



Human Motion Planning Based on Recursive Dynamics and Optimal Control Techniques

JANZEN LO, GANG HUANG and DIMITRIS METAXAS

*Department of Computer and Information Science, School of Engineering and Applied Sciences,
University of Pennsylvania, 200 South 33rd St., Philadelphia, PA 19104, U.S.A.;*
E-mail: dnm@central.cis.upenn.edu

(Received: 21 March 2000; accepted in revised form: 8 March 2001)

Abstract. This paper presents an efficient optimal control and recursive dynamics-based computer animation system for simulating and controlling the motion of articulated figures. A quasi-Newton nonlinear programming technique (super-linear convergence) is implemented to solve minimum torque-based human motion-planning problems. The explicit analytical gradients needed in the dynamics are derived using a matrix exponential formulation and Lie algebra. Cubic spline functions are used to make the search space for an optimal solution finite. Based on our formulations, our method is well conditioned and robust, in addition to being computationally efficient. To better illustrate the efficiency of our method, we present results of natural looking and physically correct human motions for a variety of human motion tasks involving open and closed loop kinematic chains.

Key words: efficient dynamic simulation, articulated figures, physics-based modeling, optimal control, motion planning.

1. Introduction

The simulation of human motion and the performance evaluation of human in virtual environments is becoming increasingly important in computer animation, mechanical engineering, medical, military and space exploration applications. For example, in space exploration applications the simulation of dynamically correct astronaut movements becomes necessary for astronaut training for both intra- (IVA) and extra-vehicular activities (EVA) [25]. Dynamically correct simulation is important since our terrestrial way of limb-movement is not necessarily correct in a low-gravity environment in order to achieve the same goal. The efficiency of the dynamic simulation is also very important. The human body as an articulated body system has many degrees of freedom. The dynamic/kinematic problem of such a large system, which contains not only open chains but also closed chains, can be modeled as a sub-problem of robotic mechanisms. Previous research in the dynamic simulation of robotic mechanisms includes the study of both open-chain mechanisms [3, 7, 28] and closed-chain configurations [8, 16, 19]. The joint torques are computed during the dynamic simulation. Recently, Schaffner et al.

[25] uses inverse dynamics computation to find joint torques along a given trajectory movement. However, the use of inverse dynamics alone is not sufficient to offer insight to an astronaut to perform a given space mission task. Therefore, motion-planning problems have attracted considerable attention [1]. There are many motion-planning control methods for dynamic systems. The challenge is to find a robust control method based on an efficient dynamics formulation to achieve the best possible computational efficiency.

Industrial ergonomists and astronaut trainers, for example, often need information regarding a worker's posture and trajectory, which satisfies the minimum torque requirement during the course of his/her job. This may result in potentially hazardous postures and motions requiring excessive joint motion torques that can cause injury and should be identified and prevented. Ayoub and Lin [4] developed a biomechanics simulation of humans lifting objects by minimizing joint torques. In the robotics community, Zefran and Kumar [32] applied the variational calculus method developed by Gregory and Lin [15] to the problem of two arms holding and moving an object with unilateral constraints. However, this approach would be inefficient when dealing with large degree of freedom models since (1) the size of the 'extended state variable' vector will increase very fast; and (2) it is based on an explicit evaluation of the gradient of the objective function, thus making the resulting computation very expensive. This approach is not suitable for practical motion planning of articulated human figures in a virtual environment, since it is necessary to reformulate the dynamic equations for different articulated figures.

In computer animation, Witkin and Kass [30] introduced the *spacetime constraint* method based on optimization theory. Cohen extended the *spacetime constraint* method using the concept of windows to allow the animator to assist the optimizer [10]. Rose et al. [24] use this technique for generating transitions between the motions captured. Most methods, which are based on the *spacetime constraint* approach, use nonlinear programming to achieve the minimum of the desired objective function. Since the objective function contains multibody system dynamics computations, its gradient computation generally is difficult, and therefore can be solved numerically through the finite difference method. This may result in an ill-conditioned problem for the convergence of the optimization. Recently, Popovic and Witkin [21] presented a novel algorithm for transforming character animation sequences that preserves essential physical properties of the motion by using the spacetime constraint dynamics formulation. Though the algorithm takes dynamics into consideration, it is well suited for the highly detailed captured motion animations. Moreover when mapping the motion change of the simplified model back onto the original motion to produce a final animation sequence, it may lose some of the dynamics properties of motion. In addition, it does not address the problem of dynamics computation when the objective function includes joint torques. Finally, existing work on *spacetime constraint* method does not address the problem of controlling a simulated human figure with various topological structures to perform a given task.

In this paper we are interested in the dynamic motion control of virtual human figures with minimum torque task performance. Specifically, for a given task with initial and final postures, we need to find a sequence of continuous motion trajectories that lead a given articulated figure from the initial posture to the goal posture without violating the predefined constraints. These constraints include the limits of joint angles and joint torques for the figures. The notion of minimizing energy expenditure and forces are basic hypotheses behind human movement [29, 31]. Since the energy expenditure during human movement is a complex nonlinear function of the body motion, joint torques have been shown to be a reasonable predictor of metabolic energy [26]. In astronauts' extra vehicular activity (EVA) motions, due to motion limits confined by the spacesuit, the joint torques output is restricted. A possible consequence of EVA tasks is that astronauts are likely to be injured because of excessive force. Even in the reduced gravity of the lunar surface, some loads could exceed human capability, and result in injuries while exerting large forces such as to dislodge an object from the surface [9]. A minimum torque-based motion planning will enable a simulation to count in the environment dynamics as well as joint angle limits and joint torque limits.

In this paper, an efficient and robust human motion control approach based on spatial notation [13] for virtual human (task simulation and evaluation) trajectory generation is introduced. It allows us to efficiently handle constrained dynamics with large numbers of degrees of freedom, various articulated figure topological structures, and constraints such as joint position limits and torque limits.

The paper is organized in the following way. In Section 2, we give an outline of our approach. In Section 3, we present the way we convert the geometry of an articulated body to a dynamic tree structured model, which is suitable for dynamic simulation and motion planning. In Section 4, we introduce the kinematics and dynamics of the dynamic tree structured model. In Section 5, we illustrate how to solve the optimal control of motion based on the minimum torque criteria. In Section 6, we give the pseudo code for our motion planning problem. Finally, in Section 7 we present our experiments.

2. Approach Outline

Our approach was motivated by the desire to enable efficient optimal control of human motions based on minimum torque criteria. The approach uses the spatial notation [13] of motion and force, and exponential expression Lie algebra for matrices and vectors. Therefore, an efficient recursive computation can be implemented by one vector equation. In the following, we present the details on kinematics, dynamics and optimal control calculations.

The entire computation process breaks down to three stages:

1. *Model simplification of an articulated body.* Create an abstract character model containing the minimal number of degrees of freedom necessary to capture the essence of the specified task, which is given as initial and final postures.
2. *Kinematics and dynamics of an articulated model.* Find the kinematic and dynamic formulations for the rigid multi-body dynamic system.
3. *Optimal control of motion based on minimum torque criteria.* Compute the motion that completes the specified task, meets all the constraints specified by the problem and satisfies the minimum torque criteria.

The model simplification stage requires a significant amount of human intervention, and the optimal control stage takes most of the computation time.

3. Model Simplification of an Articulated Body

We use *Jack*, the human modeling and simulation package developed at the University of Pennsylvania [5],* to animate the optimized motion computed by our approach. The model simplification of the articulated body is based on human models used in *Jack* (Table I).

According to different tasks, the articulated body can be simplified to different kinematic and dynamic structures (Figure 1), i.e. the single open chain (or serial chain), the simple open chain (or tree-structured), and the single/simple closed chain [17]. The maximum degrees of freedom of a multi-body dynamic system we considered for the purposes of this paper are from 6 to 12 degrees of freedom.

3.1. SINGLE OPEN CHAIN-STRUCTURED HUMAN FIGURE

When the human body moves symmetrically and freely at one end (for example, weight lifting as shown in Figure 1a), it can be modeled as a single open chain with revolute joints. The fixed link (say the feet in the weight lifting example) forms the root (*Link 0*). The numbering scheme for the single open chain is that link $(i - 1)$ is connected to link (i) by joint (i) . The axis of joint (i) is the spatial vector \hat{S}_i , and its joint variable, velocity and acceleration are measured from link $(i - 1)$ to link (i) and represented as q_i , \dot{q}_i and \ddot{q}_i , respectively. The relative spatial velocity and acceleration of link (i) with respect to link $(i - 1)$ are then $\hat{S}_i\dot{q}_i$ and $\hat{S}_i\ddot{q}_i$ respectively, according to the spatial notation [13].

3.2. SIMPLE OPEN CHAIN-STRUCTURED HUMAN FIGURE

When the human body moves asymmetrically with one end fixed and the other end moving freely (for example moving two different objects simultaneously as shown in Figure 1b), it can be modeled as a simple open chain with revolute joints. There

* The dimension of the model and the placement of the joints are based on measurements from human subjects.

Table I. The basic parameters of *Jack* model for human.

