COMPUTER ANIMATION

Nadia Magnenat Thalmann
MIRALab, Centre Universitaire d'Informatique
University of Geneva
24, rue du Général-Dufour
CH-1221 Geneva 4, Switzerland
fax: +41-22-320-2927
E-mail: thalmann@uni2a.unige.ch

Daniel Thalmann
Computer Graphics Lab
Swiss Federal Institute of Technology
CH-1015 Lausanne, Switzerland
fax: +41-21-693-5328
E-mail thalmann@di.epfl.ch

I. Foundations of Computer Animation

A. Introduction

Most of the phenomena which may be represented on the screen of a workstation are typically time-dependent. The techniques of computer graphics allow the construction of 3D graphical objects using geometric modeling techniques. Moreover, in a 3D space, scenes are viewed using virtual cameras and they may be lit by synthetic light sources. In order to visualize these phenomena at any given time, it is necessary to know the appearance of the scene at this time and then Computer Graphics techniques allow us to build and display the scene according to viewing and lighting parameters. The problems to solve are how to express time dependence in the scene, and how to make it evolve over time. These problems and their various solutions are part of what is usually understood by the term **Computer Animation**.

The term "Computer Animation" suggests that computers bring something new to the traditional way of animating. Traditional animation is defined as a technique in which the illusion of movement is created by photographing a series of individual drawings on successive frames of film. Is this definition, due to John Halas [1], still true for Computer Animation ? The definition is essentially correct if we change the definition of the words photographing, drawings, and successive frames. A

definition of computer animation could be: a technique in which the illusion of movement is created by displaying on a screen, or recording on a recording device a series of individual states of a dynamic scene. We use the term "recording" also for photographing and we consider both a cine-camera and a videorecorder as a recording device.

We will see in this chapter that there are two ways of considering computer animation and its evolution. 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. For example, 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 [2]. 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 [3]. 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 mechanical 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.

B. Traditional animation and computer animation

Traditional animated films are produced as a sequence of images recorded frame-by-frame. Computer-animated films may be produced using the exact same process. What is different in this case is the way the frames were produced. In a traditional film, frames are drawings created by artists. In a computer-animated film, frames are produced by the computer based on the artist's directives. As we shall discuss later on, directives could vary from directly drawing each frame on a graphics tablet to just giving orders to three-dimensional characters. The results will be the same: a series of images produced by the computer.

C. Real-time animation and frame-by-frame animation

One interesting question is how long it takes for the computer to produce a frame. The answer is easy; from almost no time at all to an unlimited time depending on the complexity of the images and the computer used.

With powerful graphics workstations, reasonably complex scenes may be rendered in a fraction of second. For example, a typical powerful workstation has the theoretical capabilities to render 60000 polygons in one second. Now consider one second of animation: it needs at least 15 frames to represent a relatively smooth animation, because the illusion of continuous movement breaks down at slower speeds. More typically, animators use 24 frames/sec for films, and 25 or 30 frames/sec for video (depending on the country). For a 30 frames/sec film produced on a workstation, we may theoretically use 2000 (60000/30) polygons per frame. Of course, this is not really possible, because of the time necessary to compute the animation. What is important is that it is possible to see the animation directly on the workstation screen, without recording it. Such real-time animation is more and more popular. In the future, we may expect that workstations will be able to produce more complex images in 1/30 of a second. Very complex animation will be produced in a very short time, due to the research in parallel processing and multiprocessors.

D. 3D Input and virtual reality

For a long time, we could only observe virtual worlds only through the window of the workstation's screen with a very limited interaction possibility. Today, new technologies, called **Virtual Reality**, may immerse us in these computer-generated worlds or at least communicate with them using specific devices. In particular, with the existence of graphics workstations able to display complex scenes containing several thousands of polygons at interactive speed, and with the advent of such new interactive devices as the SpaceBall™, head-mounted displays and DataGlove™, it is possible to create applications based on a full 3-D interaction metaphor in which the specifications of deformations or motion are given in real-time. This new concepts drastically change the way of designing animation sequences. Let us briefly review these new devices; more details may be found in [4 5 6].

