(x, \frac{dV^T}{dx}) \leq 0$ [92]. In the numerical methods to be discussed, we will be seeking the null Hamiltonian solution where $H(x, \frac{dV^T}{dx}) = 0$.

The solution of the control problem is thus "simplified" to the solution of a first-order PDE.

V.B.2 Bicharacteristic ODEs

It has long been known that this class of PDE's is equivalent to a system of ODE's of double the dimension [9]. In the case of (5.3), the corresponding boundary value problem is described by the Hamiltonian vector fields $\frac{dx}{dt}$ and $\frac{dp}{dt}$. The vector p is denoted as the set of adjoint variables for the system, and it is also equal to the transpose of the gradient of the value function, $p = \frac{dV^T}{dx}$. The boundary value problem consists of the following "bicharacteristic" equations:

$$\begin{array}{rclcrcl} \dot{x} & = & \frac{\partial H}{\partial p}(x,p) & & \dot{p} & = & -\frac{\partial H}{\partial x}(x,p) \\ x(0) & = & x_0 & & p(T) & = & \frac{\partial \varphi(x(T))}{\partial x} \end{array} \tag{5.4}$$

The Hamiltonian will be constant and equal to zero along an optimal trajectory, thus solving the *HJB* inequality.

This boundary value problem characterizes solving the finite time $\mathcal{H}_2$ or $\mathcal{H}_\infty$ control problem with no final state constraints. We take the approach in this paper of indirectly requiring the state $x(t)$ to converge to a desired equilibrium x^* by including a penalty function of the squared error of the state from the equilibrium into the performance. This has the effect that the system will converge asymptotically to x^* with speed determined by the gains chosen.

V.B.3 Optimal State Feedback Law

The vectors x and p may be used to define the optimal control

$$u^* = -b_1^T(x)\, p \tag{5.5}$$

along each optimal trajectory. The numerical values for x, p, and u make up an open-loop solution as all variables will only be known as a function of time and only at a discrete set of points along the trajectory. Once one has calculated many such trajectories, one can begin to approximate the numerical function $x \mapsto u$, known only on a grid, by a function $u^*(x)$ given for all x (at least all x in some neighborhood) by a formula. This is done to interpolate between grid points, so one has a control law which can be quickly applied to any state. An example of an approximation method which can scale up to higher dimensions is a neural network such as can be found in the easy-to-use Matlab neural network toolbox [62]. Later in Chapter VI we use this to derive the optimal feedback controller for the jet engine control problem.

V.B.4 Nonlinear $\mathcal{H}_\infty$ Control

In the nonlinear $\mathcal{H}_\infty$ control problem, the model used (5.1) contains $\mathcal{L}_2$-bounded disturbance inputs w. The objective here is to find a controller which guarantees that the closed-loop system has an $\mathcal{L}_2$-gain less than or equal to a prespecified constant level γ from the disturbance inputs w to the output penalty functions z [92].

We write this $\mathcal{L}_2$-gain condition as

$$\int_0^T ||z(t)||^2 dt \leq \gamma^2 \int_0^T ||w(t)||^2 dt + \beta(x(0)) \tag{5.6}$$

for all disturbance functions $w(\cdot) \in \mathcal{L}_2[0,T]$, $T \geq 0$. The constant value $\beta(x(0))$ satisfies $0 \leq \beta(x) \leq \infty$, and $\beta(x^*) = 0$, where x^* is the equilibrium for the system. $z(t)$ is the response of the system for the initial condition $x(0)$.

As in eq. (5.2), one may define the nonlinear $\mathcal{H}_\infty$ control problem as an optimization of a performance index. Unlike the $\mathcal{H}_2$ problem, however, the optimization is of a minimax-type which prevents the use of standard nonlinear optimization software.

One may similarly define a Hamiltonian function $H(x,p)$, which is now in general nonconvex, for the nonlinear $\mathcal{H}_\infty$ control problem,

$$\begin{aligned} H(x,p) &= p^T a(x) + \frac{1}{2} p^T \left[\frac{1}{\gamma^2} b_2(x) b_2^T(x) - b_1(x) b_1^T(x) \right] p \\ &\quad + \frac{1}{2} h^T(x) h(x) \ . \end{aligned} \tag{5.7}$$

The solution to this problem, as in the nonlinear $\mathcal{H}_2$ control problem, may be obtained by solving for a C^1 solution $V(x)$ to the so-called *Hamilton-Jacobi-Isaacs* inequality *HJI*, $H(x, \frac{dV^T}{dx}) \leq 0$. The solution may also be calculated by solving a set of bicharacteristic equations as in (5.4).

V.C Proposed Numerical Solution of Nonlinear $\mathcal{H}_2$ and $\mathcal{H}_\infty$ Problems

The numerical approach to be discussed consists of several methods which may be used independently or in combination for solving the *HJB* and, in some cases, the *HJI* equation. These methods will calculate the open-loop solution to an optimal trajectory. The first method to be discussed is limited in that it can only solve the nonlinear $\mathcal{H}_2$ optimal control problem.

V.C.1 Direct Minimization

To solve the $\mathcal{H}_2$ control problem numerically with the direct minimization approach, one discretizes the state and control variables in time over the trajectory. One may then formulate the minimization of the cost functional subject to the differential and output constraints as a finite-dimensional, nonlinearly constrained, optimization problem. The convexity of the $\mathcal{H}_2$ problem allows conven-

tional Newton-based methods such as Sequential Quadratic Programming (SQP) to be used to solve the problem. Also, the direct minimization approach will find the solution to the *HJB* equality, $H(x, \frac{dV^T}{dx}(x)) = 0$. The program DIRCOL from Oskar von Stryk [94, 95], which uses the nonlinear optimization software NPSOL (Gill, Murray, Saunders, Wright [26]), takes this approach and, in addition, it provides accurate estimates of the adjoint variables p.

V.C.2 Multiple Shooting

The second method, multiple shooting, may be applied to either the $\mathcal{H}_2$ control problem or the $\mathcal{H}_\infty$ problem. The equations to be solved are the two-point or multi-point boundary value problem described by (5.4). An important property of this approach is that it can also handle nonconvex problems.

The prescribed boundary conditions for the problem are initial values of the state $x(0)$ and final values of the adjoint variables $p(T)$. The traditional shooting method integrates the state equations beginning at $x(0)$ and from a guess for $p(0)$ for a prespecified time T. When the numerical value obtained for $p(T)$ from the integration does not match the boundary condition $p(T) = \frac{\partial \varphi(x(T))}{\partial x}$, a zero-finding problem may be set up in terms of the constraints, and Newton's method may then be used to iterate on $p(0)$ until the boundary condition is satisfied.

The strong sensitivity of initial value problems, and the small convergence region for the Newton's method make solving these problems via simple shooting very difficult. A multiple-shooting algorithm, however, is more stable and robust to nonlinearities. This procedure consists of dividing up the time interval $[0, T]$ into a series of grid points $[0, t_1, t_2, \ldots, T]$, and a shooting method is performed between successive grid points (see Figure 5.2). A set of starting values for the state and adjoint vectors is required at each grid point in time, and continuity conditions for the solution trajectory introduce additional interior boundary conditions which are then incorporated into one large zero-finding problem to be solved.

The program MUMUS by Peter Hiltmann [33] is a multiple-shooting code

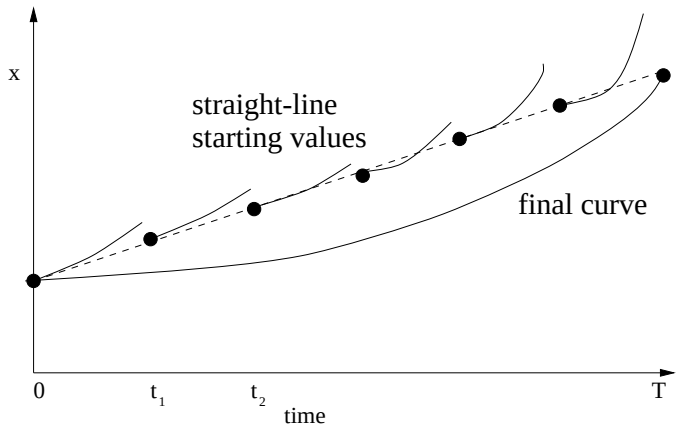

Figure 5.2: Multiple Shooting Method

which can solve two-point or multi-point boundary value problems using a modifed Newton method. This method is, in general, quicker than direct minimization, lending itself more readily to iterated calculations of trajectories. A drawback, however, is that the adjoint equations must be programmed as input to the code. For complicated problems, generating the equations by hand or with symbolic manipulation packages may pose a formidable task.

V.C.3 Shooting Out

While the multiple shooting method is commonly used for solving a boundary value problem, one may alternatively obtain optimal trajectories by starting near the desired equilibrium x^* and at selected values $\hat{p}$, then integrating the state and adjoint differential equations outwards. This process is called *shooting out.* We are, thus, able to obtain optimal trajectories for the state without having to solve a difficult two-point boundary value problem. A set of starting values (a guess for the solution trajectory) is also not needed as is the case in *direct minimization* and *multiple shooting.* This technique was successfully used by McEneaney in [63].

The disadvantage of the shooting out method is that one has less control in reaching a specified set of states. Remaining regions of the state space, that

are not able to be mapped out with optimal trajectories via *shooting out*, may be mapped with the previous multiple shooting approach using as starting values the optimal trajectories already obtained.

V.C.4 Combined Method

When one wishes to map out the value function over a large region of the state space for purposes of approximating well the optimal closed-loop controller, it is necessary to calculate a fair number of optimal trajectories. We can use the superiority of *direct minimization* or *shooting out* methods to generate several of these trajectories when little or no prior knowledge of the solution is available in order to acquire starting values for the multiple shooting method when solving for the remainder of the trajectories. The multiple shooting method works well when many different optimal trajectories are sought after since the algorithm is fast, and one may initialize each run from the previous run. One has much more control in the mapping out of the state space than one might have with shooting out, and the method is, in general, quicker than direct optimization methods.

V.C.5 Homotopy

In order to compute the value function V and the optimal control u^*, we must compute a family of optimal trajectories which approximately (maybe only coarsely) sweep out most of the state space. Supposing we have computed an optimal trajectory T_{x_1} from a state x_1 to the equilibrium x^*, to quickly compute trajectories T_x from many initial states x, simply pick x_2 near x_1 and use T_{x_1} to initialize a MUMUS or DIRCOL computation of T_{x_2}. We have found this proceeds quickly to give us many optimal trajectories (see Section VI.B.5). Henceforth, we refer to this as homotopying trajectories, since the point is to move them gradually.

When using the multiple shooting method, one may similarly homotopy in the initial conditions of the state variables, using *both* the state and adjoint trajectories previously calculated to serve as starting values for the new boundary

value problem to be solved. By calculating trajectories over the portion of state space of interest, one will know u^* and V at points over this entire area. Multivariate approximation methods may then be used to find $\hat{u}^*$ approximating u^* as a function of the state x.

After the calculations have been made, the approximation $\hat{u}^*$ of the closed-loop controller may then be implemented for use in real-time.

V.D Other Numerical Methods

There are many numerical approaches which have been tried on *HJB* and *HJI* equalities and inequalities and listing them all is beyond us. We shall try to list some of the main ones.

V.D.1 Finite differences

The method of finite differences has been used successfully for solving *HJB* and *HJI* problems of low dimension. For an account of finite difference methods and their relation to Markov chains see Kushner and Dupuis [54]. Also Falcone [22] and the doctoral thesis by P. Dower under M. James [18] have used extensively finite differences in this setting. It is not clear how these would extend to high dimensions in a manageable way, since rectangular grids have a rigid geometry.

V.D.2 "Small" basis of functions

The idea in this approach is to select basis functions f_j, expand the value function V as $\sum_{j=1}^{M} c_j f_j$, then some work is required to find the proper coefficients c_j. The main difficulty lies in selecting the basis functions f_j. The common approach in conventional numerical PDEs (in dimension < 4) is to use basis functions which are piecewise polynomials. In principle, in order to solve a first order PDE like *HJB*'s, they should be continuous. The tessellating (such as tetrahedra) of space into pieces which match up is a serious problem. Doing this flexibly in 3 dimensions

took substantial study, but of course is well developed now. The advantage is that once everything is set up, mesh refinement can be done on regions of space which may otherwise seem treacherous. However, doing this in say 5 dimensions may well be very difficult.

One can select basis functions in a way which is "ad-hoc" and much less difficult. For example, Beard and McLain [10] use physical intuition coming from a particular plant model to select 10 or 12 basis functions. For example, if the model contains sines and cosines and powers of x, then their f_j's will too.

For control purposes, one need only solve *HJI* inequalities, provided the solutions V satisfy conditions which make them what are called *viscosity subsolutions*. If continuity is not required for V and jumps are allowed, the problem becomes easier, but the control law typically has jumps.

Once a basis of f_j is selected, one can find the c_j's in several ways. One is a Galerkin procedure for solving *HJI* equations, described for example in [10]. Another is by directly substituting $V = \sum_{j=1}^{M} c_j f_j$ into the *HJI* inequality to obtain inequalities of the form

$$\begin{aligned} q(\{f_j : j = 1, \ldots, M\}, x) &\geq 0 \quad \forall x \\ \sum_{j=1}^{M} c_j f_j &\geq 0 \ . \end{aligned}$$

Here, q is quadratic in the f_j, and one elects some finite set of x to yield finitely many constraints (crucial areas can be heavily gridded). Various optimization schemes can be applied.

V.D.3 Partly Analytical Methods

Very intriguing is backstepping, which applies to special classes of systems and obtains analytical expressions for V and the controller u. Thus, when it applies, it has the potential for beating the curse of dimensionality. One advantage of obtaining analytical expressions for V and u is that often the solution need not be recomputed with changes in the parameter values or choice of equilibrium.

Backstepping does not approach saturation directly, however, although a highly experienced practitioner may well come up with compromises. This makes it difficult to compare our efforts with backstepping, especially in the jet engine compressor problem considered in Chapter VI where we will see that rate saturation dominates.