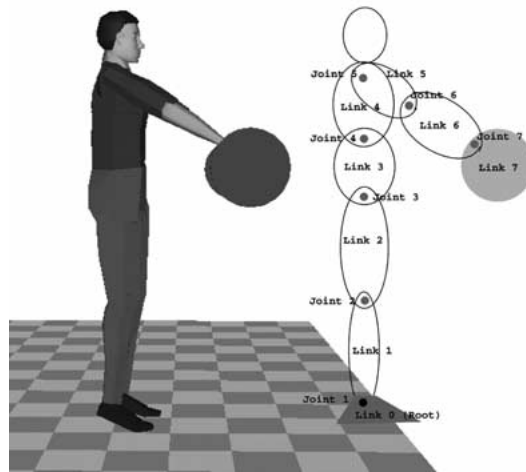
Joint	Type	DOF
wrist	Rot(y)	1
elbow	Rot(y)	1
shoulder	Rot(z) * Rot(y) * Rot(x)	3
hip	Rot(-y)	1
knee	Rot(-y)	1
ankle	Rot(z) * Rot(x) * Rot(y)	3

Link Segment	Mass (g)	Length (cm)	Radius (cm)	Width (cm)	Thickness (cm)
lower arm	1420.00	27.85	4.00		
upper arm	2480.00	30.49	6.00		
upper torso	22105.49	27.12		50.00	30.00
middle torso	8000.00	15.13		45.00	25.00
lower torso	11160.00	12.16		40.00	20.00
upper leg	8860.00	48.89	9.50		
lower leg	4080.00	47.59	7.00		
foot	1120.00				
palm	350.00				
neck	1459.63				
bottom head	1770.00				

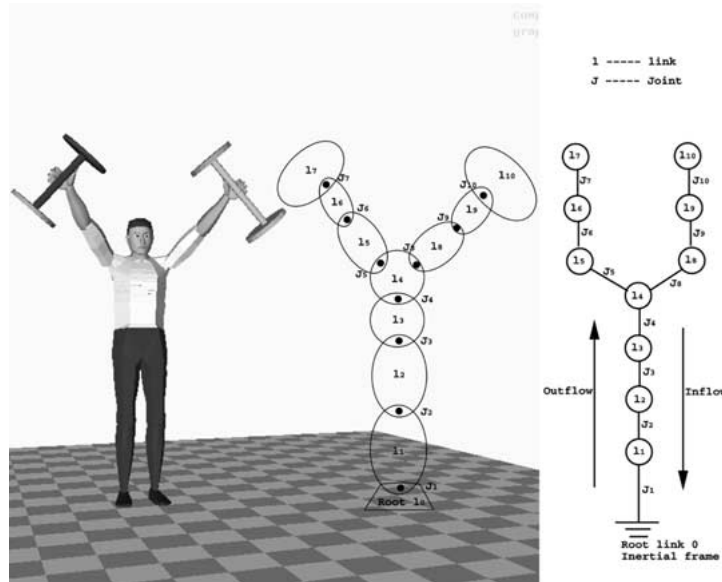
are two computation flows in a recursive algorithm [13], i.e. the outflow and the inflow. Each flow traverses all the links. The outflow recursion or iteration proceeds from the root to the leaves, and the link index increases. The inflow recursion or iteration proceeds from the leaves to root, and the link index decreases. As opposed to the single open chain model, the connectivity in the simple open chain model needs to be specified explicitly in the computation formula. Every link except the root has exactly one *parent* joint, but may have more than one *child* joints. We use index *i.child* to refer to the index number of the link outboard to link *i*, and the index *i.parent* to refer to the index number of the link inboard to link *i*. In the example shown in Figure 1b, index *4.child* for link 4, are link 5 and link 6, while index *4.parent* is link 3. Note that the index numbers for leaf link 7 (*7.child*) and link 10 (*10.child*) are null. The numbering for a tree-structure figure is not unique.

3.3. SINGLE/SIMPLE CLOSED CHAIN-STRUCTURED HUMAN FIGURE

When the human body moves symmetrically/asymmetrically with ends fixed (for example, push-ups and ladder-climbing as shown in Figures 1c and 1d), it can be



(a) Serial chain model



(b) Tree structured model

Figure 1. Kinematic models of human figure.

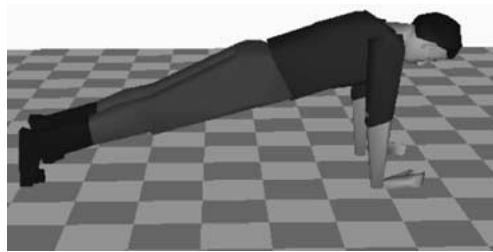
modeled as a single/simple closed chain with revolute joints, and can be treated as a single/simple open chain with constraints by cutting off the kinematic loops.

4. Kinematics and Dynamics of an Articulated Model

Here we briefly introduce the spatial algebra [13]. From the famous *Chasles theorem*, any general displacement of a rigid body can be decomposed into a rotation



(c) Closed loop structured model



(d) Pushup model

Figure 1. Continued.

about a unique axis and a translation along the axis [14]. Such a displacement can be characterized as a *screw* vector. The axis is called the *screw axis*. According to Plücker [20], a screw $\hat{\mathbf{u}}$ is composed of a pair of three-dimensional vectors: the *line vector* u and the *free vector* u_0 , which can be represented by a spatial vector. In the Euclidean space, we can describe a general rigid body motion by a six-dimensional vector over the real numbers, $\hat{\mathbf{u}}$, called spatial vector, where the first three parameters are used to describe the angular velocity, the second three parameters to describe the linear velocity with respect to some point in the rigid body. A *twist* $\hat{\mathbf{v}} = \begin{pmatrix} \omega \\ \mathbf{v}_o \end{pmatrix}$ is a motion *screw* called spatial velocity, while a *wrench* $\hat{\mathbf{f}} = \begin{pmatrix} \mathbf{f} \\ \mathbf{m}_o \end{pmatrix}$ is a force *screw* called spatial force.

Featherstone [13] presented the recursive kinematics and dynamics for the rigid multibody system using the spatial notation. We implemented the recursive kinematic and dynamic formulations in the single/simple open chain models. When the articulated figure model contains closed loops, we cut some joints within the loops so that the model becomes a single/simple open chain structure. We then enforce

the loop closure constraints in the constrained optimization formulation. For the case of pure forward dynamics simulation of such articulated figure models, a control method is also needed to force the constraints during the state variable integration, as this has been shown in the ladder climbing modeling [18].

5. Optimal Control of Motion Based on Minimum Torque Criteria

The task of motion synthesis is to find the desired motion of a human character. This ‘goal motion’ is rarely uniquely specified; rather, one looks for a motion that satisfies some set of requirements. Generally these requirements are represented either through constraints, external forces or through an objective function. For instance, we specify the initial and final postures of the human character, which we refer to as *pose constraints*. In addition, the human body has its *physical constraints* (i.e. joint limits, torque limits, penetrating constraints), the environment imposes a number of *mechanical constraints* onto the body (i.e. balancing the body in the presence of gravity), and the motion must be dynamically correct (i.e. we used dynamics constraints). When the motion is defined in this way, it straightforwardly maps onto a nonlinearly constrained optimization problem: we optimize the objective function $F(\mathbf{q}(t), t)$ parameterized in space and time, subject to all the specified constraints:

$$\text{Min } F(\mathbf{q}) \quad (1)$$

$$\text{subject to } g_i(\mathbf{q}) \leq 0, \quad i = 1, 2, \dots, l; \quad (2)$$

$$h_k(\mathbf{q}) = 0, \quad k = 1, \dots, m. \quad (3)$$

where the h_k 's are called *equality constraints*, and the g_i 's are called *inequality constraints*.

This optimization is a variational calculus problem. We use the quasi-Newton method to achieve a superlinear convergence for any initial guess. In particular, we use the BFGS [6, 11] method for its robustness.

The necessary condition for the constrained optimization problem to converge to a local minimum is to satisfy the Karush–Kuhn–Tucker (KKT) conditions (5–9). If we adjoin the constraints (2–3) by using the Lagrange multipliers λ 's a new objective function called the Lagrangian,* $L(\mathbf{q}, \lambda, \mu)$, is formed:

$$L(\mathbf{q}, \lambda, \mu) = F(\mathbf{q}) + \sum_{i=1}^l \lambda_i g_i(\mathbf{q}) + \sum_{k=1}^m \mu_k h_k(\mathbf{q}). \quad (4)$$

Then the necessary criteria for the problem (1–3) to have an optimal solution is for the following conditions to be satisfied [27]:

1. the optimum solution vector \mathbf{q}^* is realizable (5)

* This Lagrangian is not the same as in Lagrangian dynamics.

$$2. \lambda_i g_i(\mathbf{q}^*) = 0, \quad i = 1, \dots, l, \quad \lambda_i \geq 0 \quad (6)$$

$$3. \nabla F(\mathbf{q}^*) + \sum_{i=1}^l \lambda_i \nabla g_i(\mathbf{q}^*) + \sum_{k=1}^m \mu_k \nabla h_k(\mathbf{q}^*) = \mathbf{0} \quad (7)$$

$$4. \lambda_i \geq 0, \quad i = 1, \dots, l \quad (8)$$

$$5. \mu_k \text{ unrestricted in sign}, \quad k = 1, \dots, m. \quad (9)$$

With these conditions satisfied, the search line for the solution \mathbf{q}^* will be directed towards the feasible (constraints satisfied) and usable (objective function satisfied) domain of \mathbf{q} .