First, there are two main ways of recording positions and orientations: magnetic and ultrasonic. Magnetic tracking devices have been the most successful and the Polhemus 3Space Isotrack, although not perfect, is the most common one. A source generates a low frequency magnetic field detected by a sensor. Logitech's **2D/6D mouse** is based on a ultrasonic position reference array, which is a tripod consisting of three ultrasonic speakers set in a triangular position, emits ultrasonic sound signals from each of the three transmitters. These are used to track the receiver position, orientation and movement. In order to address the same problem, Spatial Systems designed a 6 DOF interactive input device called the **SpaceBall™**. This is essentially a “force” sensitive device that relates the forces and torques applied to the ball mounted on top of the device. Hand measurement devices must sense both the flexing angles of the fingers and the position and orientation of the wrist in real-time.

Currently, the most common hand measurement device is the **DataGlove™** from VPL Research..

MIDI keyboards have been first designed for music input, but it provides a more general way of entering multi-dimensional data at the same time. In particular, it is a very good tool for controlling a large number of DOFs in a real-time animation system. A MIDI keyboard controller has 88 keys, any of which can be struck within a fraction of second. Each key transmits velocity of keystroke as well as pressure after the key is pressed.

Binocular vision considerably enhances visual depth perception. Stereo displays like the **StereoView** option on Silicon Graphics workstations may provide high resolution stereo real-time interaction.

Head-mounted display systems present the rich 3-D cues of head-motion parallax and stereopsis. They are designed to take advantage of human binocular vision capabilities and present the general following characteristics:

- headgear with two small LCD color screens, each optically channeled to one eye, for binocular vision.
- special optics in front of the screens, for wide field of view
- a tracking system (e.g. Polhemus 3Space Isotrack) for precise location of the user's head in real time.

We may also mention other devices like force feedback transducers, real-time video input and real-time audio input.

II. Animation methods

A computer animated sequence is obtained by a series of images produced by the computer according to the animator's directives. We may distinguish several general methodologies for creating these sequences: rotoscopy, shape interpolation, parametric keyframe animation and procedural animation.

A. Rotoscopy

This method consists of recording the motion by a specific device for each frame and using this information to generate the image by computer. For example, a human walking motion may be recorded and then applied to a computer-generated 3D character. This approach will provide a very good motion, because it comes directly from reality. However, it does not bring any new concept to animation methodology, and for any new motion, it is necessary to record the reality again.

B. Image-based keyframe animation

Keyframe animation consists of the automatic generation of intermediate frames, called **inbetweens**, based on a set of key-frames supplied by the animator. In image-based keyframe animation or **shape interpolation**, the inbetweens are obtained by interpolating the keyframe images themselves.

In two dimensions, shape interpolation has been popular, especially because of the artist Peter Foldes, who created marvellous films based on this technique: *Hunger* (1974) and *Metadata* (1971). This is an old technique, introduced by Burtnyk and Wein [7]. Each keyframe is characterized by key points which have to correspond. Fig. 1 shows the principles to create inbetween frames by linear interpolation between corresponding points. When corresponding keyframes do not have the same number of key points, it is necessary to add extra-key points as shown in Fig. 2. A linear interpolation algorithm produces undesirable effects such as lack of smoothness in motion, discontinuities in the speed of motion and distortions in rotations, as shown in Fig. 3. Alternate methods have been proposed by Baecker [8], Burtnyk and Wein [9], Reeves [10]. According to Steketee and Badler [11], there is no totally satisfactory solution to the deviations between the interpolated image and the object being modeled.

This method may be extended to three-dimensional objects. The principle is the same when objects are modeled in wire-frame. However, the technique is much more complex when objects are facet-based, because a correspondence between facets and between vertices must be found. Vertices and facets must be added in order to have the same numbers for both objects. A complete algorithm has been introduced by Hong et al. [12].

