NEAR-REAL-TIME CONTROL OF HUMAN FIGURE MODELS

W.W. Armstrong, M. Green, and R. Lake
Department of Computing Science
University of Alberta
Edmonton, Alberta
Canada

ABSTRACT

The animation of human figures is one of the major problems in computer animation. A recent approach to this problem is the use of dynamic analysis to compute the movement of a human figure given the forces and torques operating on the body. One of the main problems with this technique is computing the forces and torques required for particular motions. As a solution to this problem an interactive interface to our dynamics routines has been produced. This interface, along with a collection of low level motion processes, can be used to control the motion of a human figure model. In this paper both the user interface to our dynamics routines and the motion processes that we use are described.

KEYWORDS: human figure animation, dynamic analysis, interactive control of human figures

1. Introduction

One of the more challenging parts of computer animation is the animation of human figures and other articulated bodies (for example, robots and animals). Over the past decade a number of techniques have been developed for the animation of human figures. These techniques vary from digitizing human movement, to the development of kinematic models of human motion. Recently the dynamic analysis of human motion has been proposed as a way of animating human figures [Armstrong and Green 1985a, Wilhelms and Barsky 1985].

Dynamic analysis has a number of advantages over other approaches to human animation. Since this technique is based on well known techniques from physics and robotics, it is capable of producing very realistic motion. The motion of the human figure is controlled by forces and torques that are applied to the limbs of the body. In most motions only a small number of the limbs are actively involved, these limbs are called the controlled limbs. The other limbs in the body either maintain the same relative position, or follow the motion of the controlled limbs. The latter motion can be automatically produced by the dynamics software, therefore, the animator only needs to specify motion information for the controlled limbs. The small volume of information required to produce motion could lead to human animation systems that are much easier to use than existing systems.

There are two main problems associated with the use of dynamic analysis for human animation. The first problem is the amount of computer time required to compute the motion of the human figure. Traditional approaches to the computation of human motion require overnight batch runs for simple animation sequences [Wilhelms 1985]. We have developed an approach to the solution of the equations of motion that is significantly faster than other techniques. This approach can produce near-real-time animation on commonly available hardware. Some of our results in this area are described in section 2.

The second problem is determining the torques and forces required to produce a particular motion sequence. Animators work in terms of body positions and complex motions, such as walking and running. They have no experience with the torques and forces required to produce the motion they want. There are two parts to our solution to this problem. The first part is the development of a number of low level motion processes. These processes generate the torques and forces required to produce particular types of motion. The second part of the solution is an interactive user interface that allows the animator to specify values for the parameters used by the motion processes, or directly apply torques and forces to the body while it is in motion. The animator can obtain immediate feedback on the effects of changes in parameter values, or the effects of torques and forces. Through this interface the animator is able to experiment with different ways of producing motion, and develop a feel for how they can be used to produce the motion he or she wants. There is also the possibility of producing canned motions that can be called upon by the animator. These motions could be parameterized so they can be customized to a particular situation. The work we have done in this area is discussed in sections 3 and 4.

2. Near-Real-Time Dynamics

One of the main drawbacks to using dynamic analysis for human animation has been the amount of computing required. Some of the formulations of the equations of motion for human figures and techniques for their solution are based on the techniques developed in mechanical engineering for the analysis of general linkages such as those found in machines. The linkage structure of the human body is not as complicated as the systems studied in mechanical engineering, where any of the links in the mechanism could directly effect the motion of any of the other links. This can give rise to a graph structure for the links. On the other hand, the human body can be viewed as a tree of links with no interconnections between the leaves on different branches. This observation significantly simplifies the equations of motion and allows for efficient solution techniques. This version of the equations of motion and techniques for their solution have been described elsewhere [Armstrong and Green 1985a,b]. At this point we will summarize the results of this work.