To solve our dynamic motion planning, we convert the constrained optimization problem to an unconstrained optimization problem using the Augmented Lagrange Multiplier method, which can then be solved using the BFGS method. The augmented Lagrangian for the constrained optimization problem can be generated as:

$$L(\mathbf{q}, \lambda, w_p) = F(\mathbf{q}) + \sum_{i=1}^l (\lambda_i \psi_i + w_p \psi_i^2) + \sum_{k=1}^m \{\lambda_k h_k(\mathbf{q}) + w_p [h_k(\mathbf{q})]^2\}, \quad (10)$$

where

$$\psi_i = \max \left[g_i(\mathbf{q}), \frac{-\lambda_i}{2w_p} \right]. \quad (11)$$

w_p is weighting factor. Note that it would need one slack variable for each inequality constraint to be converted to an equality constraint [32] in the augmented Lagrangian. The replacement of $g_i(\mathbf{q})$ with ψ_i avoids the use of the slack variable, so that the efficiency is greatly increased [23].

5.1. OBJECTIVE FUNCTION AND CONSTRAINTS

In our applications, the objective function $F(q)$ we chose is the integration of the Euclidean norm of the joint torque vector over time. We then add the following constraints: (a) the initial and final posture are specified with prescribed joint angle values; (b) the joint angles have motion range limits; and (c) each joint has its torque output limit. Therefore, the optimization problem can be rewritten as:

$$\text{Min}_{(\boldsymbol{\tau})} \quad F(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) = \frac{1}{2} \int_{t_0}^{t_f} \boldsymbol{\tau}^T \mathbf{H}^{-1} \boldsymbol{\tau} dt$$

$$\text{subject to} \quad \mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\Phi}_q^T \boldsymbol{\lambda} = \boldsymbol{\tau},$$

$$\begin{aligned}
\mathbf{q}(0) &= \mathbf{q}_0, \quad \dot{\mathbf{q}}(0) = \mathbf{0}, \\
\mathbf{q}(t_f) &= \mathbf{q}_f, \quad \dot{\mathbf{q}}(t_f) = \mathbf{0}, \\
\tau_i^{\text{low}} &\leq \tau_i \leq \tau_i^{\text{up}}, \quad i = 1, 2, \dots, l; \\
q_i^{\text{low}} &\leq q_i \leq q_i^{\text{up}}, \quad i = 1, 2, \dots, l
\end{aligned} \tag{12}$$

$$\text{loop closure: } \Phi(\mathbf{q}) = \mathbf{0}. \tag{13}$$

where $\mathbf{H}(\mathbf{q})$ in the vector dynamic equation, is the inertial matrix, the vector $\mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}})$ includes Coriolis, centrifugal and gravitational forces, λ are the Lagrange multipliers, $\Phi_q = \partial\Phi/\partial\mathbf{q}$ is the constraint Jacobian matrix. Based on (3) each inequality constraint g_i can be rewritten as: $g_i = \tau_i - \tau_i^{\text{up}}$ and $g_i = \tau_i^{\text{low}} - \tau_i$; $g_i = q_i - q_i^{\text{up}}$ and $g_i = q_i^{\text{low}} - q_i$. The equality constraints $h_k(\mathbf{q})$ are the initial and final joint angle values.

Since in our paper the cases we considered are only joint torques, the components of τ are of the same dimensions, e.g. torques. Therefore, we compute $\tau^T \tau$ instead of $\tau^T \mathbf{H}^{-1} \tau$ in our program and following sections for computation simplicity, and it should give us the same optimal result.

Notice that for an articulated figure containing a closed loop chain, the loop closure constraints are needed.

5.2. GRADIENT METHOD

To solve the nonlinear programming problem(12), we need to compute the gradient of the augmented objective function. Computing the gradient approximately by the finite difference method can save considerable amount of programming effort. However, it costs additional $(n+1)$ (n is the dimension of the problem) integration of the system equations in every iteration. As a result, the dominant portion of the computational time is spent on the gradient evaluations. Also, the solution may not converge because of the approximation. To improve the efficiency, we introduce an analytical gradient method.

The gradient of the objective function is

$$\nabla F(\tau) = \int_{t_0}^{t_f} (\nabla \tau \cdot \tau) dt \tag{14}$$

which can be evaluated once we know how to compute the derivative of a spatial vector/matrix with respect to a joint variable. In order to compute the joint torque τ and its gradient $\nabla \tau$, we need to compute the joint spatial velocity \hat{v} , the spatial acceleration \hat{a} , and their gradients, respectively. And fortunately, using the spatial notation, we can derive the analytical gradient function in a recursive way. By using Lie group and Lie algebra mathematics concepts, we derive the gradient function of spatial quantities as presented in the rest of this section.

A general spatial motion which includes translation and rotation can be described by a 4×4 homogeneous transformation. This homogeneous transformation group is called the $SE(3)$ group, and is defined as:

$$SE(3) \equiv \left\{ \left(\begin{array}{cc} \mathbf{R} & \mathbf{d} \\ 0 & 1 \end{array} \right)_{4 \times 4} \mid (\mathbf{R}\mathbf{R}^T = I) \right\}, \quad (15)$$

where \mathbf{R} is a 3×3 rotation transformation matrix between two coordinate frames, and \mathbf{d} is the translation vector between the two frame's origins.

A general spatial vector can be expressed as

$$\hat{\mathbf{v}} = \begin{pmatrix} \boldsymbol{\omega} \\ \mathbf{v} \end{pmatrix}, \quad \text{where } \boldsymbol{\omega} = \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} \quad \text{and } \mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}.$$

We define \mathbf{S} to be the space of all spatial vectors, such that:

$$\begin{aligned} \mathbf{S} &\equiv \left\{ \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ v_1 \\ v_2 \\ v_3 \end{pmatrix} \mid (\forall w_1, w_2, w_3, v_1, v_2, v_3 \in \mathfrak{R}) \right\} \\ &\cong \left\{ \begin{pmatrix} 0 & -w_3 & w_2 & v_1 \\ w_3 & 0 & -w_1 & v_2 \\ -w_2 & w_1 & 0 & v_3 \\ 0 & 0 & 0 & 0 \end{pmatrix} \right\}. \end{aligned} \quad (16)$$

Therefore, \mathbf{S} is isomorphic to the Lie algebra of the homogeneous transformation group $SE(3)$ (15).

A general transformation X_i^{i-1} from coordinate i to $(i-1)$ as defined in [13] can be a composition of a rotation q_i about the screw axis $\hat{\mathbf{S}}_i$ followed by a translation \mathbf{D}_i . Then,

$$X_i^{i-1} \equiv \mathbf{D}_i e^{\hat{\mathbf{S}}_i q_i} \in SE(3). \quad (17)$$

where

$$\mathbf{D}_i = \begin{pmatrix} \mathbf{I} & \mathbf{d}_i \\ 0 & 1 \end{pmatrix}_{4 \times 4} \quad \text{and } \hat{\mathbf{S}}_i \in \mathbf{S}.$$

Suppose $\hat{\mathbf{v}}$ is a *spatial velocity* expressed in coordinate frame $(i-1)$. Using the fact that \mathbf{S} is isomorphic to \mathfrak{R}^6 , then the rigid body coordinate transformation as defined in [13] is actually the adjoint mapping, which is equivalent to an orthogonal transformation of $\hat{\mathbf{v}}$

$$\hat{\mathbf{X}}_{i-1}^i \hat{\mathbf{v}} \equiv \text{Ad}_{(X_i^{i-1})^{-1}} \hat{\mathbf{v}} = (X_i^{i-1})^{-1} \hat{\mathbf{v}} X_i^{i-1}, \quad (18)$$

where $\text{Ad}_{(X_i^{i-1})^{-1}}$ is called the adjoint mapping.

The linear mapping $\hat{\mathbf{X}}_{i-1}^i$ (a 6×6 matrix) can be expressed as

$$\hat{\mathbf{X}}_{i-1}^i \equiv \text{Ad}_{(X_i^{i-1})^{-1}} = \begin{bmatrix} \mathbf{R}_i^T & 0 \\ \tilde{\mathbf{D}}_i \mathbf{R}_i^T & \mathbf{R}_i^T \end{bmatrix}, \quad (19)$$

where

$$\tilde{\mathbf{D}}_i \equiv \text{skew}(\mathbf{d}_i) = \begin{pmatrix} 0 & -d_3 & d_2 \\ d_3 & 0 & -d_1 \\ -d_2 & d_1 & 0 \end{pmatrix}. \quad (20)$$

Notice that $\mathbf{R}_i \cong e^{-\hat{\mathbf{S}}_i q_i}$.