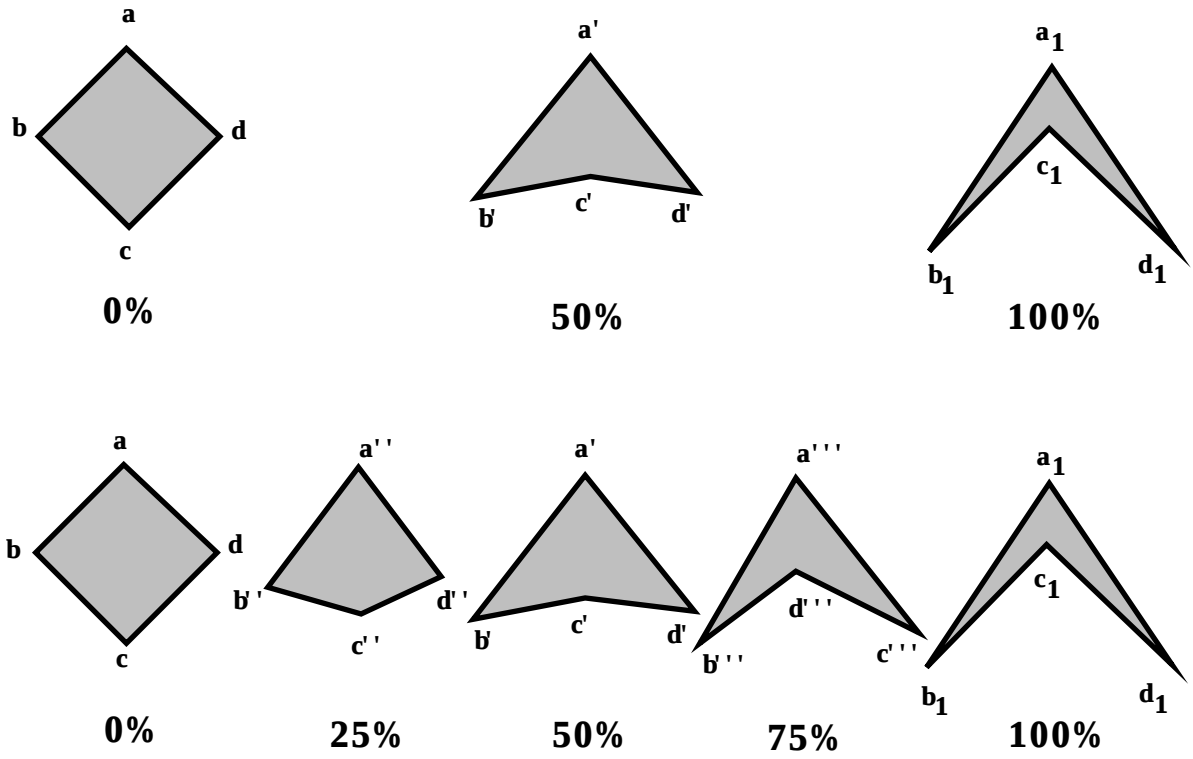

Fig.1. Linear interpolation

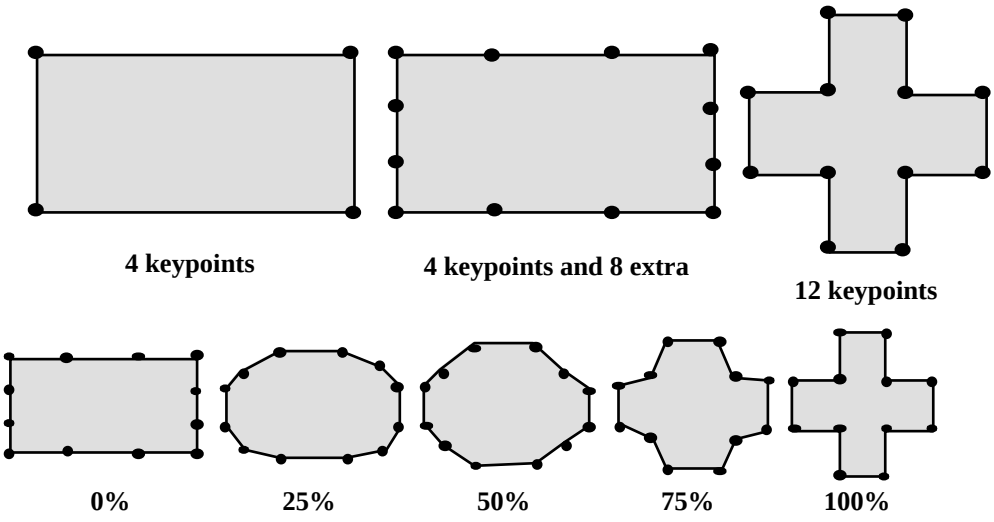

Fig.2. Adding extra vertices before interpolation

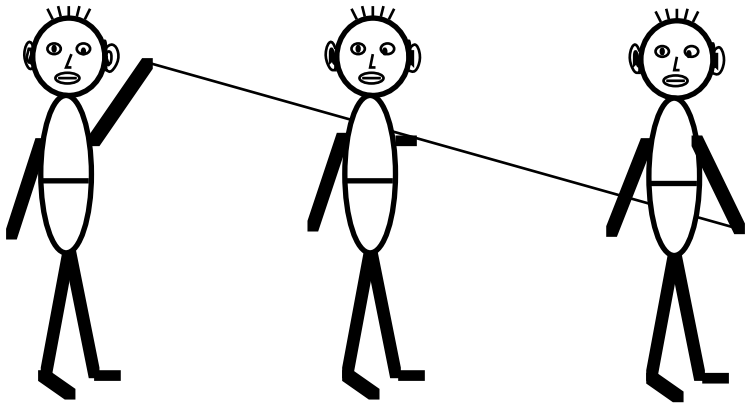

Fig.3. In this example, linear interpolation produces undesirable reduction of the arm length

C. Parametric key-frame animation

Parametric key-frame animation is based on the following principle: an entity (object, camera, light) is characterized by parameters. The animator creates keyframes by specifying the appropriate set of parameter values at a given time, parameters are then interpolated and images are finally individually constructed from the interpolated parameters. Linear interpolation causes first-derivative discontinuities, causing discontinuities in speed and consequently jerky animation. The use of high-level interpolation such as spline interpolation is preferable. For example, the Kochanek-Bartels method [13] of **interpolating splines** is based on piecewise continuous interpolation of the curve by cubic functions. The interpolating curve must be continuous at the given points only up to a certain order. The method allows us to control the curve at each given point by three parameters: tension, continuity and bias.

D. Procedural animation

In **procedural animation**, motion is algorithmically described. Let us have an example to understand the concept and compare it with keyframe animation. We assume an object falling from a height h=100. We would like to generate the animation sequence corresponding to the motion of this object. We know from physics that the equation of motion is $y = h - 0.5g\ t^2$. With an approximate value of the acceleration gravity g, we have the law $y = 100 - 5\ t^2$. It means that after 1 second, the object will be at the height $y_1 = 