Similarly, feedback linearization is not able to address saturation directly.

V.D.4 Characteristics and Direct Minimization

The method of characteristics, as they pertain to multiple shooting and shooting out, and direct minimization methods were presented in the previous section. They seem to have the most promise among methods for "fully" solving *HJB* equations. The shooting out or multiple shooting methods can be a bit difficult to enter into a program at this point, since the adjoint equations can be complicated. Direct methods, while slower, are comparatively easy to run. All of these methods find the value function $V(x)$ at each point x along a curve. In delicate regions, one can pick many x's, otherwise only a few x's may be necessary to approximate $V(x)$ well. A careful approach can, thus, help combat the curse of dimensionality.

The final desired result is a "simple" formula to approximate the function V in order to obtain a control law. Traditional Galerkin methods reverse this in that they layout the approximation scheme first (before having any information about V). Since the Galerkin scheme must mesh very well with the numerical PDE one is solving, this is much harder to do than merely approximating V given as data on a grid.

V.D.5 Miscellaneous

Other approaches include the technique using *viability kernels* as promoted by P. Saint-Pierre [82]. Bertsekas and Tsitsiklis also solve *HJB* and *HJI* equations for discrete problems in their book [12].

V.E Illustration: Pendulum

In order to demonstrate the basics and subtleties of applying these numerical techniques to a control problem, we present here a brief tutorial example of the optimal control of a pendulum. Already in this simple problem, one finds many numerical particularities which are important. For example, we discover multiple *sheets* of the pendulum. These are regions of the state space where numerically calculated optimal trajectories will stay within when iterating on the initial conditions. Yet as one progresses far enough along the sheets, it may be the case that the calculated trajectories are no longer optimal.

Later, when we deal with the more complex problem of jet engine compressor control, we must, for example, be careful to rule out multiple *sheets* or to find them. As we shall see in that case, extensive searching did not reveal any such anomalies.

The dynamical equations of motion of a pendulum are,

$$m\, l^2\, \ddot{\theta} + m\, g\, l \sin(\theta) = u$$

where θ is the position angle of the pendulum with respect to the bottom 'rest' position. The pendulum rod is assumed to be massless of length l with a point mass of m at its end, and g is the gravity constant. See Figure 5.3.

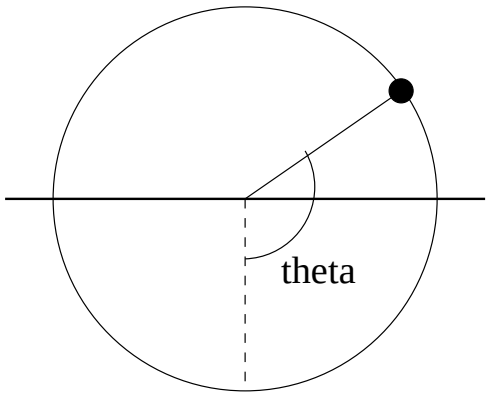

Figure 5.3: Pendulum

Let $x_1 = \theta$ and $x_2 = \dot{\theta}$, and for simplicity let $l = m = 1$. Then we write the state and output equations as

$$\begin{array}{rcl} \dot{x}_1 &=& x_2 \\ \dot{x}_2 &=& -g\sin(x_1) + u \end{array} , \qquad z = \begin{bmatrix} (x - x_f) \\ (u - u_f) \end{bmatrix} , \tag{5.8}$$

where x_f is a desired equilibrium of the pendulum and u_f is the control effort at the equilibrium. We specify a weighted quadratic cost functional to be optimized as

$$\min_u \frac{1}{2} \int_0^T \left\{ \|x - x_f\|_Q^2 + \|u - u_f\|_R^2 \right\} dt + \varphi(x(T)) \tag{5.9}$$

where $Q, S \geq 0$ and $R > 0$ are square matrices of appropriate dimension, and the norm $\|z\|_Q^2$ satisfies $z^T Q z$ for some vector z and matrix Q. The penalty function on the final state is equal to $\varphi(x(T)) = \frac{1}{2}\|x(T) - x_f\|_S^2$.

We wish to solve the *HJB* equality (5.3), $H(x, \frac{dV^T}{dx}) = 0$, for V. From the Hamiltonian, we construct the *bicharacteristic equations* (5.4) with initial and final boundary conditions

$$x(0) = x_0, \qquad p(T) = \frac{\partial \varphi}{\partial x}(x(T)) = S(x(T) - x_f) \ .$$

Once these are solved, a closed-form expression for the optimal control $u^*(x)$ along the bicharacteristic trajectory can be determined from $p = \frac{dV^T}{dx}(x)$ and the extremal value for u^*,

$$u^* = -R^{-1}\, {b_1}^T(x)\, p \ . \tag{5.10}$$

V.E.1 Nonlinear $\mathcal{H}_2$ Solution and Comparison of Numerical Methods

We compare both the *direct minimization* approach (DIRCOL) and the *multiple shooting* method (MUMUS) in our numerical experiments. Of interest are their respective run-times, the effects of varying numbers of grid points (in time), and their convergence properties.

With the multiple shooting code MUMUS, for example, it was found that run-times are not as affected by the choice of discretization as the optimization

approach of DIRCOL. Also, given a good set of starting values, MUMUS can reach the solution very quickly. The difficulty with MUMUS lies in obtaining reasonable starting values, in particular, those for the adjoint variables. For the pendulum, perhaps due to its low dimension and relatively mild nonlinearities, MUMUS was able to converge well even with the obvious choice of straight-line starting values. When iteratively calculating many trajectories with large numbers of grid points, MUMUS thus was the obvious choice.

The direct optimization package DIRCOL makes it much easier for the user to define the problem and to enter in nonlinear equality and inequality constraints when desired. The use of this approach is very useful in a preliminary analysis of an optimal control problem, even one as simple as the pendulum. Also, the *HJB* equality in the nonlinear $\mathcal{H}_2$ control problem may have more than one solution. An optimization code will converge to the stabilizing solution of the *HJB* equality, while it is not guaranteed that the shooting code will do the same. Since for small numbers of grid points the run-times of the two methods are very similar, a natural approach is to solve the problem initially with DIRCOL or by *shooting out*, then to initialize MUMUS with its output. This was the combined method previously described in Section V.C.4.

V.E.2 Multiple Sheets of the Value Function

The pendulum has been a heavily studied test problem. Even in our case, this example sheds light on some rather interesting properties of optimal control and of our choice of numerical methods. When applying the previously described homotopy method, we discover sheets along which the multiple shooting method will continue to produce solutions to the $\mathcal{H}_2$ control problem, though the calculated trajectories may no longer be optimal.

In our analysis, the state x_1 represents the position angle which determines the configuration of the system and x_2 is the angle velocity. In the coordinates chosen, $x_1 = 0.0$ refers to the bottom 'rest' position of the pendulum, and the

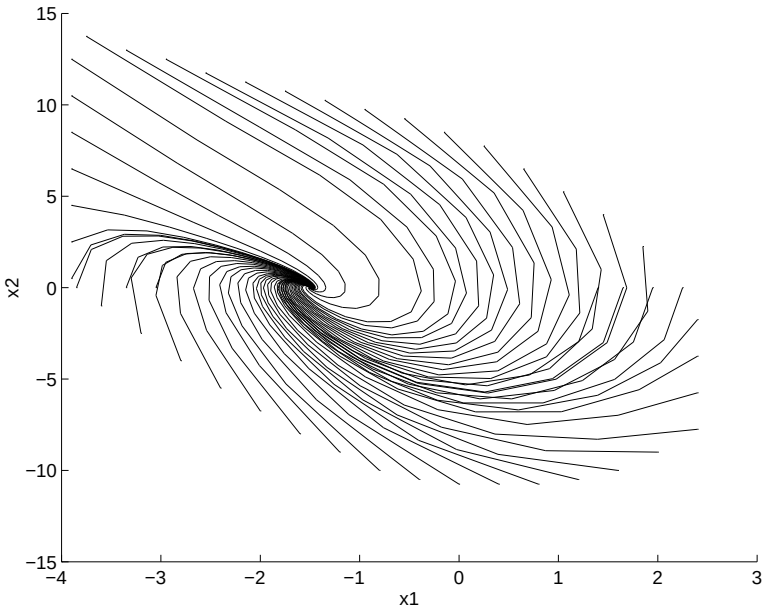

Figure 5.4: Plot of optimal trajectories calculated in the state space to the equilibrium $x = (\theta, \dot{\theta}) = (-1.5, 0)$.

desired final equilibrium is chosen as $x_f = (x_{f1}, x_{f2}) = (-1.5, 0)$ [1].

In order to calculate V and u, the homotopy method is used with the multiple shooting program MUMUS to sweep out a large region of the state space, repeatedly calculating trajectories with slightly varying initial conditions. Figure 5.4 displays the calculated trajectories. the various trajectories. We take the optimal trajectory with initial condition $(x_1(0), x_2(0))$ as an initialization for another run of MUMUS with initial condition $(x_1(0) + \delta_1, x_2(0) + \delta_2)$. The choice of (δ_1, δ_2) will determine how accurate the resulting value function and control law will be. In our experiment, a homotopy was performed over a large region of state space, roughly $-3.9 < x_1 < 2.4$, $-11 < x_2 < 14$. The final time T, to which the trajectories are calculated, was chosen so that all of the optimal trajectories converged to within a small tolerance level of the equilibrium. Finding an appropriate T required some trial and error.

[1] All angles are in radians.

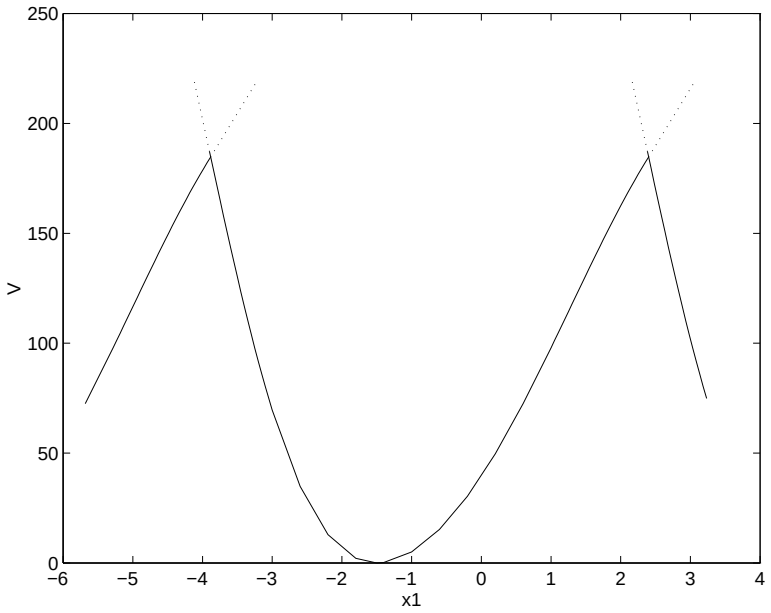

Figure 5.5: Homotopy in $x_1(0)$, $(x_2(0) = 0)$ along two sheets of pendulum. The dotted lines indicate the continuation of a *sheet* along which the numerical methods will continue to calculate solutions though they are no longer optimal.

Our experiment consisted of two phases. The initial phase consisted of a homotopy in $x_1(0)$ along the two primary sheets of the pendulum. One sheet traverses the pendulum in the clockwise direction iterating the initial condition $(x_1(0), x_2(0) = 0)$ in the negative x_1 direction. Each run uses the previous run as its initialization. The same is done in the counter-clockwise direction. Since x_1 is periodic, eventually the homotopy will calculate trajectories on two different sheets corresponding to the same initial condition. These two trajectories will be different since they will correspond to traversing the pendulum in opposite directions to reach the desired final endpoint and; hence, they will have different values for the value function. Recall from eq. (5.2) that the value function $V(x)$ is the infimum of a cost functional over all possible trajectories. The 'true' V will be the lowest of all the values obtained, while the higher values are thrown out as they are not 'optimal.' More details on combining different calculated values for V to construct

the true value function are described by McEneaney [63].

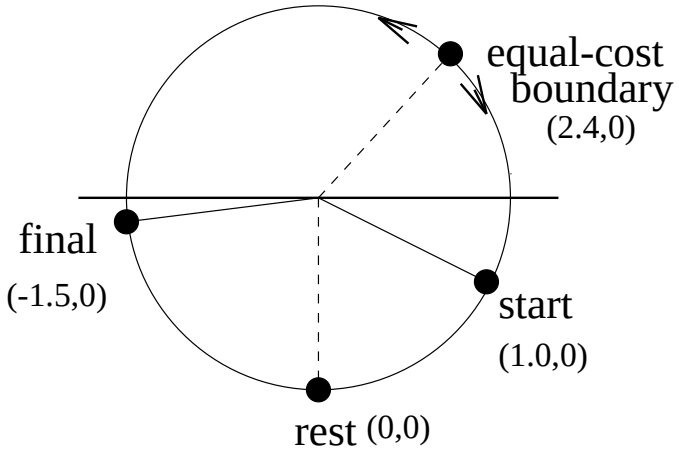

Figure 5.6: Equal-cost Boundary on Pendulum

Figure 5.5 represents a vertical slice along the x_1 axis of the value function. On the vertical axis is the value function V which represents the cost in reaching the desired equilibrium x^*. This cost is defined by the quadratic performance (5.9). Note that since x_1 is periodic in $2\pi \approx 6.3$, $x = (-3.9, 0)$ is the same point as $x = (2.4, 0)$. The value function has the same value there, $V(-3.9, 0) = V(2.4, 0) = 185$. At this point, it is equally costly for the pendulum to travel counter-clockwise as it is to travel clockwise in order to reach the desired equilibrium as shown in Figure 5.6. The state space is not a simply connected region, and it is therefore to be expected that there exist points from which two equally costly and optimal paths exist to the desired final state.

In the second phase, we iterate the initial values of the trajectories we wish to compute in such a manner as to keep $(x_1(0), x_2(0))$ near a level set of $V(x)$. This way of choosing $(x_1(0), x_2(0))$'s is not essential but easy to implement. In Figure 5.7, we have graphed the value function for $V(x) \leq 185$. Using an approximation method, we then obtain the desired closed-loop control law.