Similarly, suppose that $\hat{\mathbf{f}}$ is a *wrench* in coordinate frame $(i + 1)$. We then have the following expression

$$\hat{\mathbf{X}}_{i+1}^i \hat{\mathbf{f}} \equiv \text{Ad}_{X_{i+1}}^* \hat{\mathbf{f}} = X_{i+1}^i \hat{\mathbf{f}} (X_{i+1}^i)^{-1}, \quad (21)$$

where the linear mapping $\hat{\mathbf{X}}_{i+1}^i$ can be expressed as the following 6×6 matrix,

$$\hat{\mathbf{X}}_{i+1}^i \equiv \text{Ad}_{X_{i+1}}^* = \begin{bmatrix} \mathbf{R}_{i+1} & 0 \\ \mathbf{R}_{i+1} \tilde{\mathbf{D}}_{i+1}^T & \mathbf{R}_{i+1} \end{bmatrix}. \quad (22)$$

The *spatial cross operator* defined in [13] is actually a *Lie bracket*, a measure of the non-commutativity of two spatial vectors, i.e.,

$$\hat{\mathbf{a}} \hat{\times} \hat{\mathbf{b}} \equiv [\hat{\mathbf{a}}, \hat{\mathbf{b}}] = \hat{\mathbf{a}} \hat{\mathbf{b}} - \hat{\mathbf{b}} \hat{\mathbf{a}} = ad_{\hat{\mathbf{a}}}(\hat{\mathbf{b}}). \quad (23)$$

With the above definitions and expressions (16–23), we can derive the gradient function for the spatial vector computation in a recursive form. For example, we take the derivative of the recursive formulation of $\hat{\mathbf{v}}_i$ with respect to q :

$$\begin{aligned} \frac{\partial \hat{\mathbf{v}}_i}{\partial q} &= \frac{\partial}{\partial q} (\hat{\mathbf{X}}_{i-1}^i \hat{\mathbf{v}}_{i-1}) + \hat{\mathbf{S}}_i \frac{\partial \dot{q}_i}{\partial q} \quad \left(\frac{\partial \hat{\mathbf{v}}_0}{\partial q} = \hat{\mathbf{0}} \right) \\ &= \frac{\partial}{\partial q} ((X_i^{i-1})^{-1} \hat{\mathbf{v}}_{i-1} X_i^{i-1}) + \hat{\mathbf{S}}_i \frac{\partial \dot{q}_i}{\partial q} \\ &= \frac{\partial (X_i^{i-1})^{-1}}{\partial q} \hat{\mathbf{v}}_{i-1} X_i^{i-1} + (X_i^{i-1})^{-1} \frac{\partial \hat{\mathbf{v}}_{i-1}}{\partial q} X_i^{i-1} \\ &\quad + (X_i^{i-1})^{-1} \hat{\mathbf{v}}_{i-1} \frac{\partial X_i^{i-1}}{\partial q} + \hat{\mathbf{S}}_i \frac{\partial \dot{q}_i}{\partial q} \\ &= -\hat{\mathbf{S}}_i \frac{\partial q_i}{\partial q} e^{-\hat{\mathbf{S}}_i q_i} \mathbf{D}_i^{-1} \hat{\mathbf{v}}_{i-1} \mathbf{D}_i e^{\hat{\mathbf{S}}_i q_i} + \hat{\mathbf{X}}_{i-1}^i \frac{\partial \hat{\mathbf{v}}_{i-1}}{\partial q} \\ &\quad + e^{-\hat{\mathbf{S}}_i q_i} \mathbf{D}_i^{-1} \hat{\mathbf{v}}_{i-1} \mathbf{D}_i e^{\hat{\mathbf{S}}_i q_i} \hat{\mathbf{S}}_i \frac{\partial q_i}{\partial q} + \hat{\mathbf{S}}_i \frac{\partial \dot{q}_i}{\partial q} \end{aligned}$$

$$\begin{aligned}
&= -\hat{\mathbf{S}}_i \frac{\partial q_i}{\partial q} (\hat{\mathbf{X}}_{i-1}^i \hat{\mathbf{v}}_{i-1}) + \hat{\mathbf{X}}_{i-1}^i \frac{\partial \hat{\mathbf{v}}_{i-1}}{\partial q} + (\hat{\mathbf{X}}_{i-1}^i \hat{\mathbf{v}}_{i-1}) \hat{\mathbf{S}}_i \frac{\partial q_i}{\partial q} + \hat{\mathbf{S}}_i \frac{\partial \dot{q}_i}{\partial q} \\
&= \hat{\mathbf{X}}_{i-1}^i \frac{\partial \hat{\mathbf{v}}_{i-1}}{\partial q} + (\hat{\mathbf{X}}_{i-1}^i \hat{\mathbf{v}}_{i-1}) \hat{\times} (\hat{\mathbf{S}}_i \frac{\partial q_i}{\partial q}) + \hat{\mathbf{S}}_i \frac{\partial \dot{q}_i}{\partial q}. \tag{24}
\end{aligned}$$

Similarly, we can compute the derivative of the recursive formulation of $\hat{\mathbf{a}}_i$ with respect to q . The pseudo code for the recursive computation of the gradient with respect to a parameter q for our problem is as follows:

Given: $q_i, \dot{q}_i, \ddot{q}_i, \hat{\mathbf{M}}_i', \hat{\mathbf{f}}_i^{\text{ext}}$ for all $i \in (1, \dots, n)$;

$$\frac{\partial \hat{\mathbf{v}}_0}{\partial q} = \hat{\mathbf{0}}; \quad \frac{\partial \hat{\mathbf{A}}_0}{\partial q} = \hat{\mathbf{0}}; \quad \frac{\partial \hat{\mathbf{f}}_n}{\partial q} = \frac{\partial \hat{\mathbf{f}}_n^{\text{net}}}{\partial q};$$

Forward path:

for $i = 1$ to n do:

$$\begin{aligned}
\frac{\partial \hat{\mathbf{v}}_i}{\partial q} &= \hat{\mathbf{X}}_{i-1}^i \frac{\partial \hat{\mathbf{v}}_{i-1}}{\partial q} + (\hat{\mathbf{X}}_{i-1}^i \hat{\mathbf{v}}_{i-1}) \hat{\times} \left(\hat{\mathbf{S}}_i \frac{\partial q_i}{\partial q} \right) + \hat{\mathbf{S}}_i \frac{\partial \dot{q}_i}{\partial q}; \\
\frac{\partial \hat{\mathbf{A}}_i}{\partial q} &= \hat{\mathbf{X}}_{i-1}^i \frac{\partial \hat{\mathbf{A}}_{i-1}}{\partial q} + (\hat{\mathbf{X}}_{i-1}^i \hat{\mathbf{A}}_{i-1}) \hat{\times} \left(\hat{\mathbf{S}}_i \frac{\partial q_i}{\partial q} \right) \\
&\quad + \frac{\partial \hat{\mathbf{v}}_i}{\partial q} \hat{\times} \hat{\mathbf{S}}_i \dot{q}_i + \hat{\mathbf{v}}_i \hat{\times} \hat{\mathbf{S}}_i \frac{\partial \dot{q}_i}{\partial q} + \hat{\mathbf{S}}_i \frac{\partial \ddot{q}_i}{\partial q};
\end{aligned}$$

end for.

Backward path:

for $i = n$ to 1 do:

$$\begin{aligned}
\frac{\partial \hat{\mathbf{f}}_i}{\partial q} &= \hat{\mathbf{X}}_{i+1}^i \left(\frac{\partial \hat{\mathbf{f}}_{i+1}}{\partial q} + \hat{\mathbf{S}}_{i+1} \frac{\partial q_{i+1}}{\partial q} \hat{\times} \hat{\mathbf{f}}_{i+1} \right) \\
&\quad + \hat{\mathbf{M}}_i \frac{\partial \hat{\mathbf{A}}_i}{\partial q} + \hat{\mathbf{v}}_i \hat{\times} \left(\hat{\mathbf{M}}_i \frac{\partial \hat{\mathbf{v}}_i}{\partial q} \right) + \frac{\partial \hat{\mathbf{v}}_i}{\partial q} \hat{\times} \hat{\mathbf{M}}_i \hat{\mathbf{v}}_i - \frac{\partial \hat{\mathbf{f}}_i^{\text{ext}}}{\partial q}; \\
\frac{\partial \tau_i}{\partial q} &= \hat{\mathbf{S}}_i^y \frac{\partial \hat{\mathbf{f}}_i}{\partial q}
\end{aligned}$$

end for.

where $q_i, \dot{q}_i, \ddot{q}_i$ are the angular displacement, the velocity and the acceleration for link i , respectively. $\hat{\mathbf{M}}_i'$ is the spatial inertia for link i expressed in local coordinate frame i . $\hat{\mathbf{f}}_i^{\text{ext}}$ is the external spatial force (called *wrench*). $\hat{\mathbf{f}}_i^{\text{net}}$ is the total body *wrench* acting on link i . $\hat{\mathbf{v}}_i$ is the spatial velocity for link i . $\hat{\mathbf{A}}_i$ is spatial acceleration for

link i . $\hat{\mathbf{X}}_{i-1}^i$ is the *spatial transformation* defined in [13]. The superscript s on the spatial axis vector $\hat{\mathbf{S}}$ is an operator to reverse the positions of the pair vectors within the spatial vectors, so that the product of a spatial velocity and a spatial force is a power scalar [13].

We can see that the gradient function computation shares the same computation with the dynamic formulation. The computational flows for $\nabla\tau$ and for τ are very similar, sharing many common expressions.

