18. Dynamic Simulation as a Tool for Three-Dimensional Animation

Daniel Thalmann
Computer Graphics Lab
Swiss Federal Institute of Technology
Lausanne, Switzerland

18.1. The Movie Approach as opposed to the Physics Approach

There are two ways of considering three-dimensional computer animation (Magnenat Thalmann and Thalmann 1990) and its evolution: the movie approach and the physics approach. The first approach corresponds to an extension of traditional animation methods by the use of the computer. The second approach corresponds to simulation methods based on laws of physics, especially laws of mechanics. The purpose is not the same: traditional methods allow us to create three-dimensional characters with exaggerated movements while simulation methods are used to try to model a human behavior accurately.

For example, consider the bouncing ball example as described by Lasseter (1987). The motion is improved by introducing squash and stretch. When an object is squashed flat and stretches out drastically, it gives the sense that the object is made out of a soft, pliable material. This is a well known trick used by many traditional animators. It does not produce a realistic simulation, but it gives an impression to the viewer. A bouncing ball motion may be also completely simulated by computer using laws of mechanics such as Newton's laws and quantum conservation. Deformations may be calculated by using complex methods like finite element theory (Gourret et al. 1989). No approach is better than the other; it is like comparing a painting and a photograph. Both are representations of a particular world. If we consider character animation, it is easier to create emotions using a keyframe approach than using mechnical laws. Emotional aspects could very well be simulated using a more formal approach, they would require emotion models be incorporated in a physics-based animation system.

In this chapter, we will emphasize the simulation approach and show its advantages in the context of creating simple and complex animations.

18.2. Kinematic Simulations

Animating articulated limbs by interpolating key joint angles corresponds to forward kinematics. For generating goal-directed movements such as moving the hand to grasp an object, it is necessary to compute kinematics. Figure 18.1 shows the principles of forward and inverse kinematics.

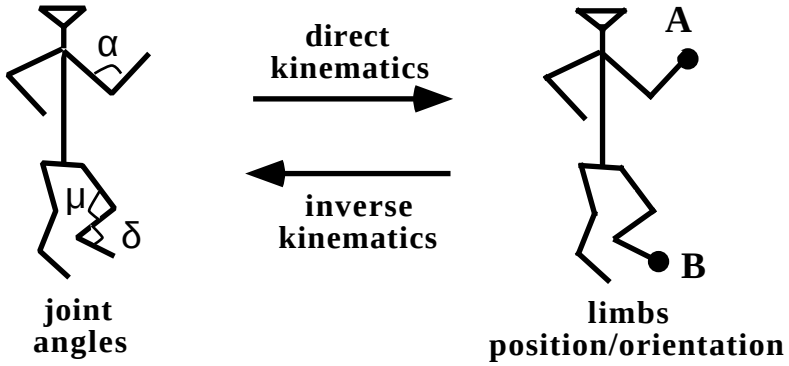

Figure 18.1. Forward and inverse kinematics

18.3. Dynamic Simulations

Kinematic-based systems are generally intuitive and lack dynamic integrity. The animation does not seem to respond to basic physical facts like gravity or inertia. Only modeling of objects that move under the influence of forces and torques can be realistic. For example, the motion in Figure 18.2 (see color section) is difficult to achieve without dynamics). Forces and torques cause linear and angular accelerations. The motion is obtained by the dynamic equations of motion. These equations are established using the forces, the torques, the constraints and the mass properties of objects.

A typical example is the motion of an articulated figure which is governed by forces and torques applied to limbs. These forces and torques may be of various kinds:

- torques coming from parent and child links,
- forces at the hinges,
- external effects such as contact with objects or arm-twisting.

There are three advantages of introducing dynamics into animation control:

- reality of natural phenomena is better rendered,
- dynamics frees the animator from having to describe the motion in terms of the physical properties of the solid objects,

- bodies can react automatically to internal and external environmental constraints: fields, collisions, forces and torques.

There are also serious disadvantages.

Typically, a hierarchy is built first, then internal parameters are set until the desired effect is obtained. This is a severe limitation, because it means that systems are hard for the animator to control. Parameters (e.g. forces or torques) are sometimes very difficult to adjust, because they are not intuitive. For a car, it is easy to choose the parameters of a spring or of a shock absorber, it is more difficult to adjust those used to simulate the equivalent forces and torques produced by muscle contractions and tensions in an animated figure. The animator does not think in terms of forces or torques to apply to a limb or the body in order to perform a motion.