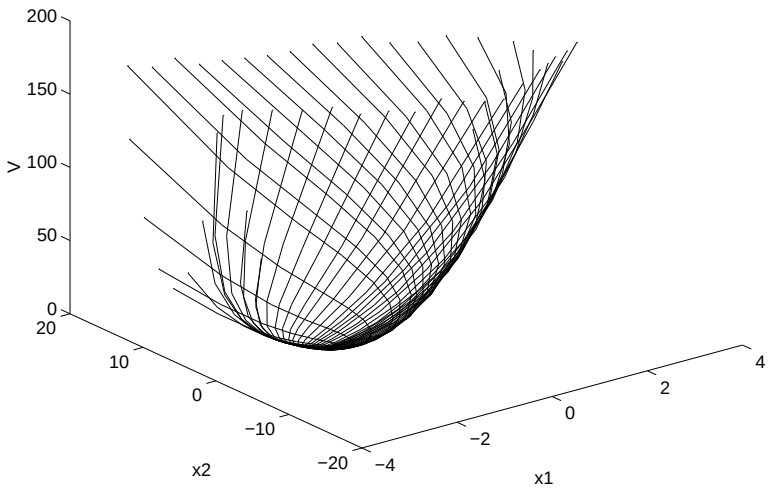

Figure 5.7: Value function of pendulum.

V.F Recursive Calculation of Robot Adjoint Equations

We have discussed how the solution to a nonlinear $\mathcal{H}_2$ or $\mathcal{H}_\infty$ optimal control problem satisfies a Hamilton-Jacobi-Isaacs equation as well as a 2-point boundary value problem described by the state equations and a set of adjoint equations (5.4). The boundary value problem may be obtained with the optimal control $u^*(x,p)$, and worst-case disturbance $w^*(x,p)$ in case of an $\mathcal{H}_\infty$ control problem, from the Hamiltonian H,

$$\dot{x} = \frac{\partial H(x,p,u^*,w^*)}{\partial p} , \qquad \dot{p} = -\frac{\partial H(x,p,u^*,w^*)}{\partial x} . \tag{5.11}$$

These equations may be represented entirely in terms of the state x and the adjoint variables p.

In recent years, powerful multiple shooting algorithms that can solve these boundary value problems have been used with greater frequency [50, 29, 63], though the task of generating the adjoint equations can often be a formidable obstacle. This has also been the case with multibody systems where the size and complexity of the equations of motion increases at a tremendous rate with increasing dimension.

The adjoint equations which involve the partial derivatives of the dynamics with respect to the state variables will even be more lengthy and complex. Symbolic manipulation packages have at least made available the possibility to obtain these equations, though storage and computational considerations place severe limits on the dimension of the problem as the entire set of equations in closed form are needed.

We, thus, recognize the utility of recursive symbolic expressions for the calculation of the adjoint equations of a general robotic system. Using the equations presented earlier for the recursive, symbolic calculation of the linearized dynamics in Section IV.F, we describe how the adjoint equations for a multibody system may be calculated recursively using the symbolic matrix operators in an efficient manner. The ability to recursively calculate the adjoint equations for this class of systems in a manner which does not suffer the common problems related to the "curse of dimensionality" opens up many possibililties for using numerical optimal control techniques on these systems. These tools and ideas have not previously been presented in the research literature.

As previously presented in the context of nonlinear $\mathcal{H}_2$ or $\mathcal{H}_\infty$ optimal control, one has a performance index to be optimized of the form

$$V(x) = \min_u \max_w \int_0^T L(x(t), u(t), w(t))\, dt \ , \tag{5.12}$$

where $L(x, u, w)$ is the integrand of the performance index for the system, and u and w are the control and disturbance respectively. In the case of an $\mathcal{H}_2$ problem, $w \equiv 0$. The state x is defined in the case of our robotics system as $x = [x_1^T \ x_2^T]^T$. Define similarly a set of adjoint variables $p = [p_1^T \ p_2^T]^T$ of the same dimension so that the Hamiltonian function is defined as

$$H(x, p) = p^T \begin{bmatrix} x_2 \\ \mathcal{M}^{-1}(u + w - \mathcal{C} - \mathcal{G}) \end{bmatrix} + L \ . \tag{5.13}$$

Using (5.11), the bicharacteristic equations for the multibody system are

$$\begin{aligned} \dot{x}_1 &= \nabla_{p_1} H &&= x_2 \\ \dot{x}_2 &= \nabla_{p_2} H &&= \mathcal{M}^{-1}(u + w - \mathcal{C} - \mathcal{G}) \\ \dot{p}_1 &= -\nabla_{x_1} H &&= -(\tfrac{\partial}{\partial\theta}\ddot{\theta})^T p_2 - \nabla_\theta L \\ \dot{p}_2 &= -\nabla_{x_2} H &&= -p_1 - (\tfrac{\partial}{\partial\dot{\theta}}\ddot{\theta})^T p_2 - \nabla_{\dot{\theta}} L \end{aligned} \tag{5.14}$$

The partial derivatives of the integrand of the performance, $\nabla_\theta L$ and $\nabla_{\dot{\theta}} L$, are dependent on the form of L. In standard cases, L is quadratic in x, u, and w (see (5.6)). Thus, its partial derivatives will be only linear in x as we can subsitute for $u^*(x,p)$ and $w^*(x,p)$ after taking partial derivatives. The equations for $\dot{x}$ are merely the forward dynamics presented in Section II.G with the optimal values u^* and w^* substituted for u and w. The form for $u^*(x,p)$ was given in (5.5) while the form for $w^*(x,p)$ follows similarly by taking the partial derivative with respect to w of the Hamiltonian. Assume that L is quadratic in its arguments so that the Hamiltonian has the form

$$H(x,u,w) = p^T(a(x) + b_1(x)u + b_2(x)w) + x^T x + u^T u - \gamma^2 w^T w \; . \tag{5.15}$$

Then we can establish the following result:

Proposition 34 *A recursive, symbolic matrix expression for calculating the adjoint equations of a multibody system is given from the general form of the adjoint equations (5.14), the optimal values for the control u^* and disturbance w^*,*

$$\begin{aligned} u^*(x,p) &= -b_1^T(x)p && (5.16) \\ &= -\mathcal{M}^{-1}p_2 = -[I - K\psi H]D^{-1}[I - K\psi H]^T p_2 && (5.17) \\ w^*(x,p) &= \frac{1}{\gamma^2} b_2^T(x)p && (5.18) \\ &= \frac{1}{\gamma^2}\mathcal{M}^{-1}p_2 = \frac{1}{\gamma^2}[I - K\psi H]D^{-1}[I - K\psi H]^T p_2 && (5.19) \end{aligned}$$

and the recursive, symbolic expressions for $\frac{\partial}{\partial\theta}\ddot{\theta}$ and $\frac{\partial}{\partial\dot{\theta}}\ddot{\theta}$ given in Section IV.F which when combined with p_2 gives

$$(\frac{\partial}{\partial\theta}\ddot{\theta})^T p_2 = \Big(- H^T(f)^{\otimes T} + H^T\phi^T\widetilde{\dot{V}_g^T}M + 2\dot{H}^T\phi^T\check{M}$$

$$+\ddot{H}^T\phi^T M\Big)\psi H D^{-1}[I - K\psi H]^T p_2 \tag{5.20}$$

$$(\frac{\partial}{\partial\dot{\theta}}\ddot{\theta})^T p_2 \;=\; (H^T\phi^T\check{M} + \dot{H}^T\phi^T M)\psi H D^{-1}[I - K\psi H]^T p_2 \tag{5.21}$$

These recursive equations may be implemented in a manner analogous to the equations presented in Chapters II, III, and IV. The values for u^* and w^* can be calculated in two sweeps of the multibody chain which when combined with the forward dynamics will add an additional two sweeps to the three normally required to calculate $\dot{x}$. The additional calculation for the adjoint equations $\dot{p}$ requires four sweeps of the multibody chain and can, thus, be performed in parallel with the augmented forward dynamics algorithm.

V.G Notes

The text of Chapter V, in part, is a reprint of the material as it will appear in *IEEE Control Systems Technology* under the title "Numerical Solution of Nonlinear $\mathcal{H}_2$ and $\mathcal{H}_\infty$ Control Problems with Application to Jet Engine Compressors," and with authors, Michael Hardt, J. William Helton, and Kenneth Kreutz-Delgado. The dissertation author was the primary researcher and author and the co-authors listed in this publication directed and supervised the research which forms the basis for this chapter.

Chapter VII

Case Study: Minimum Energy Biped Walking

VII.A Introduction

Walking is a characteristic of animals which takes them from one geographic position to another. There are many variations of walking such as moving at different speeds, starting, stopping, changing direction, and moving up and down different grades of slopes. All walking motion consists of a rhythmic displacement of body parts to maintain a forward motion. If the action is unchanging, then the motion will generally become periodic. People have a particularly unstable configuration as they walk with only two legs. After human walking has been learned, it seems to be such a simple process, yet it remains as one of the principle open research problems in robotics due its complexity and high dimension, particularly for gracile systems having long limbs and small feet.

Our goal in this chapter will be to understand better the mathematical modeling of the human walking motion. We will create a multibody system model of a biped walker and calculate its motion using its dynamical properties. As in any control problem, the input applied forces which stimulate motion in the biped (much as muscles in a person cause a person to walk) are initially unknown. By

selectively choosing the correct performance objectives for a walking system, we may solve for these applied forces as the solution to an optimal control problem. Since it has been shown by biologists [79, 89] that people naturally walk in an energy efficient manner, we make the choice to minimize the injected energy into the system as a function of the input applied forces. Our solution of the minimum energy biped walking problem produces a very natural walking motion. It is also the first reported results using a model nearly as complex and close to a human model as the one we use. Preliminary results for the soluton of this problem were first presented in [31].

Many researchers have studied the problem of controlling a walking biped robot and the associated path planning problem of generating optimal periodic trajectories for the walking motion. Several of the past innovations and recent contributions are found in the references [17, 16, 27, 25, 46, 64, 80, 81, 85]. Due to the complexity of the problem, compromising simplifying modeling assumptions were often made to make it more tractable. Consider a simple 5-link biped robot with all rotational joints and full motion degrees of freedom. It will have a 14 dimensional state space when represented with respect to generalized coordinates. This is in addition to several general nonlinear constraints which "turn-on" in a state-dependent and possibly asynchronous manner. For example, when a foot is in contact with the ground, this produces algebraic constraints. The result is a time-varying differential algebraic system.

During walking, there are naturally two phases of a step or what is also commonly called a *gait cycle*.[1] A gait cycle consists of an interval of time during which a periodic succession of events occurs. The first phase has one foot in contact supporting the body while the other swings, while in the shorter second phase both feet are in contact with the ground. During both phases we have a differential-algebraic system, where in the second phase the number of algebraic constraints will

[1] We will not consider gaits associated with running and can ignore the possibility of a third flight phase.

be greater. Additional considerations occur in between phases when the swing foot collides with the ground which translates to jump discontinuities on the velocities. Other algebraic constraints are in the form of saturation constraints on the state variables and the applied forces. We explain how our numerical procedure is able to produce solutions which can satisfy all of these constraints.

Some of the early work done by Kajita had a surprising amount of success by modeling the biped dynamics as that of an inverted pendulum with point masses [46]. The idea of searching for a passive walking motion which can approximate a motion which requires only the minimal energy was expressed in the work of McGeer [64] and later with Goswami et al. [27]. The minimal energy path is desirable for it exhibits stabilizing, attractive properties. Our experiments have shown that many walking trajectories naively chosen to approximate walking motion can require a huge increase in the needed energy over that of the optimally calculated minimal energy trajectories. Other work also investigating minimal energy biped walking motion, though with simplified dynamical models, may be found in [80, 81].

The dynamics for the biped are calculated using the recursive, symbolic models presented in Chapters II, III, and IV. Our dynamical model explicitly accounts for contact forces with the ground and checks to make sure these forces remain active. Additionally, an inelastic collision is modeled as the swing foot hits the ground which results in a sudden change in the generalized velocities and a drop in kinetic energy. As a result, the Contact and Collision Algorithms presented in Chapter III play an important part in the dynamical calculations. The numerical difficulties normally encountered with differential algebraic systems are also sidestepped with the implementation of the *Reduced Dynamics Algorithm.*

Numerical results are presented here from our extensive experiments on various aspects of minimum energy walking. We have investigated the effects of introducing input ankle torques at the point of contact of the legs with the ground as well as the underactuated case when these torques are not added. Also, an additional impulsive force can be added at the point in time that the swing leg

leaves the ground to aid the body lift off of the ground, create a smoother motion, and simulate the human's use of the foot to create a liftoff force. Several other parameters which can be varied in our model are the biped's step length, the time of one step, and the proportion of time corresponding to the contact phase. We discuss the effect of these parameters on the system energy.

VII.B Human Walking

The human walking step is composed of two different phases, each characterized by a different set of equations of motion. The first phase is the *swing phase* or single support phase when one foot is on the ground while the other swings. This phase makes up the majority of the duration of the walking step in human walking – between 80% and 85%. The second phase is called the *double support phase* as both feet are on the ground while the body is moving forward. This phase usually then makes up only $15 - 20\%$ of the human walking step.