Two implementations of our solution of the equations of motion have been produced. The first implementation is designed to run on a single processor. This implementation is written in C and currently runs on a DEC VAX 11/780, SUN workstation, and IRIS workstation. The time required to compute a motion sequence depends upon the inertias used for the body parts. The step size required for a stable solution of the equations of motion is proportional to the square root of the values of the inertias. In our current implementation the inertias are about a factor of 10 larger than those found in a human body. This results in a computation time that is within a factor of 3 to 10 of real-time depending on the complexity of the figure and the complexity of the motion. The animation produced by this implementation is fast enough to get a good feel for the motion while the program is running. The other implementation is designed to run on a network of processors [Armstrong et.al. 1986]. Since the human body can be viewed as a tree of limbs, subsets of limbs of the tree can be assigned to different processors, and a large amount of the computation can proceed in parallel. The results we have obtained so far indicate that real-time animation could be produced by a network of four SUN 3 workstations.

These results indicate that it is possible to construct a system where the animator can manipulate the dynamics of a human figure in real-time.

3. Control Strategies

Most human motion involves only a subset of the limbs in the body. When the animator develops these motions he or she will want to work with a small number of limbs at any point in time. Controlling more than two or three limbs in real-time is probably beyond the capabilities of most people. Thus, the animation system should allow the animator to build up his or her motion sequence on a limb-at-a-time basis. The main problem with this approach is that when one limb moves, the other limbs it is connected to are subjected to forces and torques as a result of its motion. This is a natural result of Newton's laws of motion. Some of these secondary motions may be desirable. For example, when the upper arm moves the animator will want the lower arm and all the limbs attached to it to also move. Other secondary motions are not desirable. A good example of this is when the figure reaches for an object only the arm should move and not the body as a whole.

The solution to this problem can be based on the use of a number of motion processes, which when added to the body model guarantee reasonable behavior. These motion processes are similar in function to the finite state automata used by Zeltzer [Zeltzer 1982]. A number of features of human motion can be handled by a collection of motion processes. Some of these features are: maintaining the same relative position between two connected limbs, balance, ground reaction, and the performance of simple motions, such as reaches. The motion processes can be divided into two basic categories; called limb processes and global processes. A limb process is responsible for the motion of only one limb. Each limb can have one or more motion processes associated with it at any one time. The number and types of motion processes on each limb are under the control of the animator. The global motion processes affect more than one limb. These processes are responsible for motions that require global knowledge of the state of the body. Examples of this type of process are balance and ground reaction.

The motion processes described in the following sections were motivated by the work that has been done in biomechanics (see [McMahon 1984a] for a good introduction to some of the relevant work). We have used biomechanics as a source of ideas for controlling the motion of the body, we are not trying to accurately model the human nervous or muscle systems.

3.1. Limb Motion Processes

The human figure model consists of a number of links, with each link representing a part of the body. The current model contains 14 links (head, neck, upper body, lower body, upper arm, lower arm, upper leg, lower leg, and foot). At the proximal end of each link there is a three-degree-of-freedom rotational joint connecting it to its parent. The upper body link has three extra degrees of freedom representing the translation of the body with respect to the world coordinate system. The current state of the model is given by the position of the upper body and the three rotation angles at each joint.

There are nine parameters that can be used to control the motion of each link. These parameters are the components of the the internal torques (torques generated at the joints), external torques (torques applied from outside of the body), and external forces (forces applied from outside of the body). Only the internal torques are used by the limb motion processes. At any point in time there can be one or more motion processes associated with each limb. Each of these processes contributes to the internal torque that is applied to that limb.

The motion processes are controlled by a joint information table. This table contains one entry for each degree of freedom in each joint. The table entry contains the state of the degree of freedom and parameter values that are required by the associated motion processes. The state is a bit vector that specifies the motion processes that are currently associated with that degree of freedom. There is one bit in this vector for each motion processes. If the bit is set, the motion process can affect that degree of freedom.