Another problem is the amount of CPU time required to solve the motion equations of a complex articulated body using numerical methods. This considerably reduces the possibility of interaction of the user with the system. Only very short sequences may be produced, because of the lack of complete specification for complex motions and because of the CPU time required for certain methods.

Moreover, although dynamics-based motions are more realistic, they are too regular, because they do not take into account the personality of the characters. It is unrealistic to think that only the physical characteristics of two people carrying out the same actions make these characters different for any observer. Behavior and personality of the human beings are also an essential cause of the observable differences. Several ideas for gracefulness and style in motion control are proposed by Cohen (1989).

Methods based on parameter adjustment are the most popular approach to dynamics-based animation and correspond to *non-constraint methods*. They will be discussed in more detail in Section 18.4. There is an alternative: the *constraint-based methods*: the animator states in terms of constraints the properties the model is supposed to have, without needing to adjust parameters to give it those properties. Four kinds of constraints are discussed in Section 18.5: *kinematic constraints*, *dynamic constraints*, *energy constraints* and *spacetime constraints*.

In dynamic-based simulation, there are also two problems to be considered: the *forward dynamics* problem and the *inverse-dynamics* problem (see Figure 18.3). The forward dynamics problem consists of finding the trajectories of some point (e.g. an end effector in an articulated figure) with regard to the forces and torques that cause the motion. The inverse-dynamics problem is much more useful and may be stated as follows: determine the forces and torques required to produce a prescribed motion in a system. For an articulated figure, it is possible to compute the time sequence of joint torques required to achieve the desired time sequence of positions, velocities and accelerations using various methods.

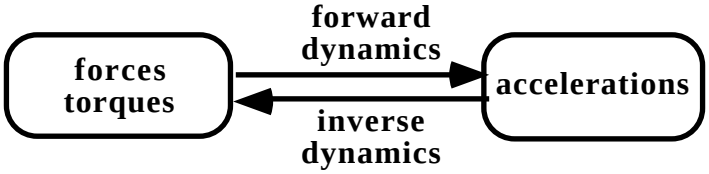

Figure 18.3. Forward and inverse dynamics

18.4. Non-Constraint-Based Methods

Non-constraint methods have been mainly used for the animation of articulated figures. There are a number of equivalent formulations which use various motion equations:

- the Newton–Euler formulation
- the Lagrange formulation
- the Gibbs–Appell formulation
- the D'Alembert formulation

These formulations are popular in robotics and more details about the equations and their use in computer animation may be found in Thalmann (1990)

18.4.1. The Newton–Euler Formulation

The Newton–Euler formulation is based on the laws governing the dynamics of rigid bodies. The procedure in this formulation is to first write the equations which define the angular and linear velocities and accelerations of each link and then write the equations which relate the forces and torques exerted on successive links while under this motion. The vector force $\vec{F}$ is given by the Newton's second law (Equation 18.1) and the total vector torque $\vec{N}$ by Euler's equation (Equation 18.2).

$$\vec{F} = m_i \ddot{\vec{r}}_i \tag{18.1}$$

$$\vec{N} = J_i \dot{\vec{\omega}}_i + \vec{\omega}_i \times (J_i . \vec{\omega}_i) \tag{18.2}$$

Newton–Euler's methods were first developed with respect to the fixed, inertial coordinate system (Stepanenko and Vukobratovicz 1976; Vokobratovicz 1978). Then methods were derived with respect to the local coordinate system.

18.4.2. The Lagrange Formulation

The equations of motion for robots (Magnenat Thalmann and Thalmann 1988) can be derived through the application of the Lagrange's equations of motion for non conservative systems:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = \varphi_i \tag{18.3}$$

where
- L = Lagrangian = T - V
- T = kinetic energy
- V = potential energy
- q_i = generalized joint coordinate representing the internal coordinate of the i-th joint

$\dot{q}_i$ = first time derivative of q_i

φ_i = generalized force applied to the i-th link; this generalized force can be thought of as the force (for sliding joints) or torque (for revolute joints) active at this joint.

Uicker (1965) and Kahn (1969) use the 4x4 rotation/translation matrices introduced by Denavit and Hartenberg (1955) (see Thalmann 1990). Once the kinetic and potential energy is expressed in terms of these matrices and their derivatives, the Lagrange equation is easily applied and the generalized forces φ_i may be found. Unfortunately, the process of evaluation is very time consuming, because it is proportional to the fourth power of the number of links.

