THE DYNAMICS OF ARTICULATED RIGID BODIES FOR PURPOSES OF ANIMATION

W. W. Armstrong and M. Green

Department of Computing Science
University of Alberta
Edmonton, Alberta,
T6G 2H1

ABSTRACT

Curves and surfaces satisfying continuity and smoothness conditions are used in computer graphics to fit spatial data points. In a similar fashion, smooth motions of objects should be available to animators in such a way that the dynamics are correct to the degree required for realism. The motion, like a curve or surface shape, should be controllable by easy manipulations of a set of control parameters or by real-time interaction between the animator and a scene generated by dynamic simulation. In this paper, the objects considered have the form of rigid links joined at hinges to form a tree. This is a reasonable first approximation to human and animal bodies. The equations of motion are formulated with respect to hinge-centered coordinates, and are solved by an efficient technique in time which grows linearly with the number of links.

RESUME

Les courbes et les surfaces lisses déterminées par des ensembles donnés de points sont souvent employées en infographie. D'une même manière, les mouvements lisses devraient être disponibles à partir d'une technique de spécification simple, de façon a donner une impression réaliste du point de vue dynamique. On envisage ici, le contrôle en temps réel (ou presque) d'une simulation dynamique, où les équations de mouvement, formulées autour des charnières d'un système de membres rigides en forme d'arborescence, sont résolues par une méthode dont le temps de calcul grandit linéairement dans le nombre de membres.

KEYWORDS: animation, dynamics, manipulators, robotics

1. INTRODUCTION

The animation of human and human-like characters is one of the major unsolved problems in computer animation (see for example [Badler 1982]). The key aspect of this problem is achieving realistic motion with a minimal amount of effort on the part of the animator. Several approaches to human figure animation have been tried in the past with some success.

One of the earliest approaches was to record or digitize the motion of a human [Calvert, Chapman, and Patla 1980]. The animated human figure mimicked this motion based on the collected data. One of the main problems with this technique was having a properly instrumented human actor perform all the actions required for the animation. Collecting and processing the motion data is a cumbersome task and there is little flexibility in editing the motion once the data has been collected. Another problem with the approach is that the human actor cannot perform dangerous actions (such as falling off a cliff) or simulate the motion of non-human characters. A related approach is to use human body positions as key frames in an animation sequence. This still requires the collection of human movement data (but with a lower volume) and has all the problems associated with key framing [Catmull 1978].

A more recent approach is based on developing a kinematic model of the human body [Zeltzer 1982]. This model is based on the anatomy of the human body and characteristics of its motion. Motion is achieved by a hierarchy of motor programs. The low level motor programs control the joint angles for a fixed set of joints. These motor programs are controlled by the middle level motor programs. The middle level motor programs can start and stop the low level programs based on the current state of the model (joint angles, center of mass, support, etc.). Both the low level and middle level motor programs are modeled as finite automata. This approach requires far less effort on the part of the animator but still has some problems. A separate set of motor programs is required for each type of motion. The study of human motion is required to construct these programs. The model can only react to the environment in a restricted way. For example, one of these models could walk over uneven terrain, but could not respond to someone pushing it.

The approach to human figure animation presented here is based on incorporating dynamics into the model of the figure. This added information greatly simplifies interacting with and controlling the model. Once the model has been properly constructed (see section 5) a wide range of motions can be achieved by applying forces or torques to joints in the model at key points in the animation. For repetitive motions these forces can be programmed in the same way as the kinematic models.

In our view of human figure animation a human figure model must have the following characteristics:

1) The model should produce realistic motion sequences when given realistic input data. In other cases the model should produce believable results.

2) The amount of information the animator must provide should be minimal and proportional to the complexity of the motion. If the character is standing still or reacting to the environment in a standard way the animator should not need to specify any motion data.

3) The model should be able to react to and act on its environment. If someone pushes the character it should react to that force. Similarly if the character runs into an object it should respond in a natural way and possibly have some effect on the object (the object moves or falls over). This obviously requires a model that accounts for such physical properties as mass, force, inertia, torque, and acceleration.

In the next three sections of this paper, we develop the dynamics of articulated rigid bodies (skeletons) and an efficient method for solving their equations of motion. In the fifth section, a technique for developing dynamic models of human-like figures is presented. A simple example is used to illustrate this technique. The final section presents some of the ideas we are currently working on.