The motion processes are executed on each iteration of the dynamics calculations. Each degree of freedom in each limb is considered separately. At start of the processing for a degree of freedom, its internal torque is set to zero. Then the state bit vector is examined to determine the motion processes that are to be executed. Each motion process uses the current state of the link, plus its own parameters (stored in the joint information table) to compute a contribution to the internal torque. At the end of this process, new internal torques have been generated for each limb in the model.

At the present time we are using four motion processes. The first process, called free swing, is a null process that does not contribute to the internal torque of the joint. This process allows the joint to move freely (without any constraints) in the degree of freedom it is attached to.

The second motion process, called friction, generates a velocity dependent friction which is used to slow the limb down when it is in motion. The friction in a joint is proportional to the relative angular velocity of the limb with respect to its parent. The constant of proportionality can be interactively controlled by the animator. This model of friction agrees with results from biomechanics [McMahon 1984a].

The third motion process, called the maintain process, is used to maintain the relative angular positions between two adjacent limbs. When producing a motion, the animator may want only a subset of the limbs to move. The other limbs in the body should stay in the same relative positions. This motion process is used to achieve this goal. When using this motion process the animator specifies a parameter, called center, which is the desired angle between the limb and its parent. The motion process maintains the angle between the two limbs close to the value of center. The angle between limbs cannot be clamped to the center value for two reasons. First, this behavior is not realistic from a biological point of view. Second, fixing an angle adds a constraint to the dynam-

ics equations and would require reformulating the solution. The following function, T(x), is used to determine the torque applied to a joint given the angle, x, at the joint, and the desired angle, center.

$$T(x) = \alpha\,(\exp(\beta(x-\text{center})) - 1) \quad \text{if } x \geq \text{center} \qquad (1)$$

$$= -\alpha\,(\exp(\beta(\text{center}-x)) - 1), \quad \text{otherwise}$$

The parameters α and β, which can be set by the animator, determine the strength of the torque applied at the joint. Initial values are supplied for center, α, and β that will maintain the human figure in a standing position. The form of this motion process is based on results from studies on muscle dynamics [McMahon 1984a] [Hatze 1977].

The fourth motion process, called the simple move process, is used to move a limb from one position to another. The animator specifies the new angle between the limb and its parent, and this motion process produces a smooth motion between the current limb position and the new limb position. When the new limb position is reached the maintain process is invoked to maintain the new position. In order to move the limb from its old position to its new position a sequence of torques (one for each time step of the dynamics calculations) must be applied to the joint. This sequence of torques must satisfy two conditions. The first condition is that the torques must be strong enough to move the limb to its new position. The second condition is that the generated motion must appear to be natural.

In order to satisfy the first condition we use expression (1) to estimate the amount of torque required to reach the new position (the new position is used as the value of center). If this torque was applied to the joint, the limb would reach its new position within one or two iterations of the calculations. Time steps on the order of 0.01 seconds are usually used in the dynamics calculations, therefore, the limb would move from its original position to its new position in a small fraction of a second. For most types of motion this change of position is far too rapid. The torques produced by expression (1) are unrealistic, but they do guarantee that the limb will reach the new position, thus it is a good starting point for our calculations.

In order to produce more realistic motion we place two constraints on the torque produced by expression (1). The first constraint is that the torque cannot exceed a certain maximum value (these values can be found in tables of maximum torques for physical activities [Plagenhoef 1971]). The second constraint is that the rate of change of torque cannot exceed a maximum value (reasonable values for this parameter are scattered in the literature). When these two constraints are applied to the torques produced by expression (1), smooth motion is produced in the first part of the action. The main problem with this technique is that the motion does not slow down as the final position is reached. Even with this problem the results look fairly realistic.