Major steps in the Lagrange method are as follows:

(1) select suitable coordinates to define the system,
(2) write the constraint relations between coordinates,
(3) write the expressions corresponding to the kinetic and potential energies,
(4) use Lagrange's equation for defining the differential equations,
(5) get enough equations in order to find all accelerations,
(6) write the expressions of the applied force for each coordinate.

Vasilonikolidakis and Clapworthy (1991a) present the application of inverse Lagrangian dynamics for the purpose of animating articulated bodies. The method is reformulated to calculate the ground reaction forces that apply to the body.

18.4.3. The Gibbs–Appell Formulation

For a body with n degrees of freedom, matrix formulation of the Gibbs–Appell equations are as follows:

$$M^{-1}(\vec{q} - \vec{V}) = \ddot{\vec{c}} \qquad (18.4)$$

$\vec{q}$ is the generalized force at each degree of freedom: M is an n x n inertial matrix and $\vec{V}$ a n-length vector dependent on the configuration of the masses and their velocity relative to each other. If $\vec{q}$, M and $\vec{V}$ are known, the equations can be solved for the generalized accelerations

$\ddot{\vec{c}}$ (n-length vector of the local angular or linear acceleration at each degree of freedom) . This corresponds to the inverse-dynamics problem.

18.4.4. The D'Alembert Formulation

The *D'Alembert's principle of virtual work* states that if a system is in dynamic equilibrium and the bodies are allowed to move a small amount (virtual displacement) then the sum of the

work of applied forces, the work of internal forces will be equal and opposite to the work of changes in momentum.

Isaacs and Cohen (1987) use the D'Alembert formulation in their DYNAMO system (see 18.5.1). In the case study, described in Section 18.8.2, the synthetic actress Marilyn draws letters, the motion calculation is also based on the Principle of Virtual Work.

18.4.5. Matrix and Recursive Formulations

Two dynamic formulations are possible:

(1) A set of simultaneous equations are formed and solved for the accelerations; this is the matrix formulation; it implies a matrix inversion and the formulation typically yields $O(n^3)$ or $O(n^4)$ complexity.

(2) A recursive relationship propagating

18.4.5.1. Matrix Formulations

Wilhelms and Barsky (1985) use the Gibbs–Appel formulation for her animation system Deva; however the method is abandoned (Wilhelms 1990), because the matrices generated are not sparse and the cost of solving for accelerations is prohibively expensive (cost of $O(n^4)$). Isaacs and Cohen (1987) also use a matrix formulation in their method of constraint simulation (see Section 18.5.1).

18.4.5.2. Recursive Formulations

Orin et al. (1979) propose that the forces and the torques be referred to the link's internal coordinate system. Armstrong (1979) and Luh et al. (1980) calculate the angular and linear velocities in link coordinates as well. Armstrong uses coordinate systems located at the joints: Luh et al. employs coordinate systems located at the link centers of mass. Armstrong (1979) presents a method based on the hypothetical existence of a linear recursive relationship between the motion of and forces applied to one link, and the motion of and forces applied to its neighbors. A set of recursion coefficients is defined for each link and the coefficients for one link may be calculated in terms of those of one its neighbors. The accelerations are then derived from the coefficients. The method is only applicable to spherical joints, but its computational complexity is only O(n), where n is the number of links. Based on this theory, Armstrong et al. (1987) designed a near real-time dynamics algorithm and implemented it in a prototype animation system. To reduce the amount of computer time required, Armstrong et al. (1987) make some simplifying assumptions about the structure of the figure and can produce a near real-time dynamics algorithm. They use frames which move with the links and an inertial frame considered fixed and non-rotating. The transformation from one frame to another is done by multiplying a column vector on the left by a 3x3 orthogonal matrix. Armstrong et al. introduce a last equation relating the acceleration at the proximal hinge of a son link s of link i to the linear and angular accelerations at the proximal hinge of i.

Hollerbach (1980) proposes a recursive formulation of the Lagrangian dynamics in three parts:

(1) backward recursion of the velocities and accelerations working from the base of the manipulator to the end link,

(2) forward recursion of the generalized forces working from the end link to the base of the manipulator,

(3) use of 3x3 rotation matrices instead of 4x4 rotation–translation matrices and homogeneous coordinates.