2. THE EQUATIONS OF MOTION

In this section, the equations of motion will be presented and explained in a tutorial fashion. The reader is assumed to be somewhat acquainted with the elementary concepts of dynamics: velocity, mass, force, acceleration, angular velocity, moment of inertia, torque, angular acceleration etc. [Goldstein 1959]. A treatment of manipulator dynamics based on Lagrangian mechanics is given in a book by R. P. Paul [Paul 1981]. Hollerbach has given a recursive Lagrangian formulation of manipulator dynamics [Hollerbach 1980] and has discussed Newtonian formulations similar to this one [Hollerbach 1979]. Our treatment is a generalization of work done for the case of linear linkages [Armstrong 1979], with more attention paid to computational efficiency.

Consider a physical quantity, such as a force. It is a vector, and has a meaning independent of any coordinate system. To represent it analytically, one has to introduce a system of coordinates using a frame, which consists of three mutually orthogonal unit vectors. The frames will always be right-handed in this paper. In a given frame, a vector is characterized by three components arranged in a one-column matrix. We shall always use this representation in our formulation of the equations of motion.

For dynamic modeling of linkages, it is convenient to choose frames which move along with the links, as though rigidly attached to them. This is particularly useful since, in the moving frame, certain quantities are constant, such as the representation of the vector from a hinge of the link to the link's centre of mass. To represent positions, we use the vector from the origin of the frame to the position in question. We note, though, that the origin of the frame is unimportant in determining the representation of a vector. In addition to frames which move with the links, we also use a fixed, non-rotating inertial frame.

The transformation of the representation of a vector, a 3-by-1 (column) matrix, from one frame to another is done by multiplying the representation on the left by a 3-by-3 orthogonal matrix. The inverse transformation is done using the transpose of the latter. We shall indicate orthogonal matrices by beginning their names with the letter R.

The cross-product of two vectors can be carried out using the representations of the operand vectors and the result in a certain frame, and will be denoted in this paper by the cross symbol ($\times$) as an infix operator.

The unit 3-by-3 matrix will be denoted by **I**. There is a very useful operation on a 3-vector v, with components $v1$, $v2$, $v3$, to get a 3-by-3 matrix V such that for any 3-vector w, the vector $v \times w$ is equal to the product of V and the column-vector w. This is the "tilde" operation:

$$\tilde{v} = V = \begin{bmatrix} 0 & -v3 & v2 \\ v3 & 0 & -v1 \\ -v2 & v1 & 0 \end{bmatrix}$$

We shall use the following quantities, some of them shown in fig. 1. Lower-case letters denote scalars (which are subscripted with the appropriate link number) and representations of vectors (superscripted), while upper-case letters denote matrices. A link, other than the root link of the tree (number 1), has one proximal hinge connecting it to its parent, which is closer to the root, and, except for leaves of the tree, one or more distal hinges. The number of a link is also used to number its proximal hinge. We denote by S_r the set of all links having link r as parent.

Scalar:

m_r the mass of link r;

Representations in the inertial frame:

a_G the acceleration of gravity;

p^r the position vector of the hinge of link r which joins it to its parent, which we shall call the proximal hinge of link r;

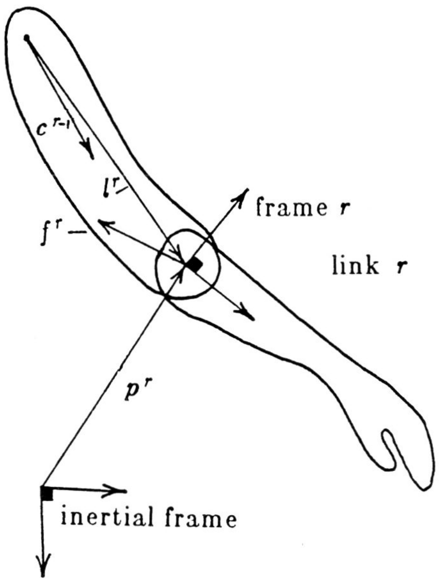

Fig. 1

v^r the velocity of the proximal hinge of link r;

f_E^r an external force acting on link r at the proximal hinge;

g_E^r an external torque acting on link r;

Representations in the frame of link r:

a^r the acceleration of the proximal hinge of link r;

ω^r the angular velocity of link r;

$\dot{\omega}^r$ its angular acceleration;

c^r the vector from the proximal hinge to the centre of mass of link r;

f^r the force which link r exerts on its parent at the proximal hinge;

g^r the torque which link r exerts on its parent at the proximal hinge;