The gradient of the augmented objective function is computed as follows (refer to Equation (10)):

$$\nabla L(\mathbf{q}, \lambda, w_p) = \nabla F(\tau(\mathbf{q})) + \nabla G(\mathbf{q}) + \nabla H(\mathbf{q}), \quad (25)$$

where the gradient of the minimum torque objective function is computed as

$$\nabla F[\tau(\mathbf{q})] = \nabla \left(\sum_{i=1}^{\text{dof}} \tau_i^2 \right) = 2 \sum_{i=1}^{\text{dof}} \tau_i \nabla \tau_i. \quad (26)$$

Referring to (10), the gradient vector of the inequality constraints is:

$$\nabla G(\mathbf{q}) = \sum_{i=1}^l \nabla (\lambda_i \psi_i(\mathbf{q}) + w_p \psi_i(\mathbf{q})^2) = \sum_{i=1}^l [\lambda_i + 2w_p \psi_i(\mathbf{q})] \nabla \psi_i(\mathbf{q}), \quad (27)$$

where

$$\psi_i(\mathbf{q}) = \max \left[g_i(\mathbf{q}), \frac{-\lambda_i}{2w_p} \right]. \quad (28)$$

Also referring to (10), the gradient vector of the equality constraints is:

$$\begin{aligned} \nabla H(\mathbf{q}) &= \sum_{k=l+1}^m \nabla (\mu_k h_k(\mathbf{q}) + w_p [h_k(\mathbf{q})]^2) \\ &= \sum_{k=l+1}^m [\mu_k + w_p h_k(\mathbf{q})] \nabla h_k(\mathbf{q}). \end{aligned} \quad (29)$$

5.3. DISCRETIZATION USING BSPLINES

The solution space $(\mathbf{q}(t), \dot{\mathbf{q}}(t), \ddot{\mathbf{q}}(t))$ of (12) and (13) is infinitely large. We discretize the time scale (t) of the trajectory, so that an infinite dimensional problem can be converted into a finite dimensional problem by reducing the function space of possible trajectories to a finite dimensional space. The nodal value \mathbf{q}^j , which specifies the control point j of the BSpline function on the time scale, can be used to describe the displacements, velocities and accelerations at selected control points on the time scale. Among these selected control points, the displacement field $\mathbf{q}(t)$

is approximated using a finite number of interpolating polynomials called BSpline *shape functions* [12].

$$q_i(t) = \sum_{j=1}^N B_j(t) \tilde{q}_i^j, \quad i = 1, \dots, m, \quad (30)$$

where $B_j(t)$ are the BSpline shape functions at time j , q_i^j is the control point value at time j for variable i (note: it is a constant with respect to time), m is the total number of degrees of freedom of the articulated figure, and N is the number of control points on the time axis.

The derivatives of Equation (30) are simply:

$$\dot{q}_i(t) = \sum_{j=1}^N \dot{B}_j(t) \tilde{q}_i^j, \quad i = 1, \dots, m, \quad (31)$$

$$\ddot{q}_i(t) = \sum_{j=1}^N \ddot{B}_j(t) \tilde{q}_i^j, \quad i = 1, \dots, m. \quad (32)$$

These equations can be substituted into Equation (12). Thus, the problem of finding the unknown trajectory function becomes the problem of finding the q_i^j 's, a finite dimensional nonlinear constrained optimization problem.

Using (30), for any given function $f \equiv f(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$, its gradient can be computed through the chain rule:

$$\begin{aligned} \frac{\partial f}{\partial \tilde{q}_i^j} &= \frac{\partial f}{\partial q_i} \frac{\partial q_i}{\partial \tilde{q}_i^j} + \frac{\partial f}{\partial \dot{q}_i} \frac{\partial \dot{q}_i}{\partial \tilde{q}_i^j} + \frac{\partial f}{\partial \ddot{q}_i} \frac{\partial \ddot{q}_i}{\partial \tilde{q}_i^j} \\ &= \frac{\partial f}{\partial q_i} B_j + \frac{\partial f}{\partial \dot{q}_i} \dot{B}_j + \frac{\partial f}{\partial \ddot{q}_i} \ddot{B}_j, \quad i = 1, \dots, m; j = 1, \dots, N. \end{aligned} \quad (33)$$

To save unnecessary computations, we only need to count in those items where the B_j 's are non-zero. Also, the shape functions and their derivatives can be stored once the BSpline function is constructed for later use. The trajectory generated this way is smooth. The precision using the BSpline function can be enhanced by increasing the number of control points.

5.3.1. Reformulation of the Minimum Torque Control Motion Based on the Use of BSplines

With the discretization method, we substitute Equations (30–32) into the minimum torque motion formulation shown in (12–13). The problem of searching $\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$, $\ddot{\mathbf{q}}(t)$ becomes the problem of searching $\tilde{\mathbf{q}}_i^j$. Therefore, the search space becomes finite. The selected nodal control points form a vector of parameters to be optimized. Therefore, the optimization problem (12) can be reformulated using (30) as

follows.

$$\begin{aligned} \text{Min}_{(\tilde{q}_i^j)} \quad & F(\tilde{q}_i^j) = \frac{1}{2} \int_{t_0}^{t_f} \tau^2(\tilde{q}_i^j) dt \\ \text{subject to} \quad & \mathbf{H}(\tilde{q}_i^j) + \mathbf{Q}(\tilde{q}_i^j) + \mathbf{\Phi}_q^T(\tilde{q}_i^j)\boldsymbol{\lambda} = \boldsymbol{\tau}(\tilde{q}_i^j), \\ & q_i(0) - \sum_{j=1}^N B_j(t_0)\tilde{q}_i^j = 0 \\ & q_i(f) - \sum_{j=1}^N B_j(t_f)\tilde{q}_i^j = 0 \\ & [\text{for closed loop, } \mathbf{\Phi}(\tilde{q}_i^j) = \mathbf{0}] \\ & \tau_i^{\text{low}} \leq \tau_i(\tilde{q}_i^j) \leq \tau_i^{\text{up}}, \\ & q_i^{\text{low}} \leq q_i(\tilde{q}_i^j) \leq q_i^{\text{up}}, \quad i = 1, 2, \dots, m, \end{aligned}$$

where m is the number of degrees of freedom of the articulated figure and N is the number of control points on the time axis.

6. The Pseudo Code For Motion Planning

We now present the pseudo code for computing the motion planning problem.

1. *Given problem*: Initial and final values of the angular displacement for all the figure joints (assume zero initial and ending velocities), and the motion duration time.
2. *Initial guess*: The initial trajectory (linearly varying between starting and ending values; we can also use other experimental data).
3. *Compute BSplines*: Using BSpline functions, which include information of the shape functions and the control point values, to interpolate the initial trajectory.
4. *Solving the problem*: Solve for angular displacements, velocities and accelerations. $(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ for all the joints in the figure. The algorithm has the following steps:
 - (a) Kinematics calculation: $\hat{\mathbf{v}}_i, \hat{\mathbf{A}}_i$ (twists).
 - (b) Dynamics calculation: $\hat{\mathbf{f}}_i, \tau_i$ (wrenches and torques).
 - (c) Constraints evaluation: equalities $h_i(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \tau)$ and inequalities $g_i(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \tau)$. The constraints include: joint angle limits; joint torque limits; postural constraints; closed loop structure constraints; other constraints.

- (d) Assembling the augmented Lagrangian: the constrained optimization problem is converted to an unconstrained one.
- (e) Gradient calculation: the gradients of the augmented Lagrangian.
- (f) Evaluation: meet the convergence criteria? If yes, output the results. If no, use line search (BFGS) to find new control point values.

7. Results

We experimented with the use of the dynamic optimal motion planning techniques for human motion studies. These include the modeling of multibody dynamic systems with large degrees of freedom and various kinds of topological structures including open and closed loop chains. We first validate the method by comparing the result of our minimum torque optimized motion of a compound pendulum with its equivalent forward dynamics simulation during a free swing. The two results agree very well. We further implement the approach to simulate human arm and leg lifting, weight lifting, chin-ups, dip-downs: on Earth, on Mars and in space. We also apply the approach to simulate push-ups, a closed loop case.

This numerical optimization method requires an initial guess. Since this is a local optimization method, the initial guess greatly affects the optimal solution. The simplest initial guess for a motion is a kinematic smooth motion, which linearly interpolates the motion between the starting and the ending postures. Nevertheless, the initial guess affects the speed of convergence. Therefore, a near dynamically correct trajectory guess could yield a faster and for globally optimal solution if such a trajectory can be computed. Based on our efficient formulation, all the experiments run in close to interactive rates, less than 120 minutes on a SGI R4400 (CPU) machine. In our experiments, we use the linear interpolation trajectories in the joint space for all the joints as the initial guess of the motion trajectory, due to the simplicity. Figures 2c, 3a and 3b show the initial guesses in spline function representation.

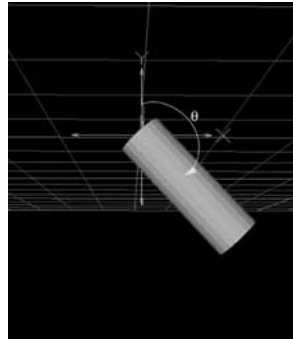
7.1. A VALIDATION STUDY: THE CASE OF A COMPOUND PENDULUM

We start with the validation of the minimum torque motion planning approach for a single compound pendulum.