18.5. Constraint-based Methods

18.5.1. Kinematic Constraints

Isaacs and Cohen (1988) discuss a method of constraint simulation based on a matrix formulation. Joints are configures as kinematic constraints, and either accelerations or forces can be specified for the links. They describe a system called DYNAMO that performs dynamic simulation on linked figures. Three means for achieving control have been defined:

1) kinematic constraints permitting traditional keyframe animation systems to be embedded within a dynamic analysis

2) behavior functions allowing the figure to react to its surroundings

3) inverse dynamics for determining the forces required to perform a specified motion.

The dynamic system is treated as an explicit time series analysis with predictor corrector methods maintaining accuracy and efficiency in the solution. Behavior functions determine, at each moment, forces acting on a linkage and/or specific motion which is to occur.

In a new paper (Isaacs and Cohen 1988) about mixed methods for complex kinemtic constraints in dynamic figure animation, they propose an integration of direct and inverse kinematics specifications within a mixed method of forward and inverse dynamics simulation.

18.5.2. Energy Constraints

More generally, an approach to imposing and solving geometric constraints on parameterized models was introduced by Witkin et al. (1987). Constraints are expressed as energy functions, and the energy gradient followed through the model's parameter space.

Witkin et al. propose the following list of geometric constraints which may be cast in the form of energy functions:

- attachment to a fixed point: a point on a surface is attached to a specific point on space.

- surface-to-surface attachment: points on two surfaces must coincide, their tangent planes at those points must also coincide and the surfaces should not interpenetrate.

- floating attachment: a point on an object is attached to some point on a second object; the point of contact may slide freely on the second object
- slider constraint: a point on an object is restricted to lie on a line in space
- collision: collision constraints may be imposed without calculating surface intersections; an implicit function is used; it is zero everywhere on the object's surface, negative inside and positive outside.

It is also possible to impose constraints directly on model parameters.

18.5.3. Dynamic Constraints

Using dynamic constraints, Barzel and Barr (1988) build objects by specifying geometric constraints; the models assemble themselves as the elements move to satisfy the constraints. Their modeling system consists of:

- instantiating primitive bodies (sphere, torii, rods etc.) with geometrical characteristics, density, rotational inertia tensor
- external applied forces: gravity, springs, damping forces given by specific parameters like damping coefficients or spring constants
- geometric constraints of various types:
  - . point-to-nail constraint: a point in the body is fixed to a specified location; the body may swivel or swing about the constrained point.
  - . point-to-point constraint: this is the joint between two bodies which may move about it, but the two constrained points should stay in contact
  - . point-to-path constraint: a point on an object should follow an arbitrary path
  - . orientation constraint: objects are rotated in order to be aligned
  - . point-to-line constraint: a point on an object is restricted to lie on a given line
  - . sphere-to-sphere constraint: two spheres are required to touch, but they may slide along each other

Attachment to a point and slider constraints like described in Section 18.5.2 are also proposed.

Once a model is built, it is held together by constraint forces. Constraint forces are calculated to apply to the bodies such that they behave in accordance with user-specified geometric constraints; the computation of these forces is the typical inverse dynamics problem, which consists of two parts: (1) finding forces to meet a constraint, and (2) finding forces to maintain a constraint.

Platt and Barr (1988) extend dynamic constraints to flexible models. They introduce two new types of constraints:

- reaction constraints: they can force a point to follow a path, or to lie on the outside of a polygonal model. They cancel forces that violate the constraint. Reaction constraints are applicable to constraining elastic models on a point-by-point basis.

- optimization constraints: two types exist:

  1) penalty method where an extra energy that penalizes incorrect behavior is added to the physical system

  2) augmented Lagrange constraints method which is a constrained optimization method that adds differential equetions that compute Lagrange multipliers of the physical system.

18.5.4. Spacetime Constraints

Witkin and Kass (1988) propose a new method, called Spacetime Constraints, for creating character animation. The requirements contained in the description, together with Newton's laws, comprise a problem of constrained optimization. The solution to this problem is a physically valid motion satisfying the constraints.

In this new approach, the character motion is created automatically by specifying:

- *what* the character has to be
- *how* the motion should be performed
- what the character's *physical structure* is
- what physical *resources* are available to the character to accomplish the motion

The problem to solve is a problem of constrained optimization. The solution is a physically valid motion satisfying the *what* constraints and optimizing the *how* criteria. The spacetime constraint approach conforms to principles of traditional animation as anticipation or suqash-and-stretch.