Represented in the frame of the parent of link r:

l^r the vector from the proximal hinge of the parent of link r to the proximal hinge of link r

(constant in the frame of the parent);

Rotation matrices:

R^r converts vector representations in the frame of link r to the representations in the frame of the parent link;

$R^{r\,T}$ the inverse (= transpose) of R^r;

R_I^r converts from representations in frame r to representations in the inertial frame;

$R_I^{r\,T}$ the inverse (= transpose) of R_I^r

Matrix represented in frame r:

J^r the moment of inertia matrix of link r about its proximal hinge;

The first equation of motion expresses the fact that the rate of change of angular momentum of link r is equal to the applied torques from various sources.

$$J^r \dot{\omega}^r = g_\Sigma^r - m_r \, c^r \times a^r + \sum_{s \in S_r} l^s \times R^s f^s \quad (1)$$

where

$$\begin{aligned} g_\Sigma^r = & - \omega^r \times (J^r \omega^r) - g^r \\ & + \sum_{s \in S_r} R^s g^s + R_I^{r\,T} g_E^r \\ & + m_r \, c^r \times R_I^{r\,T} a_G \end{aligned} \quad (2)$$

The right-hand sides of (1) and (2) will now be explained. The part g_Σ^r has been separated out since that will help to explain the solution of the equations as well as accelerate the solution implemented on the computer. The term $- m_r \, c^r \times a^r$ comes from the fact that the frame in which this equation is formulated is accelerating with respect to the inertial frame, giving the effect of the force $- m_r \, a^r$ applied at the centre of mass of the link. In the next term, the forces f^s coming from all the sons of link r are first transformed from the coordinates of the son link, where they are represented, to the frame of r by applying R^s. Then the cross product of l^s with the force gives the torque at the proximal hinge of link r due to the force from the son link.

In equation 2, the term $- \omega^r \times (J^r \omega^r)$ is a torque coming from the rotation of the frame r with angular velocity ω^r which causes the angular momentum, $J^r \omega^r$, to appear to rotate. This torque term would not appear if the equations had been formulated in the inertial frame. In the inertial frame, however, the inertia matrix J^r would not be constant, making that frame less appropriate for formulating the equations for purposes of computer simulation.

The torque $- g^r$ is the negative of the torque which, by definition, link r exerts on its parent at its proximal hinge. It is the parent's reaction which is "equal and opposite". To be added in next are the torque terms coming from the son links, which must be converted from their representations in the sons' frames, and the external torque, which must be converted from its representation in the inertial frame. It doesn't matter where the torques are applied, since the links are considered rigid. Finally, the force of gravity, $m_r \, a_G$, converted from the inertial frame, causes a torque at the proximal hinge of link r when applied at its centre of mass.

The next equation of motion gives the force f^r acting on the parent of link r at the proximal hinge of link r. As above, it is convenient to separate out a term f_Σ^r which does not involve the quantities a^r or others which depend on it according to our solution method.

$$\begin{aligned} f^r = & f_\Sigma^r - m_r \, a^r + m_r \, c^r \times \dot{\omega}^r \\ & + \sum_{s \in S_r} R^s f^s \end{aligned} \quad (3)$$

where

$$\begin{aligned} f_\Sigma^r = & - m_r \, \omega^r \times (\omega^r \times c^r) \\ & + R_I^{r\,T} (f_E^r + m_r \, a_G) \end{aligned} \quad (4)$$

In equation 3, the term $- m_r \, a^r$ comes from the fact that the frame r is accelerating, and the next term from the fact that it is rotating at an accelerating angular velocity, which causes the centre of mass to accelerate with respect to the inertial frame. Finally, the forces from the son links are converted to frame r and communicated to the parent at the hinge.

In equation 4, we get first a centrifugal force from the rotation of the frame and second a contribution from the external force and gravity acting on the link.