95$, after 2 sec. at the height y2 = 80, after 3 sec. at the height y3 = 55, after 4 sec. at the height y4 = 20. At the time t=5, the object is already on the floor. A typical keyframe animation consists of giving to the computer the following set: {<0,100>, <1,95>, <2,80>, <3,55>, <4,20>} and to expect that the computer will generate frames for each 1/25 sec. By a linear interpolation, it will be unrealistic. Using a quadratic

interpolation, the result will be exact, because the physics law is a quadratic law. However, it is more direct to use the equation of motion as follows:

```
e.g.   create OBJECT (...);
       TIME = 0;
       while Y > 0 do
             Y = 100 - 5*TIME^2
             MOVE_ABS (OBJECT, X,Y,Z);
             draw OBJECT;
             record OBJECT
             erase OBJECT
             TIME:=TIME+1/25;
```

The advantages of using algorithmic (e.g. physical) laws for animation is that it will give the exact animation. It is in fact a pure simulation. Physical laws are applied to parameters of the animated objects (e.g. positions, angles). Control of these laws may be given by programming as in ASAS [14] and MIRA [15] or using an interactive **director-oriented approach** as in the MIRANIM [16] system. With such an approach, any kind of law [17] may be applied to the parameters. For example, the variation of a joint angle may be controlled by kinematic laws as well as dynamic laws.

Shape interpolation, parametric keyframe animation and procedural animation may be described in a more general and unified way. An animated object is characterized by a set of state variables that drive its motion. The evolution of the state variables is defined by an evolution law. The three types of animation may be redefined using the following terminology as shown in Fig. 4.