18.6. Collisions and Responses to the Environment

An important advantage to the use of dynamic simulation is the processing of interactions between bodies. The interaction may be first identified then a response may be generated. The most common example of interaction with the environment is the collision. Analytical methods for calculating the forces between colliding rigid bodies have been presented. Moore and Wilhelms (1988) modeled simultaneous collisions as a slightly staggered series of single collisions and used non-analytical methods to deal with bodies in resting contact. Hahn (1988) prevented bodies in resting contact as a series of frequently occuring collisions. Baraff (1989) presented an analytical method for finding forces between contacting polyhedral bodies, based on linear programming techniques. The solution algorithm used is heuristic. A method for finding simultaneous impulsive forces between colliding polyhedral bodies is described. Baraff (1990) also proposed a formulation of the contact forces between curved surfaces that are completely unconstrained in their tangential movement. A collision detection algorithm exploiting the geometric coherence between successive time steps of the simulation is explained. Von Herzen et al. (1990) developed a collision algorithm for time-dependent parametric surfaces.

Terzopoulos et al. (1987) and Platt and Barr (1988) proposed to surround the surfaces of deformable models by a self-repulsive collision force, this is a *penalty* method. Lafleur et al. (1991) addresses the problem of detecting collisions of very flexible objects, such as clothes with almost rigid bodies, such as human bodies. In their method, collision avoidance also consists of creating a very thin force field around the obstacle surface to avoid collisions. This force field acts like a shield rejecting the points. The volume is divided into small contiguous non-overlapped cells which completely surround the surface. As soon as a point enters into a cell, a repulsive force is applied. The direction and the magnitude of this force are dependent on the velocities, the normals and the distance between the point and the surface. More details may be found in Chapter 17.

Hahn (1988) describes the simulation of the dynamic interaction among rigid bodies taking into account various physical characteristics such as elasticity, friction, mass and moment of inertia to produce rolling and sliding contacts. Terzopoulos and Fleischer (1988) propose dynamic models for simulating inelastic behaviors: viscoelasticity, plasticity and fracture.

18.7. Dynamic Simulation of Animal Locomotion and Human Walking

For many years there has been a great interest in natural gait simulation. Zeltzer (1982) describes the use of finite state control to simulate human walking. Girard and Maciejewski (1985) use inverse kinematics to interactively define gaits for legged animals. Boulic et al. (1990) describe a global human walking model with kinematic personification. The model is built from experimental data based on a wide range of normalized velocities. It is based on a simple kinematic approach designed to keep the intrinsic dynamic characteristics of the experimental model. Figure 18.4 shows an example.

Figure 18.4. A walking model

Although Girard's model (Girard 1987) also incorporates some dynamic elements for adding realsim, it is not a truly dynamic approach. Also Bruderlin and Calvert (1989) propose a hybrid approach to the human locomotion which combines goal-oriented and dynamic motion control. Knowledge about a locomotion cycle is incorporated into a hierarchical control process. Decomposition of the locomotion determines forces and torques that drive the dynamic model of the legs by numerical approximation techniques. Rather than relying on a general dynamic model, the equations of motion of the legs are tailored to locomotion and analytically constrained to allow for only a specific range of movements.

McKenna and Zeltzer (1990) describe an efficient forward dynamic simulation algorithm for articulated figures which has a computational complexity linear in the number of joints. The simulation is capable of generating gait patterns and walking phenomena observed in nature. Vasilonikolidakis and Clapworthy (1991b) propose a method based on inverse Lagrangian dynamics for animating human walking. The issue of ground reaction forces is addressed and solved using a reformulation of the Lagrangian algorithm.

Miller (1988) describe a dynamic simulation of the motion of legless figures like snakes and worms. Animals are modeled as mass–spring systems. Muscle contractions are simulated by animating the spring tensions. Miller (1991) also describes an interactive dynamic-based system which controls the dynamic and time-dependent aspects of simulation.

18.8. Case Studies

18.8.1. An Example of a Program of Dynamic Simulation: Dynamik

Dynamik use both direct dynamics and inverse dynamics. The basic simulation algorithm uses direct dynamics, but the constraints (collisions, spatial constraints etc.) are solved by inverse dynamics.

In order to make the simulations more realistic, the concept of friction has been also introduced. Only one kind of friction is taken into account: the friction due to the resistance to rotation at the joints. Friction due to air is ignored.

The Dynamik system is composed of:

(1) an editor of dynamic bodies
(2) a dynamic simulator
(3) a playback system

The purpose of the editor is to build articulated bodies and to specify dynamic data. Data concerning a selected limb may be interactively changed.