The last equations describing the motion relate the acceleration at the proximal hinge of a son link s of link r to the linear and angular accelerations at the proximal hinge of r:

$$R^s a^s = a_c^s + a^r - l^s \times \dot{\omega}^r \quad (5)$$

where

$$a_c^s = \omega^r \times (\omega^r \times l^s) \quad (6)$$

In summary, then, the equations of motion for each link are formulated in terms of a moving frame attached to the proximal hinge of the link, and consist of equations giving the effect of torques (1), (2), equations giving the effect of forces (3), (4), and equations relating the accelerations at parent and son nodes (5), (6). In order to get the motion, we must solve these equations either to get the torques given the motion, which would be reasonable to control the linkages along a prescribed path, or to get the motion (accelerations, velocities, positions and orientations of the links through time) given the torques at the hinges and the external forces and torques. We choose the latter approach, since our aim is ultimately to let the animator control the motion by application of torques using hand and other controllers in real time. The previous approach has been treated by Luh et al. [Luh, Walker, and Paul 1979].

3. SOLUTION OF THE EQUATIONS OF MOTION.

Methods for converting the equations of section 1 to a matrix form are well-known. The resulting square matrix can be quite large, with a line for every degree of freedom in the system. Here we have six degrees of freedom for link 1, the root, three for position and three for orientation, and three degrees of freedom for the orientation of every other link (the hinges constrain the position, but not the orientation in our case). A simple model of the human body without hands or feet has at least 12 links, leading to a 39-by-39 matrix. We can expect this number to increase drastically if the links are required to be flexible or deformable in any way. The growth in terms of the number of links is quadratic by that technique. The method we are going to use grows linearly with the number of links, and is appropriate, we feel, for animation purposes, where the number of links may be large.

The hypothesis which aids in solving the equations is that there are linear relationships between the acceleration $\boldsymbol{a}^r$ of the link and the amount of angular acceleration it undergoes and between $\boldsymbol{a}^r$ and the reactive force on the parent. Think of giving the configuration of links distal to a certain hinge a push at the hinge which causes a certain acceleration. This will cause a certain angular acceleration and a certain reactive force, which can be expressed as follows:

$$\dot{\boldsymbol{\omega}}^r = K^r a^r + d^r \tag{7}$$

$$f^r = M^r a^r + f'^r \tag{8}$$

If we assume this linearity for all the sons of link r, then we can show it inductively for link r, and at the same time develop a computational method for solving the equations of motion by calculating the "recursive" coefficients K^r, d^r, M^r, and f'^r from the distal links towards the root. We encourage the reader to write down equation (1) and carry out the following simple substitutions: First the quantity f^s of a son link s is replaced by the corresponding expression from (8) with s instead of r. Then the acceleration $\boldsymbol{a}^s$ appearing therein is rewritten using the right hand side of (5) prefixed by $R^{s\,T}$. The cross product $l^s \times \dot{\boldsymbol{\omega}}^r$ is replaced by the product of the matrix $\tilde{l^s}$ with the column vector $\dot{\boldsymbol{\omega}}^r$. This enables us to collect the terms in $\dot{\boldsymbol{\omega}}^r$ and obtain

$$\begin{aligned}\dot{\boldsymbol{\omega}}^r = T^r \,(\, g_\Sigma - m_r\, c^r \times a^r \\ + \sum_{s \in S_r} \tilde{l^s}\, R^s \,(\, f'^s + M^s R^{s\,T} (\, a_c^s + a^r\,)))\end{aligned} \tag{9}$$

where we have set

$$T^r = (\, J^r + \sum_{s \in S_r} \tilde{l^s}\, R^s M^s R^{s\,T} \tilde{l^s}\,)^{-1} \tag{10}$$

Now we can determine the recursive coefficients for (7):

$$K^r = T^r \,(\, \sum_{s \in S_r} \tilde{l^s}\, R^s M^s R^{s\,T} - m_r\, \tilde{c^r}\,) \tag{11}$$

and

$$\begin{aligned}d^r = T^r \,(\, g_\Sigma + \sum_{s \in S_r} \tilde{l^s}\, R^s \,(\, f'^s \\ + M^s R^{s\,T} a_c^s\,))\end{aligned} \tag{12}$$

The next step is to determine the recursive coefficients for the force f^r. We first substitute in (3) for the forces f^s using (7) for s instead of r. The quantity $\boldsymbol{a}^s$ is replaced using (5), and the resulting $\dot{\boldsymbol{\omega}}^r$ values are replaced using (7).