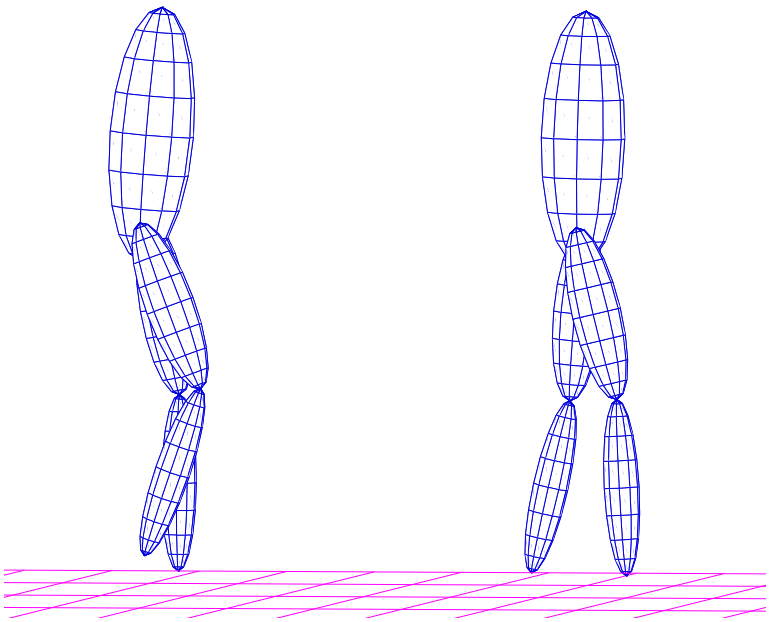

Figure 7.1: Walking Phases

Also of interest are the transitions between phases. Immediately at the beginning of the swing phase is the *moment of liftoff*. Here, the foot is just propelling the body forward so that the leg loses contact with the ground. The other transition between the swing and double support phases is characterized by a *collision* of the swing foot with the ground. Figure 7.1 gives a graphical depiction of our biped model first in the swing phase, then in the double support phase.

To avoid any confusion, some more detailed definitions can be useful. The *cadence* is defined as the number of steps in a standard time frame (e.g. steps/min). The *step length* is the distance between the same point on each foot during the double support phase. The *stride length*, on the other hand, is the distance traveled between two successive foot strikes of the same foot and is equal to two step lengths. Each stride is, thus, composed of one right and one left step length. All measurements given will be in meters.

Experiments have shown humans to walk in an energy efficient manner. In [79], a detailed study is presented of energy expenditure in actual human walking, where researchers have explored the relationships which exist between real energy data and human walking motion. A simple yet fairly accurate relationship is one in which the energy expended is quadratic in the forward velocity,

$$E_w = 32 + 0.005v^2 \ . \tag{7.1}$$

Here v is the average forward velocity in m/min and E_m is the energy requirement in $cal/kg/min$ for the average human subject. An often more desirable set of units for walking energy is to measure it per distance traveled (m = meters) rather than per time elapsed (min = minutes) since this conveys more the notion of energy economy. We denote this form of energy as E_m. Its units are $cal/kg/m$, and it is related to E_w and the previous relationship by

$$E_m = \frac{E_w}{v} = \frac{32}{v} + 0.005v \ . \tag{7.2}$$

This function will now have a hyperbolic shape. This and most other functional relationships such as (7.2) indicate a minimum energy motion of an 80 m/min

walking velocity with an energy expenditure of 0.8 *cal/kg/m*. The experiments also show average cadences of 105 *steps/min* and an average step length of 0.75 *m* for an adult male.

VII.C Biped Model

We believe we are able to capture many of the essential characteristics of the human walking motion with a 5-link planar biped which walks in the two-dimensional sagittal plane, the vertical plane bisecting the front of the biped. The model contains two links for each leg plus a large, massive torso which also functions as the base of the tree-structured multibody system. Though the motion is constrained to the 2-dimensional vertical sagittal plane, the links are modeled with a 3-dimensional elliptical shape and a uniform distribution of mass. The physical data corresponding to the model which we will use as a basis for our experiments can be found in Table 7.1.

Table 7.1: Biped Model Physical Data

Link	Mass	Length	Radius
Torso	20 kg	0.72 m	0.12 m
Upper Leg	7 kg	0.50 m	0.07 m
Lower Leg	4 kg	0.50 m	0.05 m

The influence of the feet may be modeled in ways which do not increase the dimension of the system. The two main contributions of the feet to the control of the biped, when not expressly considering friction, are the introduction of ankle torques and the liftoff force produced by the heel coming off the ground together with the foot rolling over the ball of the foot. It is possible to include ankle torques in the model by treating these as external forces influencing the tips of each leg at the points of contact as described in Section III.E. Rather than modeling a liftoff force which lasts the entire duration of the double contact phase as is the case with the foot, we model the liftoff force as an instananeous impulsive force occurring at

the moment of liftoff. This last technique has certain numerical advantages though it cannot completely reproduce the effect of the foot as will be shown in the reports of our numerical experiments.

There exists four points of actuation in the biped model: one applied torque at each hip and one at each knee. If ankle torques are additionally added, then there will be two more applied torques, one at the point of contact of each leg with the ground. The bounds on the ankle torques are generally smaller than those of the joints as the ankles cannot provide as great a force as at the hips and knees. The state consists of fourteen states, seven position variables and seven velocity position variables. As the motion is completely modeled in the two-dimensional sagittal plane, the torso's position is described by one orientation angle relative to the inertial frame plus an x and y position coordinate represented in the torso's local coordinate system. The remaining position coordinates are the angles of the upper and lower legs relative to their predecessor links. The description of the state and control variables are as follows:

x_1 torso orientation angle relative to the ground.

x_2 x-position of the torso hip (axis which connects the joints of both legs) relative to the origin of the inertial frame in torso local coordinates

x_3 y-position of the torso hip relative to the origin of the inertial frame in torso local coordinates

x_4 torso orientation angle velocity

x_5 x-velocity of torso hip in torso local coordinates

x_6 y-velocity of torso hip in torso local coordinates

x_7 hip angle of leg 1 relative to torso

x_8 hip angle velocity of leg 1 relative to torso

x_9 knee angle of leg 1

x_{10} knee angle velocity of leg 1

x_{11} hip angle of leg 2 relative to torso

x_{12} hip angle velocity of leg 2 relative to torso

x_{13} knee angle of leg 2

x_{14} knee angle velocity of leg 2

u_1 applied torque at hip of leg 1

u_2 applied torque at knee of leg 1

u_3 applied torque at hip of leg 2

u_4 applied torque at knee of leg 2

u_5 applied torque at ankle of leg 1

u_6 applied torque at ankle of leg 2

VII.D Biped Dynamics

The state equations of the biped walker are of the same form as that of the forward dynamics of a multibody system experiencing contact forces.

$$\ddot{\theta} = \mathcal{M}^{-1}(u + J_c^T f_c - \mathcal{C} - \mathcal{G}) \ . \tag{7.3}$$

In equation (7.3), $\mathcal{M}$ is the square, positive-definite mass-inertia matrix, $\mathcal{C}$ is the vector of Coriolis and centrifugal forces, $\mathcal{G}$ is a vector of gravitational forces, u are the applied torques at the links, J_c is the constraint Jacobian, and f_c is the constraint force.

Our 5-link biped model has 7 degrees of freedom (and therefore a 14-dimensional state space) when not under any constraints. Already in this case, recursive, symbolic dynamical models are more efficient for calculating the forward dynamics than other non-recursive procedures which require constructing and inverting the entire mass-inertia matrix, $\mathcal{M}$. $\mathcal{O}(\mathcal{N})$ recursive algorithms, where $\mathcal{N}$ is the number of degrees of freedom of the system, are advantageous for superior calculational efficiency with increasing degrees of freedom. Due to the symbolic nature

of the algorithm, another of the benefits of this modeling approach is that changes made to the kinematic or dynamical parameters simply translate to a different initialization of the algorithm. No extra calculations are required when parameter changes are made.

VII.D.1 Dynamics with Contact and Collision

As we are dealing with a tree-structured system, we evaluate the forward dynamics using the algorithm presented in Figure 2.2. The contact forces experienced at the tips of the legs are calculated with the Contact Algorithm from Chapter III. Similarly, the Collision Algorithm from Chapter III determines the impulse force experienced at the moment of collision which produces a sudden change in the joint velocities of the system.

In our numerical trials, we will solve for symmetric, periodic gaits which will allow us to study just the window consisting of one step. We can then restrict our attention to just one swing phase followed by one double support phase. The step begins in phase 1 with leg 2 just leaving the ground to begin the swing phase and finishes at the end of phase 2 with leg 1 finishing the double support phase in the same relative initial position as leg 2 when the step began. In phase 1, we need only account for one contact force vector for the ground acting on the support leg while in phase 2, one contact force vector needs to be calculated for each leg. A contact force vector will have two nonzero components in the x and y directions corresponding to the two constraints that a foot not move in either the vertical or horizontal directions. The final joint accelerations will result in a zero linear acceleration in the vertical and horizontal directions for each contact leg tip. Figures 7.2 and 7.3 displays the beginning and end configurations in addition to the contact force vectors experienced during the two phases.

Between phase 1 and phase 2, the swing leg makes contact with the ground and a collision occurs. Not only is an impulse force calculated at the collison leg, but also one must be calculated at the leg which was already in contact at the time

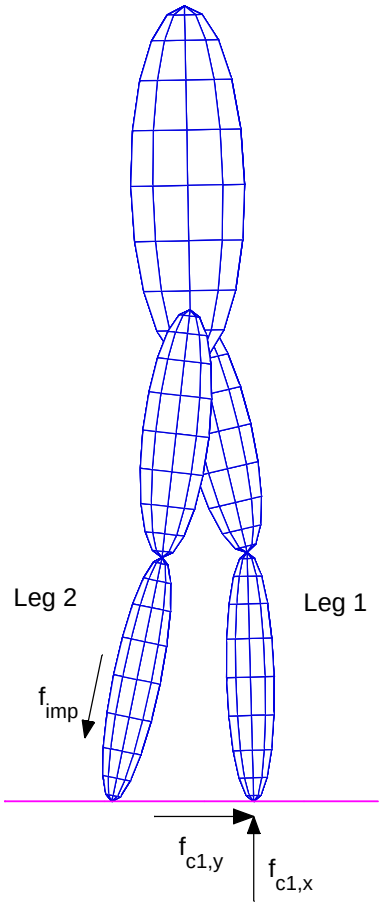

Figure 7.2: Beginning of phase 1. An impulse force f_{imp} helps to propel the body forward. The contact force f_{c1} is applied to the tip of leg 1.

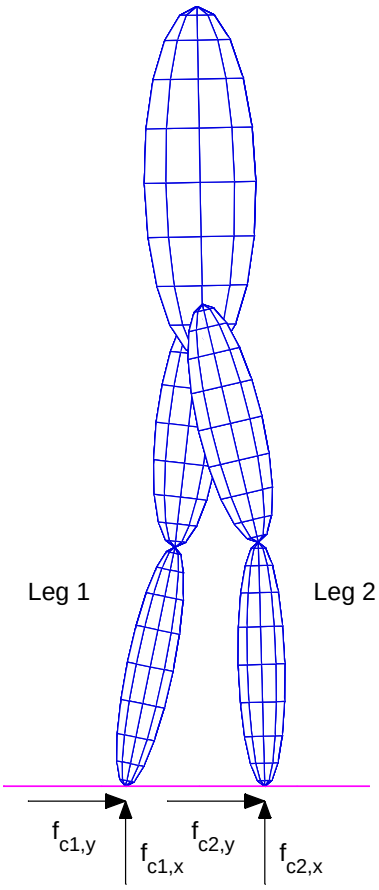

Figure 7.3: End of phase 2. The contact forces f_{c1} and f_{c2} enforce the contact constraints for legs 1 and 2 during phase 2.

of collision. If one were not introduced at this other leg, then the force of collision would cause this leg to have a sudden nonzero tip velocity. Similar to the contact forces, the sudden jump in joint velocities from collision will result in zero linear velocities for both leg tips in the vertical and horizontal directions.

An impulsive force may not only be introduced to model the effects of inelastic collisions, but it may also be used as a form of *control* to propel the system forward. We can use the same Collision Algorithm to *introduce* an impulsive force

during the walking motion to simulate the effect of a foot giving the body a liftoff force to initiate the swing phase. This force is introduced at the beginning of phase 1 at the moment of takeoff. The control impulsive force vector has only one nonzero component since its direction is restricted to lie along the major axis of the lower leg. Figure 7.2 displays the direction of the impulse liftoff force vector. In this way, the impulsive force serves as a thruster to aid the body gain enough velocity to reach the point where it can begin to fall forward again. This function in humans is served by the heel of the foot raising off of the ground while the foot rotates around the ball of the foot.

VII.D.2 Reduced Dynamics

A key component of our dynamical modeling is the use of the Reduced Dynamics Algorithm presented in Chapter III. We have already mentioned that because of the contact constraints we are faced with a differential-algebraic system. Two courses of actions are possible when it is necessary to integrate the dynamics, one being the use of specially tailored integration routines which often require the partial derivatives of the various contact constraints. The preferable approach, however, is to use a reduced unconstrained set of dynamics which evolve on the constraint manifold. Then it is possible to use standard integration procedures.

Essentially, the holonomic contact constraints in our model allow the system to be described by another set of unconstrained dynamics of reduced dimension. Because they are unconstrained, integration of the reduced dynamics produces much less numerical difficulties. This algorithm allows us to evaluate the reduced dynamics in a very efficient manner. There then exists a direct correspondence to the full set of dynamics by which one can obtain the full state from the reduced state.

In the first single contact phase of the biped motion, the contact constraint reduces the total degrees of freedom from 7 to 5. Thus, using the Reduced Dynamics Algorithm, an unconstrained 10-dimensional state space can represent the system

during this period instead of the full 14 dimensions plus algebraic constraints. The remaining 4 dependent states and their time derivatives can be determined from the 10 independent states and their time derivatives. The independent states are those corresponding to the torso and to leg 2, namely x_i, $i = 1, \ldots, 6,\ 11, \ldots, 14$.