- the mass of the limb may be modified
- the mass center is initialized at the geometric center, but it could be changed if necessary
- the limb length and radius may be also edited

There are also specific data for the joints:

- a friction factor may be entered; it is applied to each rotation axis and corresponds to an energy loss
- joint angles may be modified to change the limb orientation
- angle limits for the current joint relative to each rotation axis
- relative and absolute positions

Dynamic controls should be introduced to define motions to be applied to the articulated bodies. These controls consist of constraint activations. Available constraints include: external forces, external torques, "wind" forces, collisions against planes or parallelepipeds, limb connections, positioning in 3-space and rotational motions.

The dynamic simulator computes the complete simulation according to user parameters such as time range simulation, gravity and display options.

The playback system plays, in real-time, the simulation calculated by the dynamic simulator. An example of simulation produced by the system is shown in Figure 18.5 (see color section).

18.8.2. Marilyn Draws Letters

Consider the problem of writing letters on a blackboard. To simulate the writing of a letter, the hand trajectory may be introduced as a kinematic constraint. But how can we control the dynamic behavior of the whole arm and the realism of its motion ? When we write, it is not the hand trajectory which controls the arm motion but the arm muscles which move the hand. A dynamic approach is much more realistic. For this reason, we have produced (Arnaldi et al. 1989) a dynamics-based animation sequence consisting of Marilyn's arm, where the hand reaches a point from a rest position and then successively draws letters O and M from this point. Figure 18.6 (see color section) shows an example.

To produce this sequence, we used a mechanical approach which treats open and closed chains indifferently . It is based on the principle of virtual work (Hégron et al. 1988) associated with Lagrange's multipliers. It can be expressed as letting the Lagrangian parameters of one multibody system (positional and rotational parameters of each object) be submitted to holonomic constraints (joint between two objects) and non holonomic constraints (e.g. wheel which is rolling without sliding on the ground).

By application of the principle of virtual work, computations are performed in a symbolic way to produce the equations of motion. The next step in the animation process is to solve this set of equations numerically for each frame. To do this a simple Newton–Raphson algorithm is used with the jacobian matrix symbolically computed.

Consider again our example of Marilyn: her arm is a four link chain: clavicle, upper arm, lower arm, hand. Each link is modeled as a cylinder to simplify computation of mechanical properties. One type of joint is used to build the arm: ball and socket. To render muscle actions, we add, for each degree of freedom, a spring, a damper and a torque. For realistic mechanical modeling, we can use a non linear spring and a viscous damper to represent passive viscoelastic characteristics of muscles (Winter 1979). Thrusts are added to ensure that motion lies within human capabilities (some degrees of freedom of the ball and socket are restricted, such as in the elbow joint).

Let us now describe the different steps of the arm motion. First, from an initial position, the hand reaches a point on a plane (step 1). Second, from this point the hand draws the letter on the plane (step 2).

As mentioned in the previous sections, it is very difficult to find the different torques to apply to each degree of freedom of the arm to ensure that the hand describes the right trajectory. The inverse problem has to be solved: we know explicitly all the points on the reference trajectory, and, without applying any torque explicitly the hand has to respect this trajectory.

One solution is to create a new joint between a point on the hand and the trajectory. Let X_h be the point on the hand and X_t the point on the trajectory (this point moves with time), the constraint expression f is now:

$$f_t = X_h - X_t \tag{18.5}$$

To solve the motion equation with respect to this constraint, a penalization scheme is used. The joint is computed as a spring between the hand and the trajectory. This is a static solution because at a time step X_t is specified and the constraint f_t is met.

A second solution consists of applying automatic control theory as used in robotics. No specification of an explicit trajectory is now necessary. We will only specify the constraint to be met as:

$$f = X_h - X_g \tag{18.6}$$

where X_g represents the goal.

We require this constraint to be met not immediately, but to evolve through time in accordance with a specified damped oscillation differential equation. The introduction of critical damping is here of special interest in controlling the motion of X_h as shown by Barzel and Barr (1988).

Acknowledgements

The author would like to thank Arghyro Paouri for the design of images. The Dynamik system has been implemented by Rejean Gagné from the University of Montreal. The dynamic Marilyn's hand animation has been performed in cooperation with Gerard Hégron and Bruno Arnaldi from IRISA, University of Rennes. The research was partly supported by "le Fonds National Suisse pour la Recherche Scientifique", Natural Sciences and Engineering Council of Canada and the FCAR foundation.