The motion in the second half of the action can be improved by decreasing the torque applied to the limb as it approaches the new position. The angular mid-point of the motion is fairly easy to determine (the average of the initial and final angles). After this point the torque applied to the limb should be decreasing. This can be achieved by setting the maximum torque to the torque value at the mid-point of the action. At each iteration after the mid_point, the maximum torque is decreased by the square root of the ratio of the distance from the current position to the goal, to the distance from the center position to the goal. That is, we use the following expression:

$$\text{max\_t} = \text{center\_t} \quad ((\text{current} - \text{goal}) / (\text{half} - \text{goal})) \qquad (2)$$

where: center_t = the torque at the angular mid-point of the action
current = current joint angle
goal = final joint angle
half = the mid-point of the action

The above expression seems to produce a smooth motion in the second half of the action.

3.2. Global Motion Processes

The global motion processes use the states of several limbs in order to control the global motion of the body. These processes can generate torques and forces that are applied to several of the body's limbs. At the present time two global processes are used in our software. These processes maintain the balance of the figure, and its reaction to the ground.

At the present time, a very simple approach is taken to balancing the figure. First, the difference between the positions of the top and bottom of the body is calculated. A restorative force based on this difference is then applied to one or more limbs of the body, depending upon the type of motion. This technique can be used to keep the body in a standing position, but in general it is too restrictive. A more realistic balancing technique would be based on the limbs that are in contact with the ground. Torques generated by these limbs could be used to keep the body in balance.

One of the main problems with balance is that different types of balance may be required for different types of motion. In the case of diving and gymnastics, a balance process may hinder the motion. Walking is based on falling forward [McMahon 1984b], so an external balance process could make it impossible for the figure to walk. In other motions, such as reaching and lifting, balance is very important, so for these motions a balance process must be used. This suggests that either a sophisticated model of balance must be developed, or it must be under the control of the animator.

The human figure must be able to react to any of the objects it comes in contact with. The most common of these objects is the floor or ground. In equilibrium the floor will exert a force on the body equal to the body's weight. This solution cannot be used in animation, since the motion of the body will change the force applied to the floor by the body. The approach that we have used is to monitor the position of the body's feet (or another points of contact with the floor). A spring force is applied to the feet in order to keep the feet at the floor level. This technique works as long as the feet stay close to the floor. If the body is falling towards the floor, a more sophisticated technique is required. The friction between the feet and the floor must also be considered, otherwise the feet will slide all over the floor. A good discussion of ground reaction can be found in [Wilhelms 1985].

4. Software Architecture

The interactive animation system that we have developed is divided into two main components, which are shown in fig. 1. The first component, called the front end, is responsible for displaying the human figure model and interacting with the user. This component of the animation system resides on an IRIS 1400 workstation and is responsible for controlling the animation system. The second component is the dynamic analysis program. This program performs all the dynamics calculations for the human figure model. The second component can reside on the IRIS workstation, or on one or more of the other processors on our local network.