TYPE OF ANIMATION	STATE VARIABLES	EVOLUTION LAWS
shape interpolation	vertices	linear interpolation spline interpolation Reeves interpolation
parametric keyframe animation	parameters	linear interpolation spline interpolation
procedural animation	parameters	physical laws

Fig. 4. State variables and evolution laws

E. Choreography in 3D animation

1. Organization of a scene

An animation sequence is composed of one or several related scenes and is characterized by a description, called a **script**. Each scene contains static objects grouped into a decor. Movable objects, called **actors** change over time, by transformations and movements defined by the animator. These transformations are controlled by laws that are applied to variables which

drive the transformations. The transformations may be very simple, like rotations, or very complex, including torsions, flexion, or deformations. Decors and actors are colored and lit by light sources. Finally, decors and actors are viewed using virtual cameras. These cameras may evolve over time as though manipulated by cameramen.

In order to create all the entities and motions, coordinate and synchronize them, known collectively as **choreography**, the director should use an animation system.

2. Camera animation

A scene is only meaningful when it is viewed. But there are many ways a scene may be viewed. It depends on the position of the viewer, where the view is directed, and the viewing angle. Such characteristics, and others, are generally grouped into an entity called a **synthetic** or **virtual camera.** A basic synthetic camera is characterized by at least two parameters: the **eye** and the **interest point**. The eye is a point and it represents the location of the camera; the interest point is the point towards which the camera is directed. A **viewing angle** may also be defined for controlling how wide the observer view is. One of the most impressive effects in computer-generated films is the possibility of rotating around a three-dimensional object or entering inside any complex solid. Although people generally find these effects very spectacular, they are in fact quite easy to produce by animating the camera parameters. The use of several cameras allows the simulation of special effects [18] like fade, wipe, and cross-dissolve effects. Although classical camera motions are often used, there are many situations where it may be more attractive to design a nonlinear trajectory for the camera. Such a trajectory is called **a camera path** and is very often generated using a spline.

We may consider several real-time direct metaphors for controlling camera motion. Ware and Osborne [19] describe three kinematics-based metaphors for moving through environments. Turner et al. [20] describe an approach using the laws of classical mechanics to simulate the virtual camera motion in real time in response to force data from the various 3-D input devices.

III. Animation of Articulated Bodies

A. Skeleton definition

Most animated characters are structured as articulated bodies defined by a skeleton. When the animator specifies the animation sequence, he/she defines the motion using this skeleton. A **skeleton** [21] is a connected set of segments, corresponding to limbs, and joints. A **joint** is the intersection of two segments, which means it is a skeleton point where the limb which is linked to the point may move. The angle between the two segments is

called the **joint angle**. A joint may have at most three kinds of position angles: flexing, pivot and twisting. The **flexing** is a rotation of the limb which is influenced by the joint and cause the motion of all limbs linked to this joint. This flexing is made relatively to the joint point and a flexing axis which has to be defined. The **pivot** makes rotate the flexing axis around the limb which is influenced by the joint. The **twisting** causes a torsion of the limb which is influenced by the joint. The direction of the twisting axis is found similarly to the direction of the pivot. Fig. 5 shows an example of skeleton for a human-like figure.

Name	Number	Angles
VERTEBRA 1	2	FTP
VERTEBRA 2	3	FTP
VERTEBRA 3	4	FTP
VERTEBRA 4	5	FTP
VERTEBRA 5	6	FTP
LEFT CLAVICLE	7	FP
RIGHT CLAVICLE	11	FP
LEFT SHOULDER	8	FTP
RIGHT SHOULDER	12	FTP
LEFT ELBOW	9	FT
RIGHT ELBOW	13	FT
LEFT WRIST	10	FP
RIGHT WRIST	14	FP
LEFT HIP	15	F
RIGHT HIP	20	F
LEFT THIGH	16	FTP
RIGHT THIGH	21	FTP
LEFT KNEE	17	F
RIGHT KNEE	22	F
LEFT ANKLE	18	F
RIGHT ANKLE	23	F
LEFT TOE	19	F
RIGHT TOE	24	F