References

Armstrong WW (1979) Recursive Solution to the Equations of Motion of an N-Link Manipulator, *Proc. 5th World Congress Theory Mach.Mechanisms*, Vol.2, pp.1343-1346

Armstrong WW, Green M, Lake R (1987) Near Real-Time Control of Human Figure Models, *IEEE Computer Graphics and Applications*, Vol. 7, No 6, pp.28-38.

Arnaldi B, Dumont G, Hégron G, Magnenat-Thalmann N, Thalmann D (1989) Animation Control with Dynamics, in: Magnenat-Thalmann N, Thalmann D (eds) *State-of-the-Art in Computer Animation*, Springer, Tokyo, pp.113-124.

Baraff D (1989) Analytical Methods for Dynamic Simulation of Non-Penetrating Rigid Bodies, *Proc. SIGGRAPH '89, Computer Graphics*, Vol. 23, No3, pp.223-232.

Baraff D (1990) Curved Surfaces and Coherence for Non-Penetrating Rigid Body Simulation, *Proc. SIGGRAPH '90, Computer Graphics*, Vol. 24, No4, pp.19-28.

Barzel R, Barr AH (1988) A Modeling System Based on Dynamic Constraints, *Proc. SIGGRAPH '88, Computer Graphics*, Vol. 22, No4, pp.179-188.

Boulic R, Magnenat-Thalmann N, Thalmann D (1990b) A Global Human Walking Model with real time Kinematic Personification, *The Visual Computer*, Vol.6, No6, pp.344-358.

Bruderlin A, Calvert TW (1989) Goal-Directed Dynamic Simulation of Human Walking, *Proc. SIGGRAPH '89, Computer Graphics*, Vol. 23, No3, pp.233-242.

Cohen MF (1989) Gracefulness and Style in Motion Control, *Proc. Mechanics, Control and Animation of Articulated Figures*, MIT, Boston.

Denavit J, Hartenberg RS (1955) A Kinematic Notation for Lower Pair Mechanisms Based on Matrices, *J.Appl.Mech.*, Vol.22, pp.215-221.

Girard M (1987) Interactive Design of 3D Computer-Animated Legged Animal Motion, *IEEE Computer Graphics and Applications*, Vol. 7, No 6, pp.39-51.

Girard M, Maciejewski AA (1985) Computational Modeling for the Computer animation of Legged Figures, *Proc. SIGGRAPH '85, Computer Graphics* , Vol.19, No3 , pp.263-270.

Gourret JP, Magnenat Thalmann N, Thalmann D (1989) Simulation of Object and Human Skin Deformation in Grasping Task, *Proc. SIGGRAPH '89, Computer Graphics*, Vol.23, No.3, pp.21-30.

Hahn JK (1988) Realistic Animation of Rigid Bodies, *Proc. SIGGRAPH '88, Computer Graphics* , Vol. 22, No 4 , pp.299-308.

Hégron G, Arnaldi B, Dumont G (1988) Toward General Animation Control, in: Magnenat-Thalmann N, Thalmann D (eds) *New Trends in Computer Graphics*, Springer, Heidelberg, pp.54-63.

Hollerbach JM (1980) A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamics Formulation Complexity, *IEEE Trans. on Systems*, Man, and Cybernetics, Vol. SMC-10, No11, pp.730-736.

Isaacs PM, Cohen MF (1987) Controlling Dynamic Simulation with Kinematic Constraints, Behavior Functions and Inverse Dynamics, *Proc. SIGGRAPH'87, Computer Graphics*, Vol. 21, No4, pp.215-224.

Isaacs PM, Cohen MF (1988) Mixed Methods for Complex Kinematic Constraints in Dynamic Figure Animation, *The Visual Computer*, Vol. 4, No6, pp.296-305.

Kahn ME (1969) *The Near-Minimum-Time Control of Open-Loop Articulated Kinematic Chains*, Stanford Artificial Intelligence project, AIM-106.

Lafleur B, Magnenat Thalmann N, Thalmann D, Cloth Animation with Self-Collision Detection, in: Kunii TL (ed.) *Modeling in Computer Graphics*, Springer-Verlag, Tokyo

Lasseter J (1987) Principles of Traditional Animation Applied to 3D Computer Animation, *Proc. SIGGRAPH '87, Computer Graphics*, Vol. 21, No4, pp.35-44.

Luh JYS, Walker MW, Paul RPC (1980) On-line Computational Scheme for Mechanical Manipulators, *Journal of Dynamic Systems, Measurement and Control*, Vol.102, pp.103-110.