This gives

$$\begin{aligned}M^r = -\, m_r \mathbf{I} + \mathrm{m_r}\, \tilde{\mathrm{c}}^{\mathrm{r}}\, \mathrm{K^r} \\ + \sum_{s \in S_r} R^s M^s R^{s\,T} (\, \mathbf{I} - \tilde{\mathrm{l}}^{\mathrm{s}}\, \mathrm{K^r}\,)\end{aligned} \tag{13}$$

and

$$\begin{aligned}f'^r = f_\Sigma + m_r\, c^r \times d^r \\ + \sum_{s \in S_r} (\, R^s f'^s \\ + R^s M^s R^{s\,T} (\, a_c^s - l^s \times d^r\,))\end{aligned} \tag{14}$$

4. COMPUTATION OF THE SOLUTION TO THE EQUATIONS

There are certain frequently used constants which should be precomputed:

J^r, $m_r \mathbf{I}$, $m_r c^r$, $m_r \tilde{c^r}$, $m_r a_G$, $\tilde{l^r}$.

At each iteration step, the equations of motion must be solved for the angular accelerations of the links by first determining the coefficients of the linear expressions (7) (8) in a pass inward from the distal links to the root link. Then the equation (8) for the root link, number 1, can be used to obtain the acceleration $\boldsymbol{a}^1$, since f^1, the force on the parent of the root link, is zero (there is no parent). Equation (5) can then be used repeatedly to get the acceleration of the son links, whose angular accelerations are computed using (7), now with known coefficients, on a pass outwards from the root link. The constraint forces f^r at the hinges can also be computed during this pass, but this is not required unless we are checking the solution.

After the solution phase at each time step, comes the integration phase, where $\dot{\boldsymbol{\omega}}^r$ is multiplied by the time step $\boldsymbol{\delta t}$ to get the increment of $\boldsymbol{\omega}^r$, and the latter is multiplied by $\boldsymbol{\delta t}$ to get an incremental rotation vector for the link. The incremental rotation vectors are used to update the rotation matrices which specify the orientations of the links: Rf, R^s (for son links only). The linear acceleration and velocity of link 1 can also be determined as well as its position, which, together with the orientations of all links, given by Rf, allows all the hinge positions in the inertial frame to be determined (for the purpose of graphics display, for example).

There is one deficiency in the above method, however, namely that some of the quantities which are computed at each integration step can be expected to vary slowly, and hence recomputation is not required at every increment $\boldsymbol{\delta t}$. Hence all the computations will now be divided into two bands: a "fastband" executed at each time step, and a "slowband" executed completely once every $\boldsymbol{\nu}$ executions of the fastband. The slowband computation can be split into $\boldsymbol{\nu}$ parts and executed uniformly over the fastband iterations in a real-time system.

The resulting organization of the computation is as follows:

Slowband computations inbound (i. e. from the leaves to the root):

Do for each link r starting at the leaves of the tree:

$$\textit{Compute } a_c^s = \omega^r \times (\omega^r \times l^s) \tag{15}$$

$$\textit{Compute } Q^s = R^s M^s R^{s\,T} \text{ for } s \in S_r \tag{16}$$

$$\textit{Compute } W^s = \tilde{l}^s Q^s \text{ for } s \in S_r \tag{17}$$

$$\textit{Compute } T^r = (J^r + \sum_{s \in S_r} W^s \tilde{l}^s)^{-1} \tag{18}$$

$$\textit{Compute } K^r = T^r (\sum_{s \in S_r} W^s - m_r \tilde{c}^r) \tag{19}$$

$$\begin{aligned}\textit{Compute } M^r = {} & (m_r \tilde{c}^r) K^r - m_r \mathbf{I} \\ & + \sum_{s \in S_r} Q^s (\mathbf{I} - \tilde{\mathrm{l}}^{\mathrm{s}} \mathrm{K}^{\mathrm{r}})\end{aligned} \tag{20}$$

$$\begin{aligned}\textit{Compute } g_\Sigma^{1r} = {} & - \omega^r \times (J^r \omega^r) \\ & + Rf^{\,T} g_E^f \\ & + (m_r c^r) \times Rf^{\,T} a_G\end{aligned} \tag{21}$$

(assuming g_E^f is slowly-varying).

$$\begin{aligned}\textit{Compute } f_\Sigma^\xi = {} & - \omega^r \times (\omega^r \times (m_r c^r)) \\ & + Rf^{\,T} (f_E^f + m_r a_G)\end{aligned} \tag{22}$$

(assuming f_E^f is slowly-varying).

Fastband computations, inbound:

Note: at this point the hinge torque values g^r from the controls manipulated by the animator are inserted. One approach to controlling these values is presented in the following sections, and others are under development.