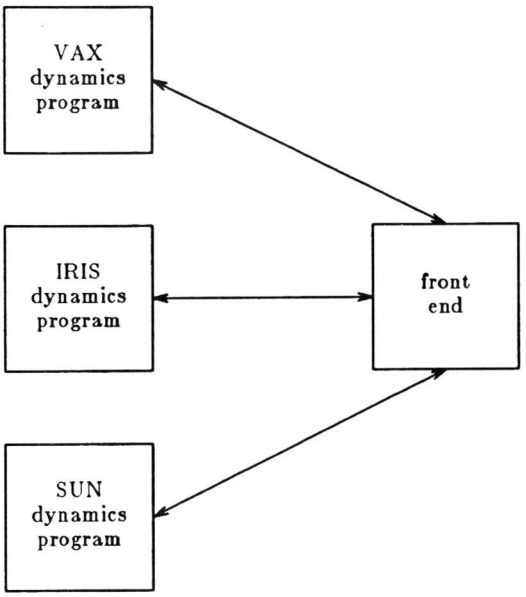

Fig. 1 Software Architecture

The two components of the animation system communicate by sending packets over an interprocess communications facility (either a pipe, or a socket in the case of an ethernet connection). The front end invokes the back end process when the user wants to perform dynamics computations. Upon invocation, the dynamics program reads a start-up file containing a number of parameters for the computation, and performs the first step in the computation. At the end of the first step the dynamics program sends a set of packets to the front end. There is one packet in this set for each limb in the body, giving its current joint angles. There is also a packet specifying the current position of the root limb within the world coordinate system. At this point the front end program responds with one or more packets. These packets are used to change the state of a limb, pass parameters for the motion processes, or specify a torque or force to be applied to the body. The last packet in this exchange is a Next_step packet sent from the front end to the dynamics program. At this point the dynamics program starts the next calculation cycle. This packet exchange ensures that the front end and the dynamics program are always in step.

A packet exchange need not occur at each time step of the computation. The time step used in the computation is of the order of 0.01 seconds. This time step is too fine for display and the types of control we are using. Currently, the packet exchanges occur every 0.05 seconds of simulation time. This rate is sufficient for both display and control.

This division of the animation system into separate processes has two main advantages. First, separating the dynamics computations from the display and user interface allows the use of either the single processor or distributed versions of the computations with the same user interface. In other words, as far as the animator is concerned interacting with the single processor and distributed version of the computations is the same. The only noticeable difference is the speed of computation. This allows the animator to take advantage of the available computing resources without changing his mode of operation. Second, the use of separate processes allows several people to work on the project without interfering with each other.

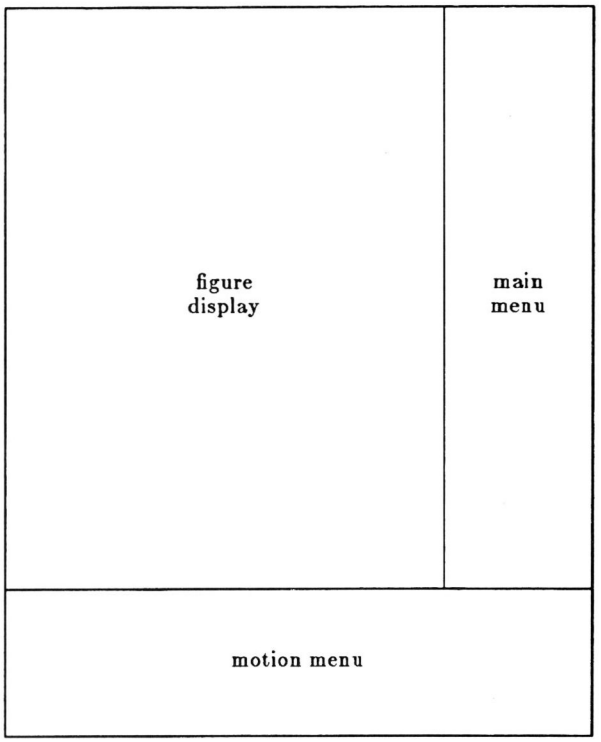

Fig. 2 Screen Layout

The screen layout for the front end is shown in fig. 2. The display screen is divided into three main sections, called the figure display, main menu, and motion menu. The figure display is used for displaying a graphical representation of the human body. The human body is represented by a three dimensional polygon model, with four to six polygons defining each of the limbs. Each of the polygons in a limb has a different colour facilitating the identification of the different sides of the figure. This type of model allows us to display the human figure at a rate of 14 frames per second.

The main menu contains a collection of commands that can be invoked when the dynamics computations are not being performed. The commands on this menu are used to change the eye position, record the motion produced by the dynamics routines, start the dynamics computations, playback a motion sequence that has been previously computed, save a motion sequence on a disk file, and retrieve a previously computed motion sequence from disk.