Magnenat-Thalmann N, Thalmann D (1988) Mechanics and Robotics for Animating Synthetic Actors, *SIGGRAPH '88 Course Notes on Synthetic Actors: The Impact of Artificial Intelligence and Robotics on Animation*, ACM, pp.85-98.

Magnenat-Thalmann N, Thalmann D (1990) *Computer Animation: Theory and Practice*, 2nd edition, Springer-Verlag, Tokyo.

McKenna M, Zeltzer D (1990) Dynamic Simulation of Autonomous Legged Locomotion, *Proc. SIGGRAPH '90, Computer Graphics*, Vol. 24, No 4, pp.29-38.

Miller G (1988) The Motion Dynamics of Snakes and Worms, *Proc. SIGGRAPH '88, Computer Graphics*, Vol. 22, No4, pp.169-173.

Miller G (1991) MacBounce: A Dynamics-Based Modeler for Character Animation, *Proc. Computer Animation '91*, Springer-Verlag, Tokyo.

Moore M, Wihelms J (1988) Collision Detection and Response for Computer Animation, *Proc. SIGGRAPH'88, Computer Graphics*, Vol.22, No.4, pp.289-298.

Orin D, McGhee R, Vukobratovic M, Hartoch G (1979) Kinematic and Kinetic Analysis of Open-Chain Linkages Utilizing Newton-Euler methods, *Mathematical Biosciences*, Vol.31, pp.107-130.

Platt JC, Barr AH (1988) Constraint Method for Flexible Models, *Proc. SIGGRAPH '88, Computer Graphics* , Vol. 22, No4 , pp.279-288.

Stepanenko Y and Vukobratovic M (1976) Dynamics of Articulated Open Chain Active Mechanisms, *Mathematical Biosciences*, Vol.28, pp.137-170.

Thalmann D (1990) Robotics Methods for Task-level and Behavioral Animation, in: Thalmann D (ed) *Scientific Visualization and Graphics Simulation*, John Wiley, Chichester, UK, pp.129-147.

Terzopoulos D, Platt JC, Barr AH, Fleischer K (1987) Elastically Deformable Models, *Proc.SIGGRAPH'87, Computer Graphics*, Vol. 21, No 4, pp.205-214.

Terzopoulos D, Fleischer K (1988) Deformable Models, *The Visual Computer* , Vol.4, No6, pp.306-331.

Uicker JJ (1965) *On the Dynamic Analysis of Spatial Linkages Using 4x4 Matrices*, Ph.D Dissertation, Northwestern University, Evanston, IL.

Vasilonikolidakis N, Clapworthy GJ (1991a) Inverse Lagrangian Dynamics for Animating Articulated Models, *Journal of Visualization and Computer Animation*, Vol.2, No3.

Vasilonikolidakis N, Clapworthy GJ (1991b) Design of Realistic Gaits for the Purpose of Animation, *Proc. Computer Animation '91*, Springer-Verlag, Tokyo.

Von Herzen B, Barr AH, Zatz HR (1990) Geometric Collisions for Time-Dependent Parametric Surfaces, *Proc. SIGGRAPH '90, Computer Graphics*, Vol. 24, No4, pp.39-48

Vukobratovic M (1978) Computer method for Dynamic Model Construction of Active Articulated Mechanisms using Kinetostatic Approach, *Journal of Mechanism and Machine Theory,* Vol.13, pp.19-39.

Wilhelms J (1990) Dynamic Experiences, in: Badler NI, Barsky BA, Zeltzer D (ed) *Making Them Move*, Morgan Kaufmann, San Mateo, CA, pp.265-279.

Wilhelms J, Barsky BA (1985) Using Dynamic Analysis to Animate Articulated Bodies such as Humans and Robots, in: Magnenat-Thalmann N, Thalmann D (eds) *Computer-generated Images*, Springer, Tokyo, pp.209-229.

Winter DA (1979) *Biomechanics of Human Movement*, Wiley Interscience, Chichester, UK.

Witkin A, Fleischer K, Barr AH (1987) Energy Constraints on Parameterized Models, *Proc. SIGGRAPH '87, Computer Graphics*, Vol.21, No4 , pp.225-232.

Witkin A, Kass M (1988) Spacetime Constraints, *Proc. SIGGRAPH '88, Computer Graphics*, Vol.22, No4 , pp.159-168.

Zeltzer (1982) Motor Control Techniques for Figure Animation, *IEEE Computer Graphics and Applications* , Vol. 2, No9 , pp.53-59.