Recall that one of the primary difficulties of the Reduced Dynamics Algorithms is that the inverse kinematics must be used to solve for the dependent states. For the biped, this is easy since our problem is equivalent to solving for the joint angles of a 2-link manipulator when its endpoints are known. The following well-known solution comes from Spong and Vidyasagar [87]. In Figure 7.4 is displayed the inverse kinematics problem where θ_1 and θ_2 are the desired angles, a_1 and a_2 are the lengths of the upper and lower legs respectively, and α_1 and α_2 are two intermediate angles. We assume that one end of the 2-link arm has been transferred to the origin while the other end has coordinates (x, y). From the Law of Cosines,

$$D = \cos(\theta_2) = \frac{x^2 + y^2 - a_1^2 - a_2^2}{2a_1 a_2} \ . \tag{7.4}$$

Also, $\sin(\theta_2) = \pm\sqrt{1 - D^2}$. In our case, since the knees only bend in one direction, the minus sign is always used so that

$$\theta_2 = \tan^{-1} \frac{-\sqrt{1 - D^2}}{D} \ . \tag{7.5}$$

It is apparent that θ_1 may be obtained easily from α_1 and α_2. The angle $\alpha_1 = \tan^{-1}(y/x)$ while $\alpha_2 = \tan^{-1}\left(\frac{a_2 \sin\theta_2}{a_1 + a_2 \cos\theta_2}\right)$. The final expression for θ_1 is

$$\theta_1 = \tan^{-1}(y/x) - \tan^{-1}\left(\frac{a_2 \sin\theta_2}{a_1 + a_2 \cos\theta_2}\right) \ . \tag{7.6}$$

The dependent position states then are $x_7 = \theta_1 - x_1$ and $x_9 = \theta_2$.

Also, the velocities are known at both ends of the 2-link manipulator. The Collision Algorithm can then be used to give the unique joint velocities which satisfy the known velocity constraints. This is done by initializing the algorithm for a 2-link manipulator with the known spatial velocity at the joint connecting the upper leg and the torso. The joint angles have been previously calculated and

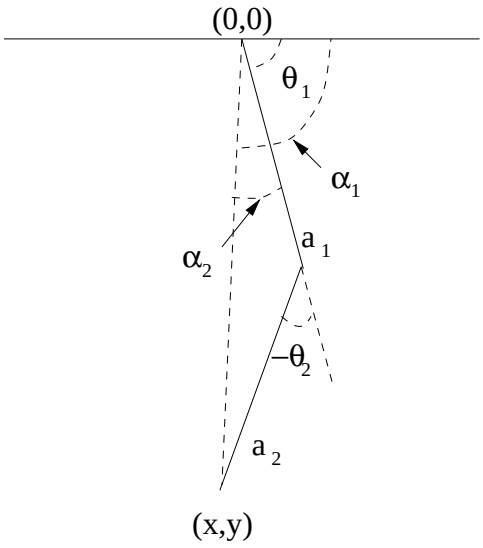

Figure 7.4: Inverse Kinematics Problem for 2-link Leg

the joint velocities are arbitrary as there will only be one solution. The updated velocities will then corroespond to the values of the states x_8 and x_{10}.

In the second double control phase, the addition of contact constraints on the other leg has the effect of further reducing the degrees of freedom so that we can work with a system with only a 6-dimensional state space. The independent states will be only the six states corresponding to the torso. The joint angles for each leg may be derived from the above-mentioned inverse kinematics procedure. The Collision Algorithm also gives the joint angle velocities for each leg as was the case in phase 1.

In summary, if ξ_1 and ξ_2 equals the independent position and velocity state variables during phase 1, and ζ_1 and ζ_2 similarly represent the independent variables in phase 2, then the reduced dynamical equations for both phases of the

walking step are

$$\begin{array}{rclcrcl} & \text{Phase 1} & & & & \text{Phase 2} & \\ \dot{\xi}_1 & = & \xi_2 & & \dot{\zeta}_1 & = & \zeta_2 \\ \dot{\xi}_2 & = & f_1(\xi_1, \xi_2, u) & & \dot{\zeta}_2 & = & f_2(\zeta_1, \zeta_2, u) \\ \xi_1 & = & [x_1,\, x_2,\, x_3,\, x_{11},\, x_{13}]^T & & \zeta_1 & = & [x_1,\, x_2,\, x_3]^T \\ \xi_2 & = & [x_4,\, x_5,\, x_6,\, x_{12},\, x_{14}]^T & & \zeta_2 & = & [x_4,\, x_5,\, x_6]^T \end{array} \tag{7.7}$$

where $u \in R^6$ when ankle torques are included in the biped model and $u \in R^4$ in a model without ankle torques. The dynamical expressions for f_i represent the reduced dynamics which are of lower dimension than equation (7.3) and account for the contact constraints.

$$\begin{aligned} f_1(\xi_1, \xi_2, u_1) &= \mathcal{M}_\xi^{-1}(u_{\xi,1} - \mathcal{C}_\xi - \mathcal{G}_\xi) \\ f_2(\zeta_1, \zeta_2, u_2) &= \mathcal{M}_\zeta^{-1}(u_{\zeta,1} - \mathcal{C}_\zeta - \mathcal{G}_\zeta) \end{aligned} \tag{7.8}$$

The Reduced Dynamics algorithm evaluates $\dot{\xi}_2$ and $\dot{\zeta}_2$ using the forward dynamics algorithm and the Contact Algorithm.

VII.E Minimum Energy Performance

It is generally accepted that humans follow closely an energy efficient motion strategy [79]. In accomplishing our goal of approximating the human walking behavior, it is natural to also have these same performance objectives for our biped model. Our problem is a control problem as we need to establish the values for the applied joint forces which will cause the system to move in an energy efficient manner. To minimize the injected energy into the system, we choose as our performance the integral of the applied joint forces,

$$J = \int_0^T u^T u \, dt = \int_0^{T_1} u_1^T u_1 \, dt + \int_{T_1}^T u_2^T u_2 \, dt \ , \tag{7.9}$$

where T_1 is the time at the end of the first phase, and T is the time at the end of the second phase. The vector u_1 are the applied forces at time t during phase 1 while u_2

are the applied forces during phase 2. The above quantity is not a measure of the mechanical work performed on the system, nor are we able to determine the change in energy of the body from its value. This is rather a measure of the required cost in energy to move the system. The change in energy may reflect the work done by the electric motors which actuate the various joints of the biped. For a simple actuation model (as for a commonly used direct drive DC motor), its units will also be in Joules *(J)*, the units of energy, up to a system-dependent proportionality constant. This approach provides a more numerically tractable way of reaching our performance objectives.

If an impulsive force is introduced at the moment of liftoff, $u_{imp,1}$, this force should also be included in the performance. This is an instantaneous force which enters the performance outside of the integral.

$$J = u_{imp,1}^T u_{imp,1} + \int_0^{T_1} u_1^T u_1 \, dt + \int_{T_1}^{T} u_2^T u_2 \, dt \tag{7.10}$$

This general form of minimum energy performance was also used in [81].

Because we are modeling the collision of the foot with the ground as completely inelastic, the system will experience a sudden drop in the kinetic energy at the moment of collision. Since the potential energy is not affected by the collision, the total energy will also drop by the same amount. Figure 7.12, which is found in Section VII.G on our numerical experiments, displays a typical plot of the kinetic, potential, and total energies all centered around zero of a typical minimum energy walking step of a biped robot. The kinetic energy is responsible for most of the trend seen in the total energy as the potential energy varies less. The energy that is lost at collision must all be injected back into the system over the remainder of the step with the input torques. By minimizing the integral of the squared magnitude of the control torques over the step, we are also minimizing the loss of energy experienced at collision and, thus, working towards conserving as much energy as possible during the step.

VII.F Numerical Optimal Control

Direct optimization methods for optimal control are characterized by the minimization of a cost functional which is a function of the system state and the control input u. An example of such a method is the program DIRCOL [94, 95], which can handle implicit or explicit boundary conditions, arbitrary nonlinear equality and inequality constraints on the state variables, and multiple phases where each phase may contain a different set of state equations. DIRCOL functions by packaging the optimal control problem along with its constraints into a constrained, nonlinear minimization problem which is solved by an SQP-based optimization code NPSOL (Gill, Murray, Saunders, Wright [26]).

DIRCOL discretizes the state and control variables in time over the trajectory. The fineness or coarseness of the discretization can have a large influence over the time required to generate a solution. A new version using sparse optimization techniques should be much faster. The output of the numerical optimal control program will be the optimal open-loop solution for the control $u(t)$ and the corresponding state trajectory $x(t)$ at the choice of grid points in time. With regards to the biped walking control problem, the final values of $u(t)$ and $x(t)$ at the end of a step will be symmetrically constrained to match the initial values for u and x at the beginning of the step. This produces an optimal periodic solution for the desired walking trajectory.

We discussed the use of direct minimization techniques in Section V.C of Chapter V. This numerical approach, and particularly DIRCOL, was selected over all the others for the ease with which many different kinds of constraints can be added to the control problem. The numerical procedure is very reliable and also informative when the program cannot arrive at an optimal solution. DIRCOL has the capability to define unknown constant parameters which may be allowed to vary within a specified range. These parameters may be used to additionally optimize over the step length, final time, and/or forward velocity.

Some examples of the very nonlinear constraints that the final solution must satisfy and which must be inputted into DIRCOL are that the foot land at the specified step length when this is fixed or at the value set for the parameter when it is variable. The vertical component of the contact forces experienced by the feet during each of the phases must also remain positive, otherwise the feet will leave the ground and a different set of equations will govern the system dynamics. Finally, the vertical component of the contact force of the foot that is about to leave the ground must reach zero at the end of phase 2 in preparation for liftoff. We now outline more explicitly the various constraints of the optimization problem.

VII.F.1 Box Constraints and Polar Position Coordinates

The simplest constraints imposed on the problem are the box constraints. These are magnitude constraints on the state, control, and parameter variables. It is always preferable to choose these form of constraints whenever possible as they are much more numerically tractable than general inequality constraints which are nonlinear functions of the states, controls, and parameters. Through a change of coordinates it is sometimes possible to reformulate a nonlinear inequality constraint as a simple box constraint with a different set of variables. This simple trick we utilize in the biped control problem.

Recall that there exist two position variables describing the x and y position coordinates for the torso and, consequently, the entire biped robot. These position coordinates are represented in the local torso coordinate system. One of the nonlinear inequality constraints which we would normally be forced to impose is that the hip remain at a distance from the origin no greater than the length of an extended leg. This requirement affects leg 1 which supports the body during the swing phase. It is important since with the use of the Reduced Dynamics Algorithm the position and velocity variables for leg 1 are not part of the state used in the optimization process. Their values must be calculated via inverse kinematics and the Collision Algorithm every time the dynamics need to be evaluated. If the hip

is too far from the origin, then we will not have sufficient information to determine the state of leg 1 plus the system will have entered a free-flying configuration during phase 1 or a single-support configuration during the double support phase.

By converting these Cartesian coordinates to polar coordinates r and θ, it is then possible to place a simple magnitude constraint on r which will serve the same function as the nonlinear inequality constraint previously mentioned. Define the following variables

$$\begin{aligned} x'_2 &= r = \sqrt{x_2^2 + x_3^2} \\ x'_3 &= \theta = \tan^{-1}(x_3/x_2) \\ x'_5 &= (x_2 x_5 + x_3 x_6)/x'_2 \\ x'_6 &= (x_2 x_6 - x_5 x_3)/x'^2_2 \end{aligned} \tag{7.11}$$

where x'_5 and x'_6 are the respective inertial time derivatives of x'_2 and x'_3. These alternative states will replace x_2, x_3, x_5, and x_6 in the numerical optimal control program. We need to use linear Cartesian coordinates when evaluating the dynamics so that before this is done we must make the substitutions

$$\begin{aligned} x_2 &= x'_2 \cos x'_3 \\ x_3 &= x'_2 \sin x'_3 \\ x_5 &= x'_5 \cos x'_3 - x'_2 x'_6 \sin x'_3 \\ x_6 &= x'_5 \sin x'_3 + x'_2 x'_6 \cos x'_3 \end{aligned} \tag{7.12}$$

After evaluating the dynamics and before passing the time derivative of the state to the integration routine, we again make the substitutions

$$\begin{aligned} \dot{x}'_2 &= x'_5 \\ \dot{x}'_3 &= x'_6 \\ \dot{x}'_5 &= (x_2 \dot{x}_5 + x_5^2 + x_3 \dot{x}_6 + x_6^2)/x'_2 - (x_2 x_5 + x_3 x_6) x'_5 / x'^2_3 \\ \dot{x}'_6 &= (x_2 \dot{x}_6 - x_3 \dot{x}_5)/x'^2_2 - 2(x_2 x_6 - x_5 x_3) x'_5 / x'^3_3 \end{aligned} \tag{7.13}$$

During the optimization trials to be discussed later, we typically place the

following box constraints on the state and control variables:

$$\begin{array}{rcrcr}
1.35 & \leq & x_1 & \leq & 1.60 \\
0.80 & \leq & x_2' & \leq & 0.998999 \\
-3.00 & \leq & x_3' & \leq & 3.00 \\
-25.00 & \leq & x_4 & \leq & 25.00 \\
-25.00 & \leq & x_5' & \leq & 25.00 \\
-25.00 & \leq & x_6' & \leq & 25.00 \\
-4.00 & \leq & x_{11} & \leq & -2.60 \\
-25.00 & \leq & x_{12} & \leq & 25.00 \\
-1.00 & \leq & x_{13} & \leq & -0.0895 \\
-25.00 & \leq & x_{14} & \leq & 25.00 \\
0.00 & \leq & x_{15} & \leq & 10^8
\end{array} \tag{7.14}$$

$$\begin{array}{rcrcr}
-30.00 & \leq & u_1 & \leq & 30.00 \\
-30.00 & \leq & u_2 & \leq & 30.00 \\
-30.00 & \leq & u_3 & \leq & 30.00 \\
-30.00 & \leq & u_4 & \leq & 30.00 \\
-15.00 & \leq & u_5 & \leq & 15.00 \\
-15.00 & \leq & u_6 & \leq & 15.00
\end{array} \tag{7.15}$$

These box constraints will be the same in both phases with the exception of u_6 which refers to the ankle torque of the swinging leg in phase 1. Thus, we replace its box constraint with

$$0.00 \leq u_6 \leq 0.00 \tag{7.16}$$

during phase 1.