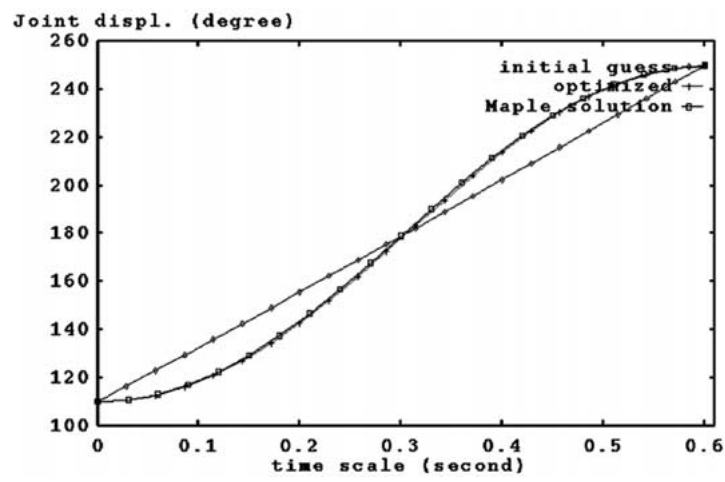
We first do the forward dynamic simulation of a cylindrical pendulum with a mass equal to 8 kg, a diameter of 15 cm and a length of 44 cm (Figure 2a). The differential equation of motion for the system is

$$I_0 \ddot{\theta} - mgl \sin(\theta) = 0, \quad (34)$$

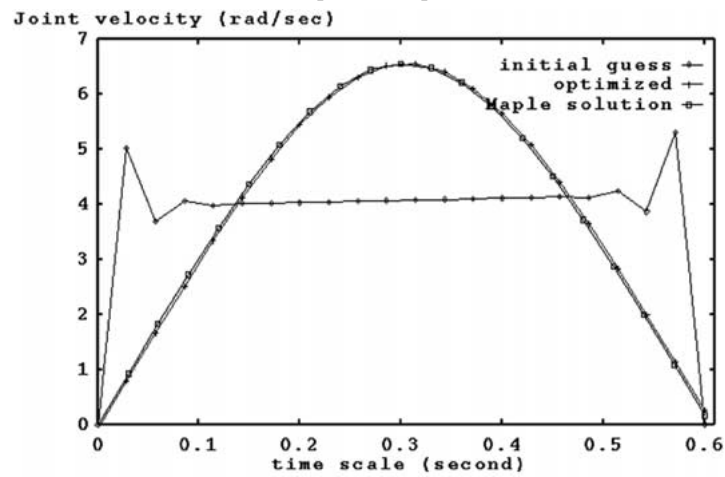
where I_0 is the moment of inertia of the cylinder about the swinging point, m is the mass, l is the length of the cylinder, and g is the gravity constant. Notice that there is no external torque other than that induced by gravity. Since the starting swing angle



(a) Model

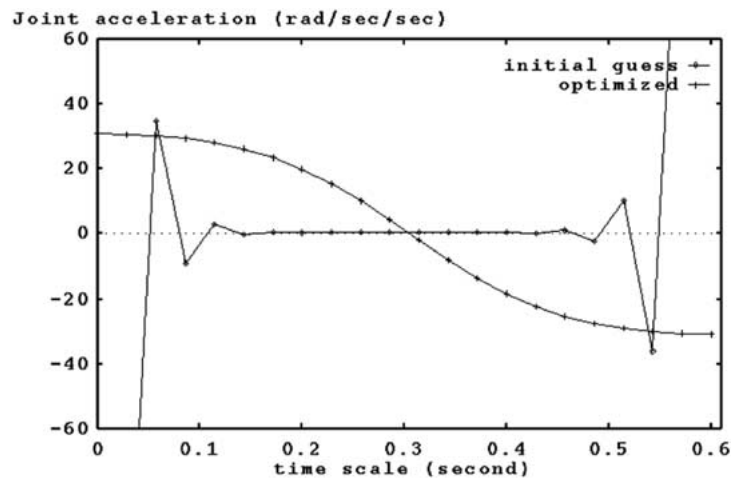


(b) Displacement profile

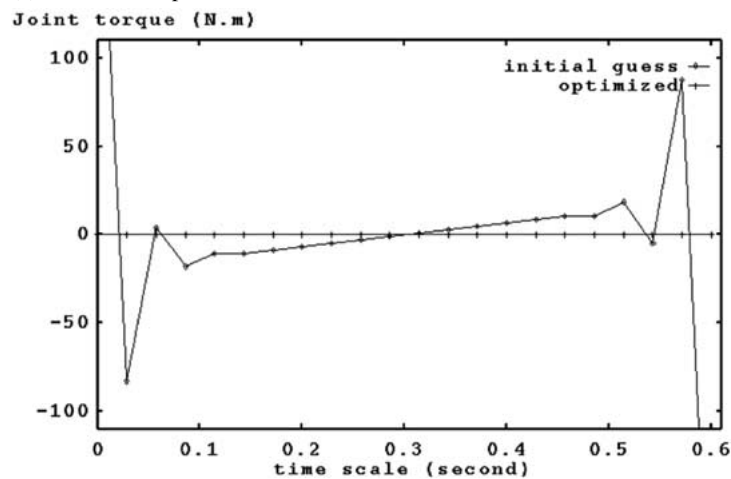


(c) Velocity profile

Figure 2. Minimum torque motion of a compound single pendulum, compared with the *Maple*TM forward dynamics simulation of a torque-free motion for the same pendulum.



(a) Acceleration profile



(b) Torque profile

Figure 3. Minimum torque motion of a compound single pendulum.

θ can have any value other than a small one, this is not a simple harmonic vibration. Therefore, we cannot find a closed form solution. We use *Maple*TM to solve the differential equation (34). The initial position of the pendulum is 110 degrees from the vertical y axis. The pendulum executes a harmonic motion swing about the vertical (y) axis. The trajectory plots of the angular displacement and the velocity are shown in Figures 2b and 2c. From the plot in Figure 2c, we can measure the approximate half period of the motion, which is about 0.6 seconds.

Taking the half period obtained from the above forward dynamics simulation as the duration of the motion, and the starting position of 110 degrees and ending position of 250 degrees from the *Maple* solution in Figure 2b, we can execute

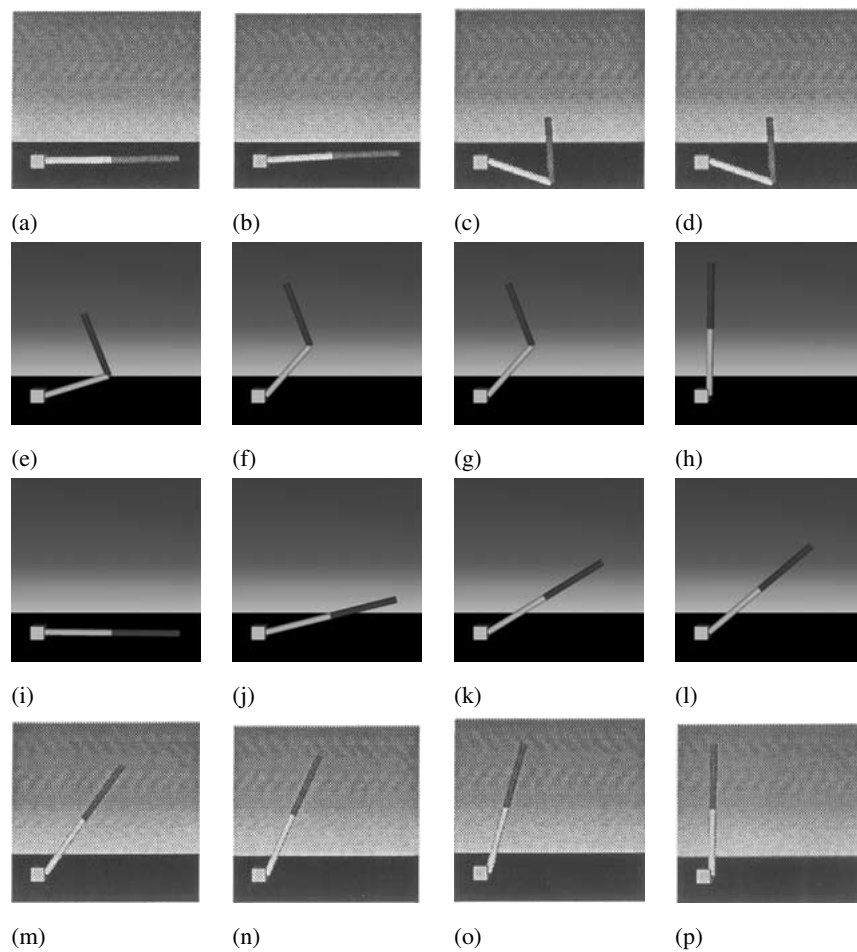


Figure 4. (a-h) Animation sequence from one arm weight lifting with minimum joint torque. (i-p) Animation sequence from one arm motion with kinematic interpolation.

our program to do the motion planning using our motion planning approach. The results are shown in Figures 2 and 3). Our approach finds a minimum torque motion for the cylindrical pendulum under no other external joint torque than that induced by gravity. The minimum torque trajectory found is zero, as shown in Figure 3b. Therefore, the planned motion using our approach does reproduce the forward dynamics simulation of the free swing of the pendulum. The displacement and velocity trajectories of the optimal solutions agree very well with the results from the above forward dynamic simulation given by *Maple*, except at the two end points (Figure 2). The large variations at the two end points of the initial guess in the velocity profile plot are due to the spline function interpolation, which is not second order continuous at those end points.