Once the dynamics computations have been started they can be interrupted in two ways. If the user presses one of the mouse buttons, the dynamics computations are suspended at the next packet exchange, and control is transferred to the main menu. The dynamics computations can be restarted by selecting the dynamics command from the main menu. This facility allows the user to change the eye position, or record parts of the motion while the computations are in progress.

The other way of interrupting the dynamics computations is to move the mouse into the motion menu area. At this point the cursor changes shape, and the user can select one or more commands from the motion menu. The motion menu contains the name of each joint in the body and the name of the parameters for the motion processes. In order to change a parameter value for a motion process, the user selects the joint and parameter name from the menu, and then selects the value command. At this point the user is prompted for the new

value for the parameter. Similarly the user can change the state of any of the limbs in the body. When the user is finished modifying the motion processes, he or she can move the mouse into the figure display in order to resume the computation.

This user interface has three main advantages over batch dynamics computations. First, at any point in the computation the user can suspend the computation, and then playback (in real-time) the motion sequence that has been produced. This allows the user to terminate computations that are not producing the desired motion before the end the the motion sequence. This saves both animator and machine time. Second, the animator can interactively change the motion as it is being computed. This allows the animator to react to the motion of the human figure, and frees him from precisely timing the movements of the figure. Third, the near-real-time computation of the motion allows the animator to experiment with different types of control strategies.

5. Summary

In this paper we have reviewed some of the work that has been done on applying dynamic analysis to the animation of human figures. We have also summarized the work we have done on producing efficient algorithms for solving the equations of motion and their implementation in both a single processor and multiple processor environments.

The significant new material in this paper is the discussion of automatic motion processes for controlling the human figure and the interactive system that we have developed for human figure animation. This interactive animation system allows the animator to take advantage of the power and flexibility of the new dynamic analysis techniques.

Acknowledgements

The work reported here was supported by the Natural Science and Engineering Research Council of Canada and the University of Alberta. We would also like to thank our support staff, in particular S. Sutphen and M. Olafsson, for their assistance.

References

[Armstrong and Green 1985a] Armstrong W.W, M.Green, "The Dynamics of Articulated Rigid Bodies for the Purposes of Animation", Graphics Interface'85, p.407-415, 1985.

[Armstrong and Green 1985b] Armstrong W.W. M.Green, "The Dynamics of Articulated Rigid Bodies for the Purposes of Animation", The Visual Computer, vol.1, no.4, 1985.

[Armstrong et.al. 1986] Armstrong W.W., T.A. Marsland, M. Olafsson, J. Schaeffer, "Solving Equations of Motion on a Virtual Tree Machine", Technical Report, Department of Computing Science, University of Alberta, 1986.

[Hatze 1977] Hatze H., "A Mycocybernetic Control Model of Skeletal Muscle", Biological Cybernetics, vol. 25, p.103-119, 1977.

[McMahon 1984a] McMahon T.A., *Muscles, Reflexes, and Locomotion,* Princeton University Press, Princeton N.J., 1984.

[McMahon 1984b] McMahon T.A., "Mechanics of Locomotion", International Journal of Robotics Research, vol.3, no.2, p.4-28, 1984.

[Plagenhoef 1971] Plagenhoef S., *Patterns of Human Motion: a cinematographic analysis,* Prentice-Hall, Englewood Cliffs NJ, 1971.

[Wilhelms 1985] Wilhelms J.P., *Graphical Simulation of the Motion of Articulated Bodies Such as Humans and Robots, with Particular Emphasis on the Use of Dynamic Analysis,* PhD Thesis, University of California, Berkeley, 1985.

[Wilhelms and Barsky 1985] Wilhelms J., B. Barsky, "Using Dynamic Analysis to Animate Articulated Bodies such as Humans and Robots", Graphics Interface'85, p.97-104, 1985.

[Zeltzer 1982] Zeltzer D., "Motor Control Techniques for Figure Animation", IEEE Computer Graphics and Applications, vol.2, no.9, p.53-59, 1982.