The additional state variable x_{15} refers to the value of the integral (7.9). For numerical reasons, the Mayer objective functional approach is taken where the optimization problem is to minimize $x_{15} + u_{imp,1}^T u_{imp,1}$ with $\dot{x}_{15}$ being the integrand of (7.9).

VII.F.2 Inequality Constraints and Boundary Conditions

We give the boundary conditions for one step. For the beginning and final time, if $R(x_1)$ is the rotation matrix mapping inertial coordinates to torso coordinates and *step* is the length of the step taken, let $hp = [hp_1\ hp_2\ hp_3]^T = R(x_1)[step\ 0\ 0]^T$. This quantity is the displacement of the hip in torso coordinates. Then, the **explicit initial and final time boundary conditions** are:

$$\begin{aligned} x_1^+ &= x_1^- & u_1^+ &= u_3^- \\ x_2^+ &= x_2^- + hp_1 & u_2^+ &= u_4^- \\ x_3^+ &= x_3^- + hp_2 & u_3^+ &= u_1^- \\ x_4^+ &= x_4^- & u_4^+ &= u_2^- \\ x_5^+ &= x_5^- & u_5^+ &= 0 \\ x_6^+ &= x_6^- & u_6^+ &= u_5^- \end{aligned} \tag{7.17}$$

where the variables with a + are the values at the final time and those with a − at the initial time. The remaining state variables do not need to be specified as the boundary conditions occur at a moment of double contact of the feet with the ground. The dependent positions and velocities can be determined from the first six states.

It was mentioned earlier that it is possible to include in our model an impulsive liftoff force at the beginning of phase 1. When this is done, all the joint velocities in the system will undergo a discontinuous jump as a result of the impulsive action. The jump in the velocities may be calculated as a function of the previous state y and a liftoff impulsive force vector $u_{imp,1}$ as described in Section III.E. Define the function $x^- = IMP(y, u_{imp,1})$ which calculates the state vector x^- after introducing an impulsive force at the beginning of phase 1 to leg 2. The relationship $y = IMP^{-1}(x^-, u_{imp,1}) = IMP(x^-, -u_{imp,1})$ holds which allows us to calculate the right handed limits for the torso velocities:

$$\begin{aligned} x_4^+ &= y_4 \\ x_5^+ &= y_5 \end{aligned} \tag{7.18}$$

$$x_6^+ = y_6$$

These boundary conditions replace those for x_4, x_5, and x_6 in (7.17).

The boundary conditions in between phases enforce continuity of the position state and control variables, and they prescribe the jump in the velocity state variables due to the collision. The function $xc = COLL(x)$ gives the states after collision after determining the required jump in the velocities from the Collision Algorithm. xc is a state vector of the same dimension as x. Then the **explicit boundary conditions in between phase 1 and 2** are:

$$\begin{aligned} u_i^+ &= u_i^- \qquad i = 1, \ldots, 6 \\ x_i^+ &= x_i^- \qquad i = 1, \ldots, 3 \\ x_i^+ &= xc_i \qquad i = 4, \ldots, 6 \end{aligned} \tag{7.19}$$

where here the + indicates the right-hand side limit, at the beginning of phase 2, and the − indicates the left-hand side limit at the end of phase 1.

Implicit boundary conditions are characterized by nonlinear equations to be satisfied by the boundary values for the states, controls, and parameters. The required implicit boundary conditions that we impose on the problem at the initial and final time are that the initial tip position of leg 2 start at the appropriate step length away from leg 1 and with an initial zero tip velocity if no impulsive liftoff control force $u_{imp,1}$ is added. If the impulsive force is added, then the starting velocity will depend on the magnitude of the impulsive force. Let the function $[tp, tv] = TIP(x^-)$ evaluate the current value for the leg 2 (swing leg about to liftoff) tip position and velocity in inertial coordinates at the beginning of phase 1. Both tp and tv are 2-dimensional vectors. Additionally, we must require that the vertical component of the contact force influencing leg 1 be zero at the end of phase 2 so that the leg is ready to leave the ground at the beginning of phase 1. Let $fc = CONT(x^+, u^+)$ be the 6-dimensional spatial contact force in inertial coordinates for leg 1 at the end of phase 2. Then the **nonlinear, implicit boundary conditions**

at the initial and final time are:

$$\begin{aligned} r_1 &= tp_1 + \text{step} &&= 0 \\ r_2 &= tp_2 &&= 0 \\ r_3 &= tv_1 &&= 0 \\ r_4 &= tv_2 &&= 0 \\ r_5 &= fc_5 &&= 0 \end{aligned} \tag{7.20}$$

Note that the vector fc is a spatial vector in which the first three components are the moment forces and the next three components are the linear forces. In our case, we are concerned with the linear, vertical component; thus, we require its fifth entry to be zero.

Now, we consider the alternative case when liftoff impulsive forces are introduced at the very beginning of phase 1. In this case, we remove the effect of the impulsive liftoff force from x^- by using the previously described function $y = IMP^{-1}(x^-, u_{imp,1}) = IMP(x^-, -u_{imp,1})$. Then $[tp, tv] = TIP(y)$ gives the tip position and velocity vector for y. The positions are unchanged from the impulsive force, and the tip velocities for the state y should be 0 when the effect of the impulsive force has been removed from x^-. The boundary conditions then remain the same as in (7.20).

We additionally require that the swing leg land at the appropriate step length at the moment of collision. Let $[tp, tv] = TIP(x^-)$ be the tip position and velocity at the end of phase 1. The **implicit boundary conditions for in between phases** are:

$$\begin{aligned} s_1 &= tp_1 - \text{step} &&= 0 \\ s_1 &= tp_2 &&= 0 \end{aligned} \tag{7.21}$$

As mentioned previously, it is possible to make additional parameters variable such as step length, the time duration of the walking step, and average

forward velocity. Define the following parameters:

$$\begin{aligned} p_1 &= \textit{step length} \\ p_2 &= \textit{magnitude of liftoff impulsive force} \\ p_3 &= \textit{time of collision} \\ p_4 &= \textit{average forward velocity} \\ p_5 &= \textit{proportion of step time corresponding to phase 1} \end{aligned} \tag{7.22}$$

Let T be the final time or time duration of the walking step while $T_1 = p_3$. The value for T is also allowed to be variable though it is not necessary to give it a parameter definition. The relative time of the step during which phase 1 occurs is related to T in the following manner, $p_5 = T_1/T$. This parameter will generally be fixed for the majority of our experiments and set to 0.85 for reasons to be explained later. The previously mentioned problem would then introduce the following **additional implicit boundary conditions for the initial and final times:**

$$\begin{aligned} r_6 &= T - p_3/p_5 &&= 0 \\ r_7 &= p_4 - (p_1 \times 60)/T &&= 0 \end{aligned} \tag{7.23}$$

and **additional implicit boundary conditions for in between phases:**

$$s_3 = p_3 - T_1 = 0 \tag{7.24}$$

Note that if the proportion of total step time for phase 1, p_5, is desired to be fixed, we may set the upper and lower bounds for this parameter to be the same value.

In addition to boundary conditions, we also have nonlinear inequality constraints which must be satisfied during the entire length of either phase. The vertical components of the contact forces experienced by the feet in contact during both phases must be positive as a negative value signifies that the foot is leaving the ground. Also, the swing leg must stay entirely above ground during phase 1. Thus, the **nonlinear inequality constraints** for both phases are:

$$\text{Phase 1:} \quad fc_5(\text{Leg 1}) \geq 0$$

$$
\begin{aligned}
& & tp_2(\text{Leg } 2) &\geq 0 \\
\text{Phase 2:} & & fc_5(\text{Leg } 1) &\geq 0 \\
& & fc_5(\text{Leg } 2) &\geq 0
\end{aligned}
$$

where $fc = CONT(x,u)$ and $[tp, tv] = TIP(x)$. These constraints prevent a premature liftoff of a foot from the ground when it should be, for example, in the double support phase.

The functions used in the constraints, $IMP(x)$, $COLL(x)$, $TIP(x)$, and $CONT(x,u)$, all have spatially, recursive implementations for their evaluation.

VII.G Optimization Trials

The high degree of nonlinearity and high dimension of the problem along with all the constraints make it unreasonable to assume that by specifying the state equations, boundary conditions, and inequality constraints along with a naive initial guess of the solution, that the optimization procedure will immediately find an optimal solution. Optimization procedures are too sensitive to initial guesses. Consequently, an iterated process was undertaken which gradually approximated the actual problem. The first problem solved was merely finding the minimal energy solution to standing in place. Using that solution as an initial guess, it was possible to solve for the problem where the biped was forced to move a small distance. This iterative process continued several times with increasing distances over which the biped traveled.

During the iterative process, constraints were also gradually added such as the double support phase and the collision effect. After about 10 such optimization runs, it was possible to unfix the initial and final boundary conditions for the state and controls and merely require that the solution be periodic. In this manner, the complete problem was solved.

For most trial runs, we used 13 grid points in time, 8 in the first phase and 5 in the second phase. As the number of grid points has a large influence

on the length of each optimization run, it is preferable to use a coarse grid for trial runs, then to refine the grid when very exact solutions are desired. Sample run times varied between 25 and 35 minutes on a Sun Ultra 1 depending upon the complexity of the problem. Failure of the program to converge generally was dependent upon inherent characteristics of the model or of the problem and not of DIRCOL's inability to deal with the nonlinearity and high dimensionality. These occurrences usually led to a rethinking of the model and/or a deeper understanding of the problem.

Many different numerical experiments were made as there are many parameters which can be fixed or varied, and there are modeling flexibilities. Two main categorizations can be made in that we explore first walking without any form of lift propulsion. We then add to our biped the possibility of introducing an instantaneous impulsive force at the moment of liftoff to help the body move forward. This liftoff impulsive force also serves to better simulate the function of the feet which are only implied in our model but do not exist explicitly. The impulsive force also has its cost and is added into the performance. In both settings, the additional effect of the removal of ankle torques is investigated.

The only parameter fixed in our experiments is the percentage of time of the total step that the walker is in the double support phase. We set this parameter to be 15% as this is comparable to observed human data. It is possible with our numerical approach to also optimize over this quantity, and our results shows that a lower energy walking trajectory can be obtained by reducing this proportion. The reason for this effect is that without having the use of the feet in the model, there is little benefit (energywise) to having a double support phase. Impulsive liftoff forces do not counteract this effect as they are instantaneous. The double support phase is desirable for it essentially provides a smoother walk which resembles the human motion. A negative consequence, also, of reducing the proportion of 15% to something smaller is that the optimal control torques will then acquire excessively high control rates.

Though in the following subsections, we will explore global minimum energy walking, there also exist model variations which can be made and which are interesting for comparative purposes. There are four specific cases to be considered and graphically compared:

Model 1 Walking without an impulsive liftoff force and with ankle actuation.

Model 2 Walking without an impulsive liftoff force and without ankle actuation.

Model 3 Walking with an impulsive liftoff force and with ankle actuation.

Model 4 Walking with an impulsive liftoff force and without ankle actuation.

VII.G.1 Models 1 and 2: Walking without liftoff forces

Most everyone is familiar with the useful function of the feet in walking. As soon as the double support phase is entered, the heel of the back leg begins to lift off of the ground, thereby providing a liftoff force for the rest of the body until the point where the ball of the foot breaks contact with the ground. One of the principal differences of our biped model with human walking is that the feet are implicitly, but not explicitly, included in the model. We first explore this interesting case when no liftoff forces besides the hip and knee applied torques are available to help the back leg lift off of the ground.

In Figure 7.5 are displayed the input torques for a complete, periodic, double step. The stick walking figures on the top portion of the figure indicate for the two plots beneath it which part of the walking step the plotted points correspond to. This is done by examining the point, then its position in the step is obtained by observing the stick figure directly above it. The plots begin with the swing leg leaving the ground. At the first vertical line, the swing collides with the ground, and at the second vertical line, the other leg lifts off of the ground. The right-hand side of the plot, however, corresponds to the torques applied to the support leg. In the middle plot are the applied torques to the system with ankle

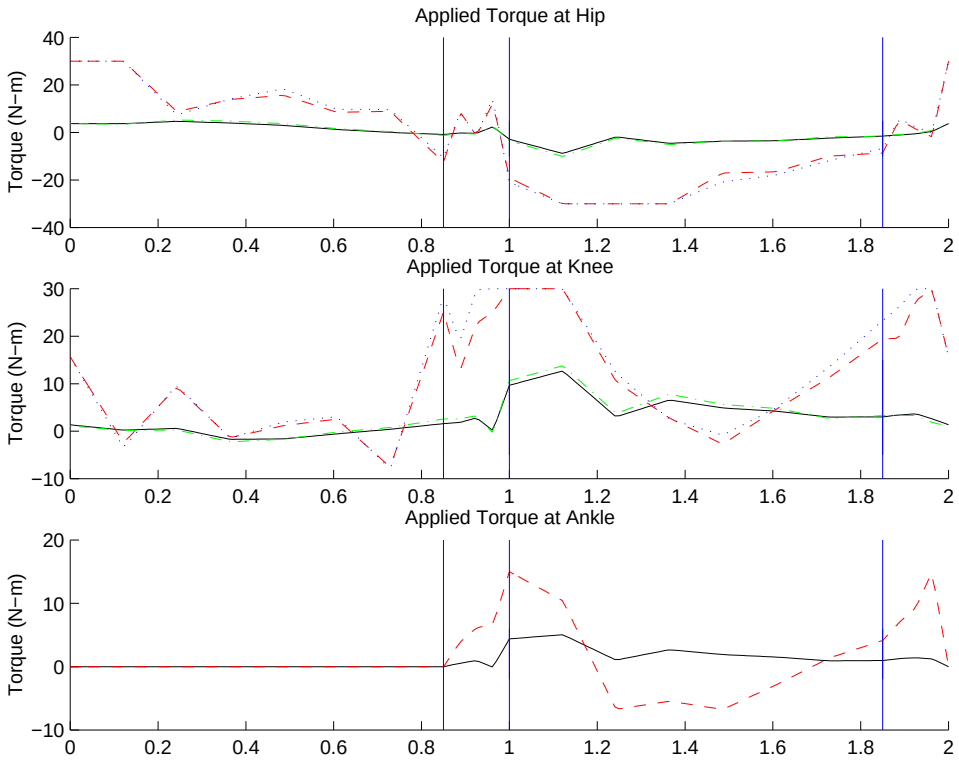

Figure 7.5: Optimal applied torques for walking without liftoff forces. Included extra forces in addition to hip, knee torques for optimal speed: Ankle (solid), None (dashed). Included extra forces for 50 m/min speed: Ankle (dashdot), None (dotted)

actuation.