Do for each link r, starting distally and proceeding towards the root:

$$\textit{Compute } g_\Sigma^\xi = g_\Sigma^{1r} - g^r + \sum_{s \in S_r} R^s g^s \tag{23}$$

$$\begin{aligned}\textit{Compute } d^r = T^r (g_\Sigma^\xi + & \sum_{s \in S_r} l^s \times \\ & (f'^{\prime s} + Q^s a_c^s))\end{aligned} \tag{24}$$

$$\begin{aligned}\textit{Compute } f'^r = {} & f_\Sigma^\xi + (m_r c^r) \times d^r \\ & + \sum_{s \in S_r} (f''^s \\ & + Q^s (a_c^s - l^s \times d^r))\end{aligned} \tag{25}$$

$$\textit{Compute } f''^r = R^r f'^r \tag{26}$$

Fastband computations, outbound:

$$\textit{Compute } a^1 = - (M^1)^{-1} f'^1 \tag{27}$$

$$\textit{Compute } \dot{\omega}^1 = K^1 a^1 + d^1 \tag{28}$$

Do the following for each link r other than the root outbound:

$$\textit{Compute } a^r = R^{r\,T} (a_c^r + a^p - l^r \times \dot{\omega}^p) \tag{29}$$

where p is the parent of the link r.

$$\textit{Compute } \dot{\omega}^r = K^r a^r + d^r \tag{30}$$

$$\textit{Compute } f^r = M^r a^r + f'^r \tag{31}$$

if required for a check of the solution.

Integration, fastband:

Compute $\boldsymbol{\omega}^r + \delta t\, \dot{\boldsymbol{\omega}}^r$ (32)

and assign to $\boldsymbol{\omega}^r$.

Compute $\delta \boldsymbol{u}^r + \delta t\, \boldsymbol{\omega}^r$ (33)

and assign to $\delta \boldsymbol{u}^r$, where the latter quantity accumulates a small rotation of link r as an "infinitesimal vector".

Integration, slowband:

Here the rotation matrices are updated, a costly process which should be done only when the errors introduced by tardy updating are sufficiently small.

Do the following for each link starting distally:

Compute $R_f^r (\mathbf{I} + \widetilde{\delta \mathbf{u}}^r)$ (34)

and assign to R_f^r. When this has been done $\delta \boldsymbol{u}^r$ must be reset to the zero vector to begin its accumulation again during the fastband integration. Also compute and store the transpose $R_f^{r\,T}$.

Orthonormalize R_f^r (35)

to prevent accumulation of errors which would ultimately destroy the simulation.

Compute $R_f^{r\,T} R_f^s$ (36)

for every $s \in S_r$, and assign to $_rR^s$. Also compute and store the transpose $R^{s\,T}$.

Compute $v^r + \delta t\, R_f^r\, a^r$ (37)

and assign to v^r.

Compute $p^r + \delta t\, v^r$ (38)

and assign to p^r.

In implementing the above solution, it is of critical importance to choose an integration step size δt several times smaller than the period of any frequency present in the system. Otherwise numerical instability will destroy the simulation. (It is amusing to watch the stick figure's arms begin to oscillate wildly as it flies off to oblivion though!) The highest frequencies will probably be the rotations of small links about their long axes under the influence of elastic torques at the hinges which maintain the alignment of the links. What one can do to keep these frequencies low is to falsely enlarge the moments of inertia about these axes. It is an open question to what extent this can be done without destroying the realism of the simulation. Such tricks are often applied in aircraft simulators, for example, and animation can undoubtedly benefit from using them heavily. A method for removing two degrees of freedom from the hinges has been described elsewhere [Armstrong 1979].

Other factors which must be appropriately adjusted are the frictions of the hinges, which create viscous damping torques proportional to the relative angular velocities of the two links involved. As we shall see, creating a zone of free play at the hinges, where the stiffness of the hinge has no effect, allows the friction to absorb the energy erroneously introduced by numerical inaccuracies, leading to a more stable model.

5. DEVELOPING FIGURE MODELS

This section describes one approach to developing human figure models based on the dynamics presented in the previous three sections. These models represent the positions and orientations of the figure along with how it behaves when it interacts with its environment. This approach is illustrated by an example presented at the end of this section.