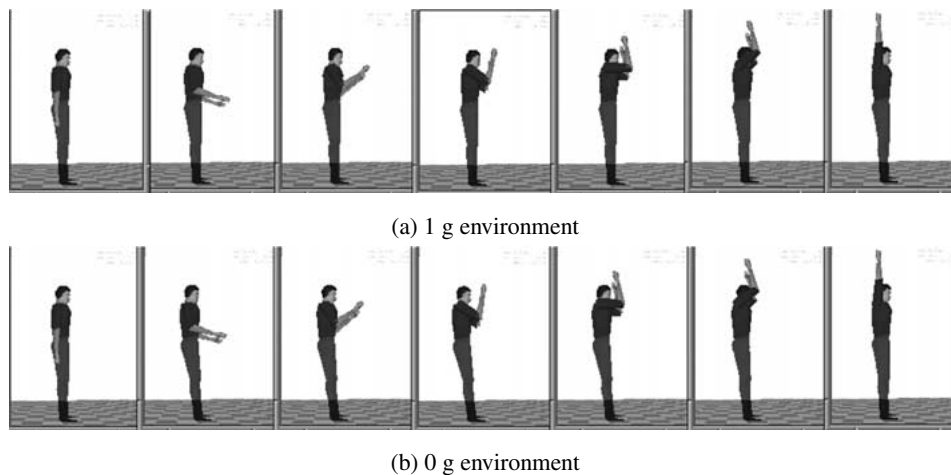


Figure 5. Arm lifting by adding in resistant joint torques.

7.2. OPEN CHAIN MODELS

7.2.1. Double Pendulum

This is the second model to compare the result of our minimum torque motion planning method with the result based on a pure kinematic motion interpolation between a starting and an ending position. The result of minimum torque motion is in harmony with the common sense (may also be mathematically confirmed) expectation that the combined moment of inertia at the base joint must be minimal. The simulation shows a folding of the double pendulum while it is in motion (Figures 4a–4h). This is in contrast to Figures 4i–4p, where the sequence of the displacement angles is kinematically interpolated evenly between the starting and ending positions. As it can be seen, our dynamic minimum torque-based solution provides the expected answer, which is physically plausible. This is in contrast to the kinematically interpolated motion. Each segment of the pendulum in the figures has a cylindrical shape with length 1 m, diameter 10 cm, and mass 7.7 kg.

7.2.2. Arm Lifting

In the following we use *Jack* to animate an example of arm lifting motion using our motion planning approach. Here, we use a parabolic curve as an initial guess of the trajectory by guessing a midpoint position. According to our experience, the speed of the solution depends on how close the initial guess is to the solution. Therefore, when an approximate curved trajectory can be perceived, using a parabolic trajectory guess yields a better optimized solution than using a straight line trajectory interpolation.

In the first experiment, we compute the motion planning for the human arm lifting on Earth. In the next experiment, we solve the motion planning problem

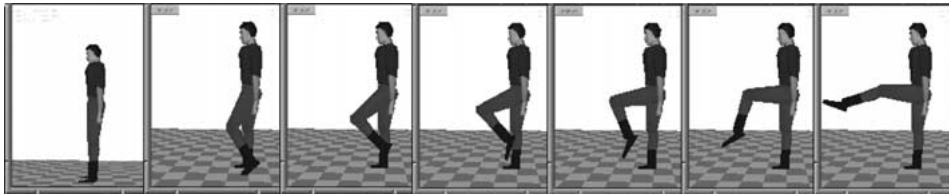


Figure 6. Leg lifting counting in resistant joint torques.

for an astronaut lifting his arms in space (zero gravity). Figures 5a and 5b show the sequence of the motions on Earth and in space, respectively. By comparing both simulations and results, we see that it takes less effort to do the same motion in space than on Earth. Furthermore, we find that in space the astronaut's body leans back when he lifts his arm, which is consistent with the conservative of the momentum law. We notice that based on the computation that the ankle joint torque is close to zero. But if the person leans back and is on Earth, the ankle joint must undertake a tremendous torque. We also simulated the arm lifting motion on Mars. Resistant torques in human joints are considered by adding spring and damping torsional forces during the simulation calculations.

7.2.3. Leg Lifting

Figure 6 is a motion sequence for leg lifting on Earth by counting in the resistant torque. The maximum torque output is at the hip, reading about 30 N.m. It peaks when the hip joint reaches an angle of about 30 degrees from the vertical line. (Note: The motion sequence for leg lifting on Earth is similar to the ones on Mars and in space, but has larger joint torques.)

7.2.4. Weight Lifting

Weight lifting is a common important task. We apply our motion planning method to such a non-trivial demonstration. The human model is 1.83 m high, and weights 81 kg. The weight to be lifted is a 25 kg barbell. Figures 7a, 7b and 7c are the sequences of the simulated motions on Earth, on Mars and in space, respectively. We also notice that in space the astronaut can lean back to minimize his joint torques when lifting the weights, while on Earth he must keep his balance.

7.2.5. Chinup

Figure 8 shows a chin-up motion sequence on Earth (1 g) and in space (0 g). On Earth, the minimum torque optimized motion accumulates potential energy obtained from the swing of the body and then releases the energy towards the end posture. Since we learned in practice that we can do chinups more easily by swinging our bodies than without swinging, our simulation looks more natural. In space, we do not have to swing our bodies to overcome the gravity, but we still need the momentum to overcome the inertia.

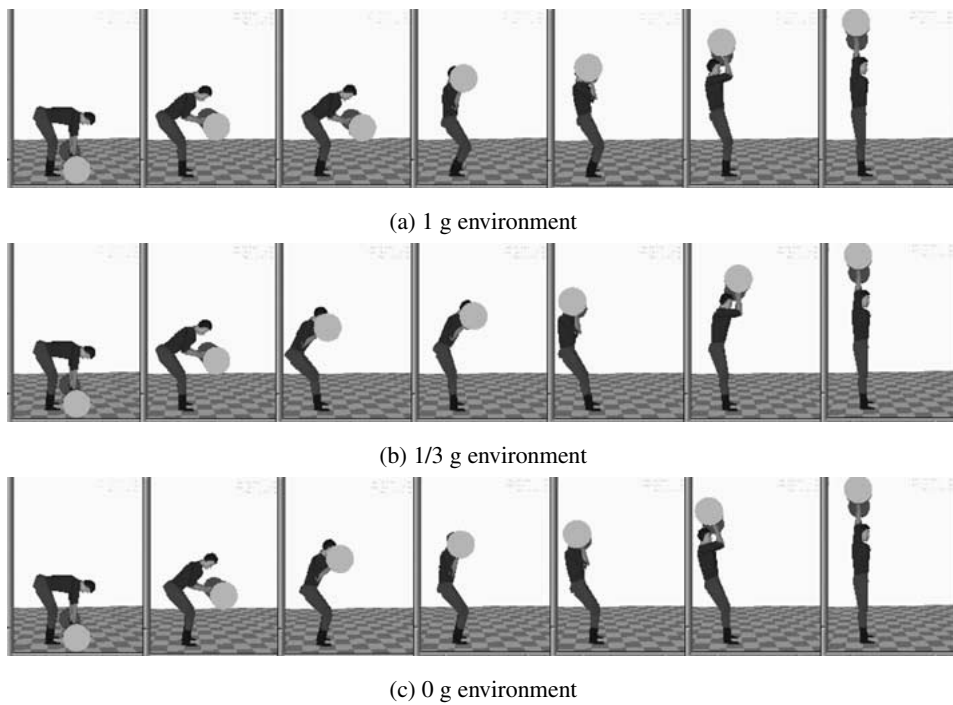


Figure 7. Weight lifting counting in resistant joint torques.

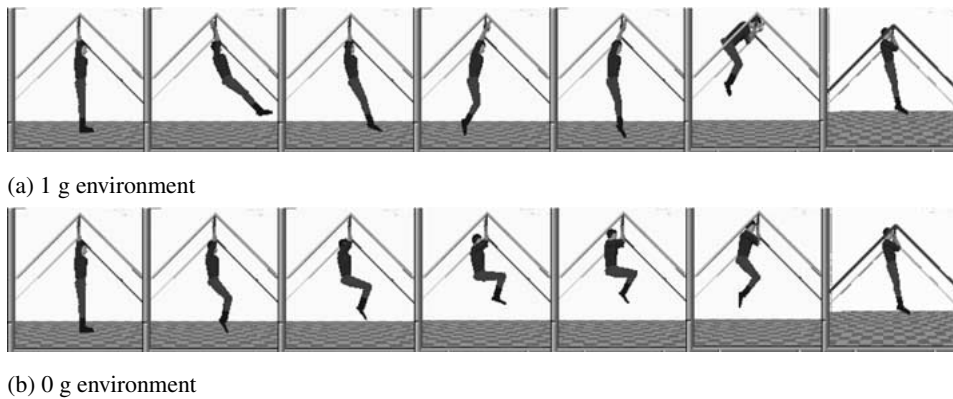


Figure 8. Chinup motion by adding in resistant joint torques.

7.2.6. Dipdown

Figure 9 shows a motion sequence for dipdown in space (0 g). Aslo from the simulation we can see that the figure does a swing motion to overcome the inertia, which helps to satisfy the minimum torque criteria for the optimal control-based motion.