The solid and dashed lines indicate the torques for the optimal walking motion for the model with and without ankle actuation respectively when the only parameter fixed is the proportion of time for the double support phase. This is fixed at 15%. The paths for these lines are seen to be almost identical signifying very little overall difference for when ankle torque actuation is included.

Also displayed in Figure 7.5 are control trajectories for the optimal walking motion when the average forward velocity is fixed at 50 m/min, also with (dashdot) and without (dotted) ankle actuation. The faster 50 m/min walk also shows little

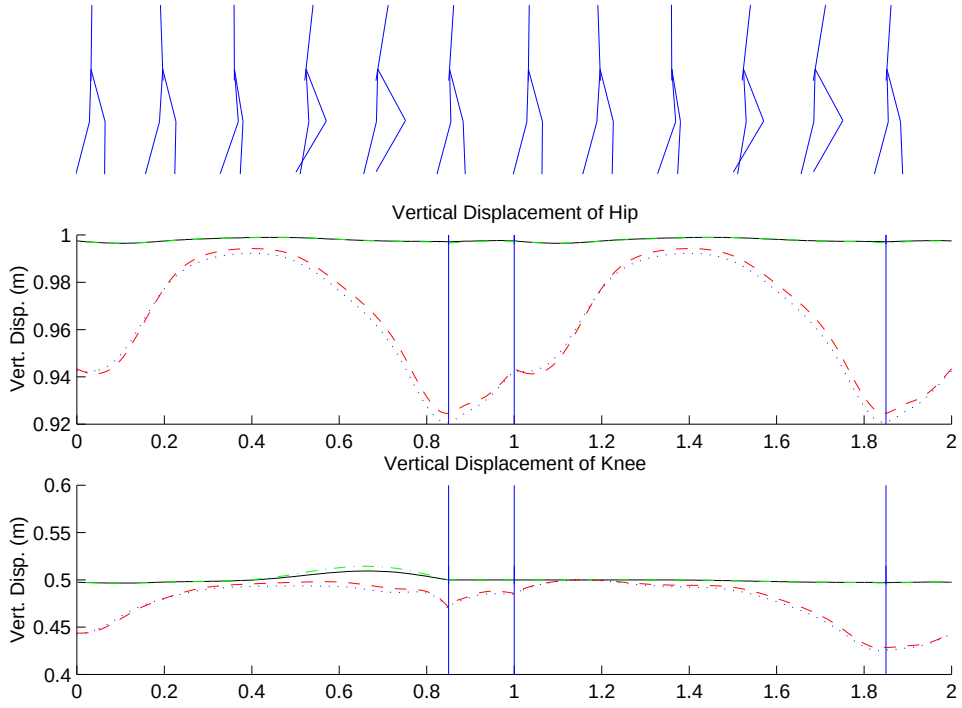

Figure 7.6: Optimal trajectories of hip, knee for walking without liftoff forces. Included extra forces in addition to hip, knee torques for optimal speed: Ankle (solid), None (dashed). Included extra forces for 50 m/min speed: Ankle (dashdot), None (dotted)

difference between including ankle actuation and not including it. This effect is universal throughout out experiments. There does exist, however, a significant difference between the slower globally optimal torques which are smoother and of smaller magnitude. The greater forces display at what times during the step the walking propulsion forces are most necessary.

In Figures 7.6 and 7.7, the movement of the hip and knee is analyzed. The height of the hip and knees stays roughly at the same level during the slower globally optimal walk while during the faster walk, the well-known sinusoidal motion effect of the hip is more apparent [79]. Interesting is also the hip forward velocity displayed in Figure 7.7. There is a large variation in the forward velocity of the faster walk

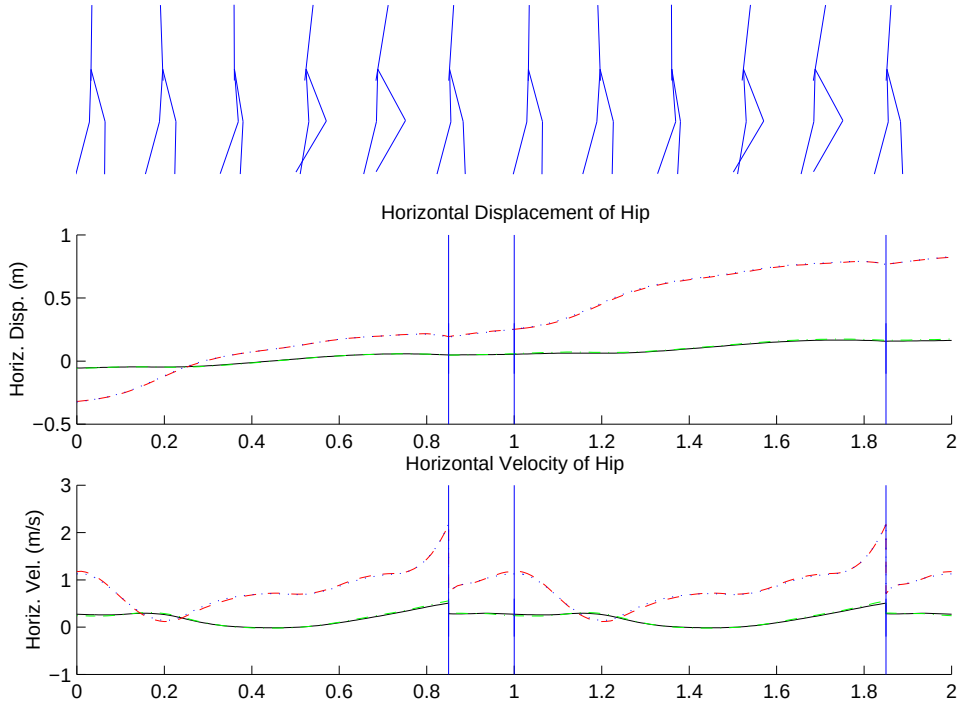

Figure 7.7: Optimal trajectories of hip position and velocity for walking without liftoff forces. Included extra forces in addition to hip, knee torques for optimal speed: Ankle (solid), None (dashed). Included extra forces for 50 m/min speed: Ankle (dashdot), None (dotted)

indicating that the system is not able to maintain a high velocity but must spend a great deal of energy during each step to reach the higher speed.

VII.G.2 Models 3 and 4: Walking with liftoff forces

In human walking, the body propels itself off of the ground with the foot and gains potential energy before it starts to fall forward to the point where the swing leg collides with the ground. To simulate this effect, we add an instantaneous impulsive force at the moment of liftoff. This force can add the needed upward velocity to the back leg and torso to more easily reach the point where the body begins to fall forward. The impulsive force is a linear force and has only one

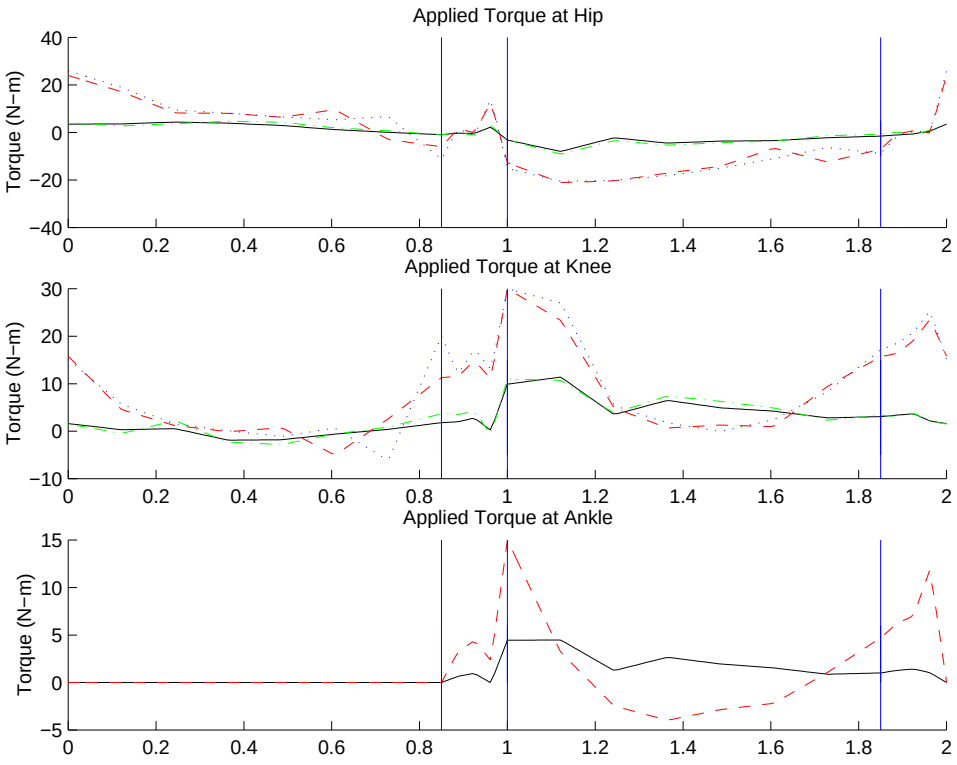

Figure 7.8: Optimal applied torques for walking with liftoff forces. Included forces in addition to hip, knee torques and impulsive force for optimal speed: Ankle (solid), None (dashed). Included forces for 50 *m/min* speed: Ankle (dashdot), None (dotted)

nonzero component along the axis parallel to the lower leg about to come off of the ground. The cost in terms of energy for introducing this extra control is included into the performance via (7.10). There exists then a tradeoff between the size of the impulsive force to use and using more or less torque forces during the walking step.

Figure 7.8, in comparison with Figure 7.5, provides evidence that the impulsive force is sufficiently beneficial to allow a significant savings in the applied torques required for moving at the optimal 50 *m/min* walk. The applied torques are much smoother and of smaller magnitude; particularly the hip torques which

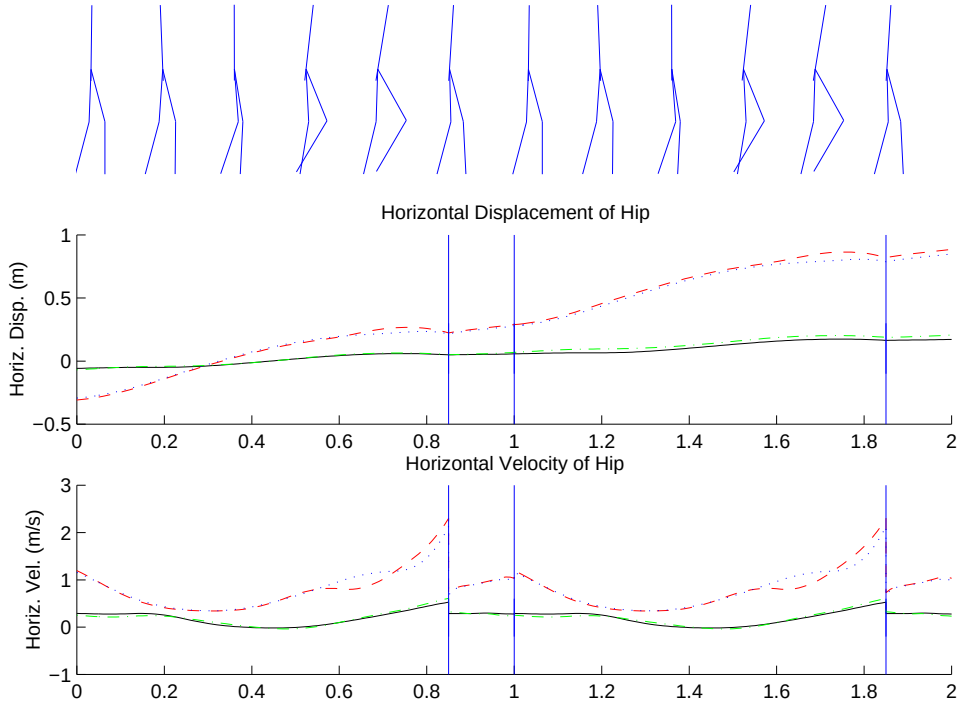

Figure 7.9: Optimal trajectories of hip position and velocity for walking with liftoff forces. Included extra forces in addition to hip, knee torques for optimal speed: Ankle (solid), None (dashed). Included extra forces for 50 m/min speed: Ankle (dashdot), None (dotted)

are surpringly close to zero for the majority of step. Figure 7.9 also displays how with the use of the impulsive force the body has a more slowly varying average forward velocity. In the cases without an impulsive liftoff force, we saw a quick reduction in the average forward velocity while the body is rising to the peak of the swing phase. With the use of the impulsive force, however, the body does better at keeping its forward momentum.