Our approach to human figure modeling consists of three steps. The first step is developing a skeleton for the figure. This skeleton consists of a set of links representing the figure's limbs and a set of joints representing the places where the limbs are connected. The set of links representing the figure form a tree with each link having at most one parent. The skeleton represents the topology and physical properties of the figure. The skeleton itself can be viewed as a stick figure. The graphical display of the figure used in an animated sequence is generated from the skeleton. Techniques for converting skeletons to graphical displays can be found in [Burtnyk and Wein 1976] and [Zeltzer 1982]. Associated with each link in the model is its mass, center of mass, and inertia. These quantities must be calculated or estimated from the figure being modeled.

In order to have realistic motion the skeleton must be augmented with additional information pertaining to the behavior of its joints. The second step in this modeling process is determining this information and incorporating it into the figure model. This step is divided into two parts. In the first part the support for the skeleton is developed. The force of gravity is continuously acting on the figure and unless it is counteracted the figure will fall or collapse. There are several ways in which the figure can be supported. One way is to attach an upward pointing force (on average equal to the mass of the object times the acceleration of gravity) to one of the links in the model. The link used depends upon the nature of the animation. For walking and similar actions the force is attached to the foot that is in contact with the ground. For gymnastics the force might be attached to the center of mass of the figure. Another approach to counteracting the force of gravity is discussed in the example at the end of this section.

The joints in the model cannot be free to move any way they wish. Each joint will have a certain range of angles which it is capable of moving through. Angles outside of this range will appear unnatural. One way of constraining the motion of the joints is to stop the joint rotation when it reaches the end of its natural range. This is not a satisfactory solution for the following reasons. First, it leads to very unnatural motions. When people move their limbs there is a definite acceleration and deceleration at the end of the motion, they don't come to an abrupt stop at the end of the motion [Schmidt 1982]. Second, the abrupt changes caused by the clamping of the rotation angles violates the assumptions used in the development of the equations presented in section 3. Third, sometimes the animator wants a limb to move outside its normal range. If a large enough force is applied to the model, one or more of the limbs should move out of its natural range.

The approach to joint behavior that we have developed is based on determining the natural range of the joint and then applying torques about the joint to keep it in this range. At each time step the torque applied to a limb is a function of the angle between it and its parent. This torque will move the limb towards the center of its normal range. The torque functions we use in our model have the shape shown in fig. 2. As can be seen from this figure there is a range of angles where no restorative torques are applied. This area of free play leads to more natural motions and decreases the tendency of the limbs to oscillate. The size of the free play zone and the slope of the function at the ends of this zone determine the behavior of the joint. A narrow free play zone with steep sides will give rise to a stiff joint. The free play zone can be moved around in order to simulate different types of motion. Note that there are torques that will cause the joint to act unnaturally.

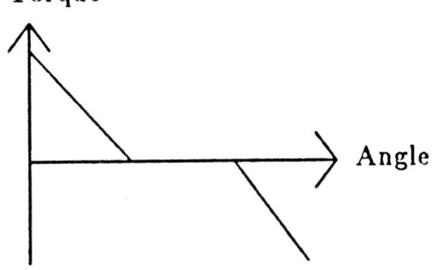

Fig. 2. A typical torque function

At this point if the model is placed in a simple environment with only gravity it will eventually reach equilibrium (in practice this occurs after one or two seconds of simulation time depending on how close the initial state of the model is to equilibrium). If the environment acts on the model it will essentially roll with the punches. The third step in our modeling approach is determining the external forces and torques required for specific motions such as walking, throwing, or diving. In most cases these forces and torques are spikes applied at fixed intervals of time or when certain conditions hold.

As an example of how this approach works consider the problem of animating a finger tapping on a table. The skeleton for this finger is shown in fig. 3. This skeleton has five links, three of which are visible in the figure. The middle three links represent the visible part of the finger and for the purposes of this example all have the same length and mass. At the left end of the figure is a link of length zero representing the attachment of the finger to the hand. This end of the finger should have a relatively constant position, therefore, it has a much higher mass and inertia (about 100 times) than the other links. In order to support the finger a variable force is applied to this link. The strength of this force is inversely proportional to the distance between the current position of the link and its desired position. The fifth link is at the other end of the finger and also has zero length. This link serves as a convenient place for applying an external force.

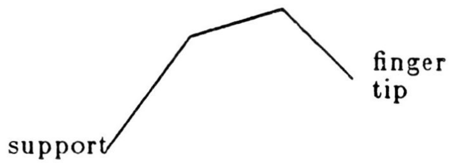

Fig. 3. Skeleton for a finger

Next the torque functions for the joints must be determined. In this case we ran the simulation for several seconds without torque functions and recorded the joint angles that appeared natural. The torque functions were based on these observations and several runs to determine appropriate slopes of the functions.