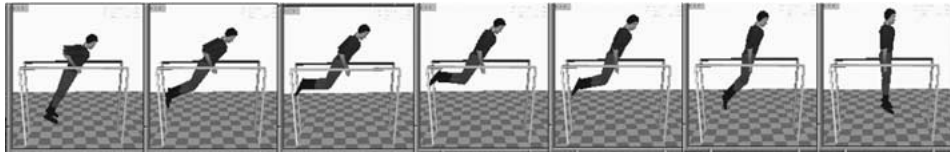


Figure 9. Dipdown motion counting in resistant joint torques.

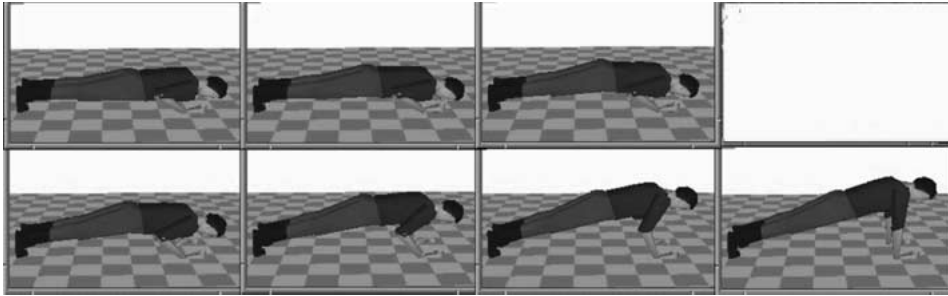


Figure 10. Pushup motion counting in resistant joint torques.

7.3. CLOSED LOOP CHAIN MODEL

The following is an example of a pushup motion planning. Since both hands and feet are in contact with ground, we simplify it as a single closed chain model. Using our method, we ‘cut’ the wrist joint, so that it becomes a single open chain. We can use the same method as that for single open chains, except that we need to have an additional constraint, which is the linear distance constraint between the ground and the hands. Referring to the loop closure $\Phi(\mathbf{q}) = \mathbf{0}$ in Equation (13), this constraint ensures that the hand is in contact throughout the motion. Figure 10 shows the motion sequence for the optimum motion simulated on Earth.

8. Conclusion

Human motion simulation requires the generation of human body movement patterns from a starting posture to a final posture to accomplish a manual task in dynamically correct virtual environments. The main focus of the paper has been the development of efficient and compact dynamic motion generation techniques for the above applications. To achieve this goal we have developed a computational motion control system that can carry out the computation of multibody dynamics and the respective gradients in an efficient recursive way. Therefore, it fits naturally and efficiently the nonlinear programming paradigm. The coupling of recursive dynamics and optimal control has shown to be an efficient way to deal with a variety of human motions consisting of different kinematic chains. The use of the spatial notation to formulate the necessary analytic gradient computations has provided a better-conditioned optimization computation and allows the robust and efficient

implementation of our method. This approach provides a systematic methodology for graphics animators to produce realistic animations of humans and animals.

The implemented approach works well for the tested problems. The joint torque trajectories along with the resulting motion can be evaluated. This provides a powerful tool for human motion studies, as there is no direct method to measure and record joint torque trajectories *in vivo*.

This is particularly promising for space exploration applications, where ergonomic problems in unfamiliar environments such as in space or other planets can occur. With our technique of minimum torque motion planning, we are able to simulate and predict possible problems for ergonomic studies, including IVA and EVA missions, so that harmful joint motion can be identified and prevented.

The approach has been coded in *Jack*. The input to the approach is the inertia and kinematic information from a *figure* file. There is no need to re-derive the dynamic motion equations and their derivatives for different figures. It would be otherwise complicated, cumbersome and error-prone to use traditional variational formulation. Our simulation results have clearly demonstrated the efficacy of the method in several experiments involving a variety of topological structures.

Acknowledgments

This research has been supported by grants from NASA (NASA-96-OLMSA-01-147), an NSF-Career Award (NSF-IRI-9624604), and an ONR Young Investigator Proposal Award (ONR-YIP-N00014-97-1-0817).

References

1. Ahrikencheikh, C. and Seireg, A., *Optimized-Motion Planning: Theory and Implementation*, Wiley-Interscience, New York, 1994.
2. Alexander, R., 'A minimum energy cost hypothesis for human arm trajectories', *Biological Cybernetics* **76**, 1997, 97–105.
3. Angeles, J. and Ma, O., 'Dynamic simulation of n -axis serial robotic manipulators using a natural orthogonal complement', *The International Journal of Robotics Research* **7**(5), 1988, 32–47.
4. Ayoub, M.M. and Lin, C.J., 'Biomechanics of manual material handling through simulation: Computational aspects', *Computers Ind. Engineering* **29**(1), 1995, 427–431.
5. Badler, N.I., Phillips, C.B. and Webber, B.L., *Simulating Humans: Computer Graphics Animation and Control*, Oxford University Press, New York, 1993.
6. Bordlie, K.W., *The State of the Art in Numerical Analysis*, Chapter III, Academic Press, New York, 1977.
7. Brandl, H., Johanni, R. and Otter, M., 'A very efficient algorithm for the simulation of robots and similar multibody systems without inversion of the mass matrix', in *Proceedings of IFAC/IFIP/IMACS International Symposium on the Theory of Robots*, Vienna, Austria, 1986.
8. Brandl, H., Johanni, R. and Otter, M., 'An algorithm for the simulation of multibody systems with kinematic loops', in *Proceedings of the IFToMM Seventh World Congress on the Theory of Machines and Mechanisms*, Sevilla, Spain, 1987.

9. Churchill, S.E. and Oser, H., *Fundamentals of Space Life Sciences*, Vol. 2, Krieger, Malabar, FL, 1997.
10. Cohen, M.F., 'Interactive spacetime control for animation', in *Computer Graphics (SIGGRAPH '92 Proceedings)*, Vol. 26, 1992, 293–302.
11. Dennis, J.E. and Schnabel, R.B., *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice Hall, Englewood Cliffs, NJ, 1983.
12. Engeln-Mullges, G. and Uhlig, F. *Numerical Algorithms with C*, Springer-Verlag, Berlin, 1996.
13. Featherstone, R., *Robot Dynamics Algorithms*, Kluwer Academic Publishers, Boston, 1987.
14. Greenwood, D.T., *Principles of Dynamics*, Prentice Hall, Englewood Cliffs, NJ, 1988.
15. Gregory, J. and Lin, C., *Constrained Optimization in the Calculus of Variations and Optimal Control Theory*, Van Nostrand Reinhold, New York, 1992.
16. Lathrop, R.H., 'Constrained (closed-loop) robot simulation by local constraint propagation', in *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, San Francisco, CA, April, IEEE, New York, 1986, 689–694.
17. Lilly, K.W., *Efficient Dynamic Simulation of Robotic Mechanisms*, Kluwer Academic Publishers, Dordrecht, 1993.
18. Lo, J., Metaxas, D. and Badler, N.I., 'Controlling a dynamic system with open and closed loops: Application to ladder climbing', in *ASME Design Automation Conference Proceedings*, Sacramento, CA, September, ASME, New York, 1997.
19. Oh, S.Y. and Orin, D.E., 'Dynamic computer simulation of multiple closed-chain robotic mechanisms', in *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, San Francisco, CA, April, IEEE, New York, 1986, 15–20.
20. Plücker, J., *Neue Geometrie des Raumes gegründet auf die Betrachtung der geraden Linie als Raumelement*, B.G. Teubner, Leipzig, 1868.
21. Popovic, Z. and Witkin, A., 'Physically based motion transformation', in *Computer Graphics (SIGGRAPH '99 Proceedings)*, Vol. 33, 1999, 11–20.
22. Press, W. and Teukolsky, S. *Numerical Recipes in C*, Cambridge University Press, Cambridge, 1992.
23. Rockafellar, R.T., 'The multiplier method of hestenes and powell applied to convex programming', *Journal of Optimization Theory and Applications* **12**(6), 1973, 555–562.
24. Rose, C., Guenter, B., Bodenheimer, B. and Cohen, M.F., 'Efficient generation of motion transitions using spacetime constraints', in *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, 1996, 147–154.
25. Schaffner, G., Newman, D.J. and Robinson, S.K., 'Inverse dynamic simulation and computer animation of extra vehicular activity', in *AIAA 35th Aerospace Sciences Meeting*, Reno, NV, American Institute of Aeronautics and Astronautics, Washington, DC, 1997.
26. Skrinar, A., Burdett, R.G. and Simon, S.R., 'Comparison of mechanical work and metabolic energy consumption during normal gait', *Journal of Orthopedic Research* **1**(1), 1983, 63–72.
27. Vanderplaats, G.N., *Numerical Optimization Techniques for Engineering Design: With Applications*, McGraw-Hill, New York, 1984.
28. Walker, M.W. and Orin, D.E., 'Efficient dynamic computer simulation of robotic mechanisms', *Journal of Dynamic Systems, Measurement and Control* **104**, 1982, 205–211.
29. Winter, D.A., *Biomechanics and Motor Control of Human Movement*, second edition, John Wiley & Sons, New York, 1990.
30. Witkin, A. and Kass, M., 'Spacetime constraints', *ACM Computer Graphics* **22**(4), 1988.
31. Yamaguchi, G.T., *Performing Whole-Body Simulations of Gait with 3-D*, Springer-Verlag, 1990.
32. Zefran, M. and Kumar, V., 'Optimal control of systems with unilateral constraints', in *International Conference on Robotics and Automation*, Nagoya, Japan, 1995.