VII.G.3 Varying Step Length, Forward Velocity, and Final Time

It was mentioned earlier how equation (7.2) describes the energy requirements of human walking motion in that it has been observed that the required

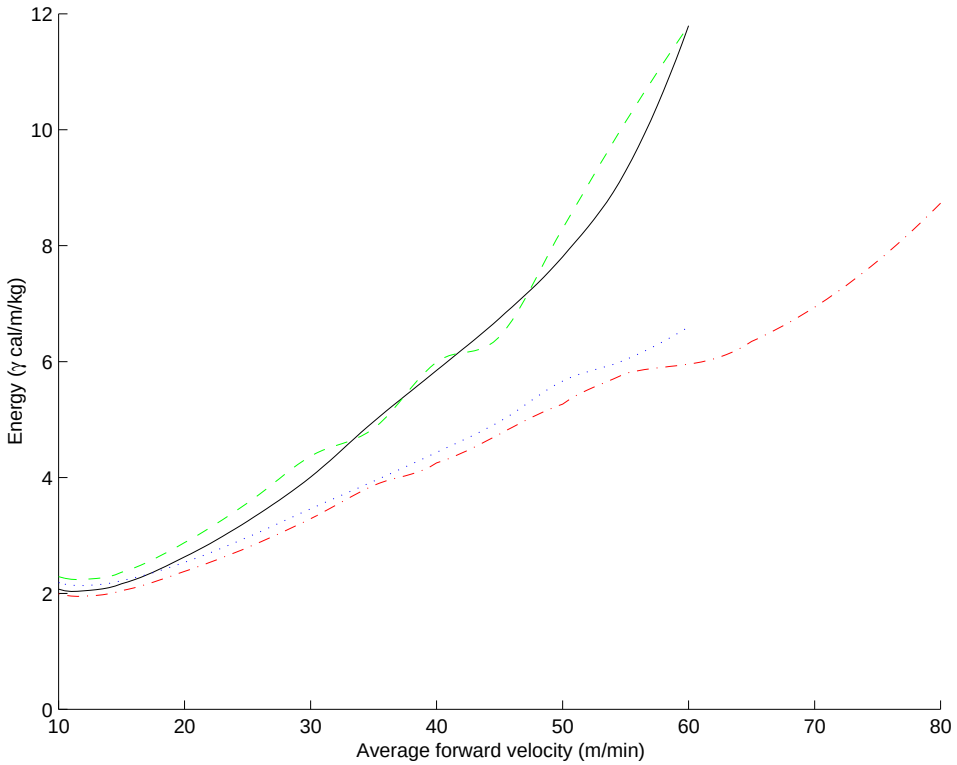

Figure 7.10: Required energy as a function of average forward velocity. Ankle actuation (solid), no ankle forces (dashed), ankle actuation and impulsive force (dashdot), impulsive force with no ankle actuation (dotted).

metabolic energy for walking has a hyperbolic relationship with the average forward velocity. For positive velocities, most experiments show an optimum of 80 *m/min*. In our experiments, we have witnessed the same effect as is displayed in Figure 7.10, though with a very different optimum of approximately 12 *m/min*. Interestingly, the optimum was almost the same for the four different models considered. The value of energy on the vertical axis of Figure 7.10 is obtained by dividing the performance (7.9) or (7.10) by 4.186 to get calories (up to an actuator-dependent proportionality constant), by the total mass of 42 *kg* for our model to get *cal/kg*, and by the final value for the step length to finally get *cal/kg/m*.

A possible conjecture for the disparity with optimal human walking is the

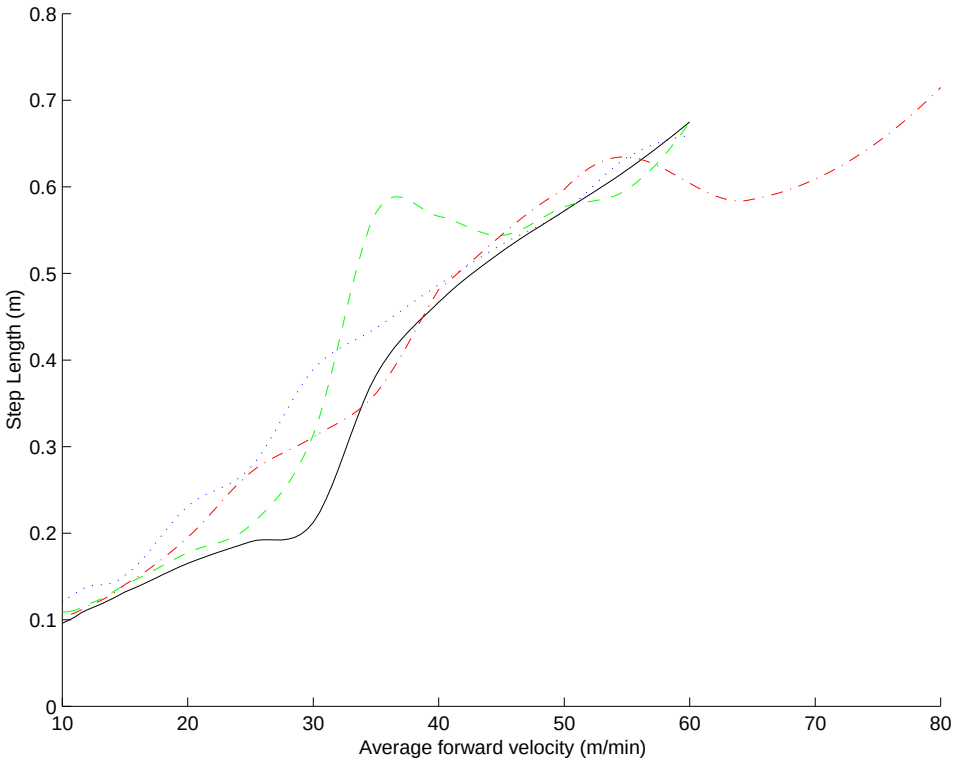

Figure 7.11: Step length as a function of average forward velocity. Ankle actuation (solid), no ankle forces (dashed), ankle actuation and impulsive force (dashdot), impulsive force with no ankle actuation (dotted).

lack of the foot effect which provides essentially an extension of the leg when the back heel lifts off of the ground propelling the body forward. Walking with the added impulsive force is seen to provide a significant energy savings, particularly with higher velocity.

Finally, Figure 7.11 displays how the optimal size of the step increases with increasing average forward velocity. The overall trend is very close for all the experimental cases considered in that the optimal step length is directly proportional to the average forward velocity.

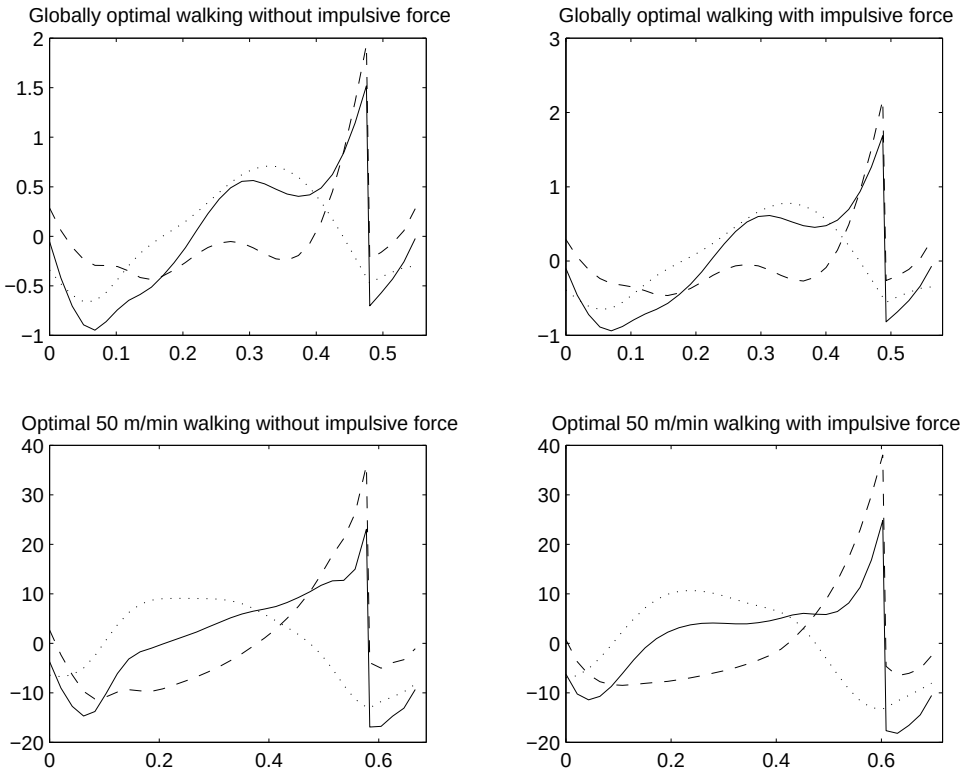

Figure 7.12: Kinetic (dashed), Potential (dotted), and Total (solid) energies centered at 0 during a walking step.

VII.G.4 Optimal Forward Velocities and Energy Requirements

In our investigations, we have been minimizing the integral of the squared applied torques and calling the solution minimum energy walking trajectories. We should emphasize that these solutions do not produce the minimum change in energy of the system as we are not minimizing the mechanical work of the system. In fact, we are minimizing a quantity proportional to the required energy for motion. In humans, this is analogous to the difference between mechanical energy and metabolic energy. As no system, not even a human, is perfectly efficient, these quantities will differ and their relationship in humans still remains a very difficult and unanswered problem [79]. In robotics, we do not have metabolic energy, but there does exist the required energy for direct drive motors at the joints to produce the

required torques. Our chosen performance (7.9) is equal, up to a system-dependent proportionality constant, to the energy used by simple direct drive motors to move the biped robot.

Figure 7.12 displays the kinetic, potential, and total energies all centered around zero of several minimum energy walking steps of a biped robot. The sharp drops in the kinetic and total energies signifies the moment of collision. Before this point, the time when the body reaches its maximum height and begins to fall is indicated by the peak of the potential energy curve. The left plots are without impulsive liftoff forces, while on the right they are modeled. The top plots are globally optimal walking steps, and the bottom are 50 *m/min* optimal walking steps. In all cases, ankle actuation is modeled.

The faster walking steps exhibit a much larger variation in its energy than globally optimal walking. The body also stays much longer in positions of high potential energy. For the faster speeds, the body doesn't slow down as it reaches the peak of its potential energy before it begins to fall forward. Rather, it continuously builds its kinetic energy until the moment of collision. Also for the faster speed, the walking step which uses the liftoff impulsive force (bottom-right) has evidently the smoothest and homogenous motion with the fewest oscillations. This particular walking step builds up its kinetic energy, then keeps moving for a long period while keeping the kinetic energy roughly constant.

Chapter VIII

Conclusions and Future Directions

This work gathers together several different research areas in the modeling and control of complex, nonlinear systems. As we stated at the outset, the main goal is to battle the well-known "curse of dimensionality" by providing numerical and algorithmic tools which enable the user to better understand and control such systems. In the areas of articulated multibody systems, efficient, recursive, and symbolic modeling techniques are further developed and refined. Multiple degree-of-freedom joints are expressly considered and valuable insight is also provided for the implementation of these methods. Later, important contributions are made for the class of such systems experiencing contacts and collisions with the environment. An entire section is devoted to the development of closed-form symbolic expressions for the derivatives of the dynamics and its components. Dynamical partial and time derivatives have numerous applications in estimation and control of which some are described in this dissertation.

A new result is the ability to evaluate recursively and efficiently the adjoint equations of a general multibody dynamical system for uses in numerical optimal control. The possibility to even calculate these equations without undergoing the laborious task of constructing the closed-form equations is not yet known in the research community, and it can open up new approaches for solving optimal control problems. Another important result found in the dynamics chapters is the Reduced

Dynamics Algorithm which is designed for use with algebraically constrained dynamical systems. This algorithm can evaluate the unconstrained reduced dynamics which account for the constraints using the same suite of recursive, dynamical algorithms. This is done without having to explicitly construct the reduced dynamics thereby providing an efficient means for the integration of differential-algebraic multibody systems. This algorithm became vital for the solution of the minimum energy biped gait generation problem presented in the last case study.

By bringing together various representations of recursive, multibody algorithms, extending the results to more general types of systems, and exploring new applications for these methods, our hope is to make further progress in bringing together a comprehensive package that can be used for the dynamics, estimation, and control of such systems. The flexibility and ease with which these symbolic equations may be manipulated were shown throughout the dissertation. In addition, its special structure makes many interesting uses of the dynamics possible. The work begun in [30], for example, on exploiting the special structure found in the multibody equations in control design is continuing. We discussed therein a backstepping approach for adaptive control. Work has also begun on using the recursive expressions for the derivatives of the spatial matrix operators of the dynamics to construct a dynamics-based extended Kalman filter estimation algorithm of multibody motion in computer vision.

In the context of nonlinear control, numerical techniques for the solution of nonlinear $\mathcal{H}_2$ and $\mathcal{H}_\infty$ control problems are shown to be viable and feasible for the solution of important practical applications such as the jet engine compressor control problem. The ability to solve such problems has been vastly underestimated in the research community. We provide a thorough description of the state-of-the-art numerical approaches which exist and have been tested on these types of problems, and we propose a particular solution method. Through illustrations and the final two case studies, we demonstrate in a step-by-step manner methodologies by which one can approach and solve various nonlinear control problems.

In the case of the jet engine compressor control problem, our approach exposed the large sensitivities to control rate saturation prevalent in this system. This case study demonstrates a how simple application of the numerical control techniques is not sufficient to completely analyze a control problem. One must consider the characteristics of the system under study and often devise special techniques to handle the difficulties one encounters. Modeling considerations can then become of primary importance. Our results suggest how a compressor model with greater actuation possibilities may be needed to effectively control the severe instabilities of surge and rotating stall.

No problem better exemplifies the complexity of the control of a complex, nonlinear system as does that of our final investigation into biped walking. It lies on the frontiers of nonlinear control with modeling complexities such as contact and collision constraints, differential-algebraic dynamical equations, high dimension, periodic control, and control and rate saturation constraints. In spite of these obstacles, we have been able to use the extensive structure brought together here to solve this problem. Previous research has not solved the problem of minimum energy gait generation with as complete a dynamical model as ours.

There exists many future avenues for expanding our work on the problem of biped walking control since with the structure and framework we have created, it now becomes relatively easy to add new features to our model. Our first step to gain more understanding into biped walking will be to explicitly model the feet to see which effect this has on our global minimum energy walking results. It is believed that with their ability to extend the legs during the leg's liftoff from the ground and their energy damping capabilities at the time of collision, a much more energy efficient walk can be produced. To finally construct such a system, it will be necessary to further investigate model additions such as arms, movement in 3 dimensions, actuator dynamics and friction. Additionally, the challenging problem of closed-loop control must be addressed, though with our solution of the open-loop problem, this becomes already more tractable.