Finally, in order for the finger to tap, an external force must be periodically applied to it. In this case, a spike lasting 0.11 seconds is applied to the finger tip every 2 seconds. The model is allowed 2 seconds to stabilize before the first spike is applied.

The simplicity of this example indicates the ease with which realistic animation can be produced with this approach.

This example has been run on both a SUN workstation (a MC68010 with no floating point hardware) and a VAX 11/780 with a floating point accelerator. A time step of 0.01 seconds was used in these runs and the slow band was calculated at every time step. On the SUN, 604 cpu seconds were required to produce an 8-second animated sequence of the finger tapping (a wire frame perspective display of the skeleton was produced every 0.1 second). On the SUN hardware the simulation is a factor of 75 slower than real time. The same program on the VAX requires 89 seconds to produce 8 seconds of animation. If the slow band is calculated every other time step, the SUN requires 385 seconds and the VAX requires 56.6 seconds to produce 8 seconds of animation. On the VAX this is close enough to real time to get some feel for the motion while it is being calculated.

6. CONCLUSIONS AND FURTHER WORK

In this paper we have presented the dynamics of articulated rigid bodies, an efficient method for solving their equations of motion, and a technique for developing human figure models based on these dynamics. The example in section 5 shows that these equations can be solved almost in real time.

This work suggests a number of topics for further research. One possible topic is reducing the time required to solve the equations of motion. Two possible approaches to this problem are developing better implementations and designing special hardware for their execution. Another research topic is developing better techniques for controlling the figure. At the present time we have not thoroughly explored all the possibilities for supporting the model against the force of gravity. Also means for controlling complicated motions needs to be developed. Finally the figure models need to be integrated with an environment where they can interact with each other and long range planning can occur.

REFERENCES

[Armstrong 1979] Armstrong, W. W., "Recursive Solution to the Equations of Motion of an N-link Manipulator", Proc. Fifth World Congress on Theory of Machines and Mechanisms, Montreal, Vol. 2, Am. Soc. of Mech. Eng., pp. 1343-1346, 1979.

[Badler 1982] Badler, N. (ed.), Special Issued of IEEE Computer Graphics and Applications on Human Body Models and Animation, vol. 2, no.9, 1982.

[Burtnyk and Wein 1976] Burtnyk, N., Wein, M., "Interactive Skeleton Techniques for Enhancing Motion Dynamics in Key Frame Animation", CACM vol.19, no.10, p.564, 1976.

[Calvert, Chapman and Patla 1980] Calvert, T.W., Chapman, J., Patla, A., "The Integration of Subjective and Objective Data in the Animation of Human Movement", Siggraph'80 Proceedings, p.198, 1980.

[Catmull 1978] Catmull, E., "The Problems of Computer-Assisted Animation", Siggraph'78 Proceedings, p.348, 1978.

[Goldstein 1959] Goldstein, H., *Classical Mechanics,* Addison-Wesley, Reading Mass., 1959.

[Hollerbach 1979] Hollerbach, J. M., "An Iterative Lagrangian Formulation of Manipulator Dynamics", Massachusetts Institute of Technology, AI Laboratory, A. I. Memo No. 533, Revised March. 1980.

[Hollerbach 1980] Hollerbach, J. M., "A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamics Formulation", IEEE Trans. on Systems, Man and Cybernetics, SMC-10, 11, pp. 730-736, 1980.

[Luh, Walker, and Paul 1979] Luh, J., Walker, M., Paul, R., "On-line Computational Scheme for Mechanical Manipulators", 2nd. IFAC/IFIP Symposium on Information Control Problems in Manufacturing Technology, Stuttgart, Germany, 1979.

[Paul 1981] Paul, P., *Robot Manipulators: Mathematics, Programming, and Control,* MIT Press, Cambridge, Mass., 1981.

[Schmidt 1982] Schmidt, R.A., *Motor Control and Learning: A Behavioral Emphasis,* Human Kinetics Publishers, Champaign Illinois, 1982.

[Zeltzer 1982] Zeltzer, D., "Motor Control Techniques for Figure Animation", IEEE Computer Graphics and Applications, vol. 2, no. 9, p.53, 1982.