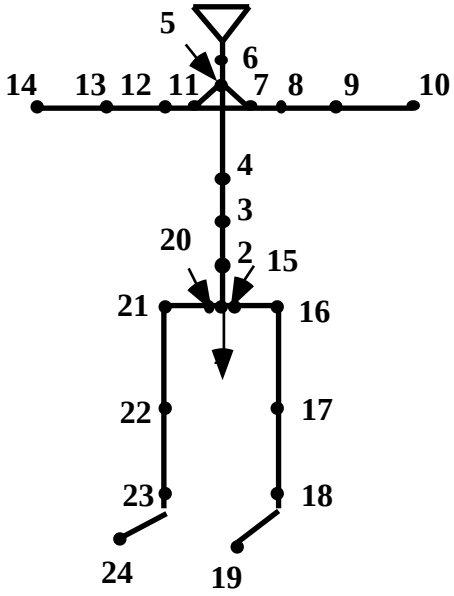

Fig. 5. A basic skeleton with the joints angles (F: flexion, T: twisting, P:pivot)

B. Kinematics methods for skeleton animation

1. Rotoscopy and keyframe animation

Skeleton animation consists of animating joint angles. Among the best-known methods in the category of geometric motion control methods for animating skeletons, we may consider rotoscopy, using sensors to provide coordinates of specific points of joint angles of a real human for each frame. As already mentioned, keyframe systems are typical of systems that manipulate angles; for example, to bend an arm, it is necessary to enter into the computer the elbow angle at different selected times. Then the software is able to find any angle at any time using for example interpolating splines.

2. Forward and inverse kinematics

The **forward kinematics** problem consists in finding the position of end point positions (e.g. hand, foot) with respect to a fixed-reference coordinate system as a function of time without regard to the forces or the moments that cause the motion. Efficient and numerically well-behaved methods exist for the transformation of position and velocity from joint-space (joint angles) to Cartesian coordinates (end of the limb). Parametric keyframe animation is a primitive application of forward kinematics. Animating articulated limbs by interpolating key joint angles corresponds to forward kinematics.

The use of **inverse-kinematics** [22] permits direct specification of end point positions. Joint angles are automatically determined. This is the key problem, because independent variables in a synthetic actor are joint angles. Unfortunately, the transformation of position from Cartesian to joint coordinates generally does not have a closed-form solution. However, there are a number of special arrangements of the joint axes for which closed-form solutions have been suggested in the context of animation [23 24 25 26]. For generating goal-directed movements such as moving the hand to grasp an object, it is necessary to compute inverse kinematics.

Fig. 6 shows the principles of forward and inverse kinematics.

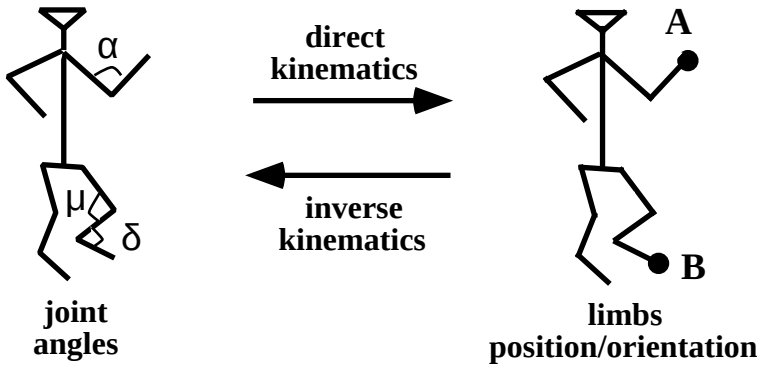

Fig. 6. Forward and inverse kinematics

In a typical system based on inverse kinematics, the animator specifies discrete positions and motions for end parts; then the system computes the necessary joint angles and orientations for other parts of the body to put the specified parts in the desired positions and through the desired motions. Such an approach works well for simple linkages. However, the inverse kinematic solutions to a particular position become numerous and complicated, when the number of linkages increases.

3. Kinematics constraints

A higher level of specification of kinematics motion is based on the use of **constraints**. The animator impose a limb end to stay at a specified location or to follow a predefined trajectory. Badler et al. [27] have introduced an iterative algorithm for solving multiple constraints using inverse kinematics. In their system, the user has to specify also the precedence of each constraint in case they cannot all be simultaneously satisfied.

C. Dynamics

1. 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. 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 [28]:

- 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.

- Parameters (e.g. forces or torques) are sometimes very difficult to adjust, because they are not intuitive.
- The amount of CPU time required to solve the motion equations of a complex articulated body using numerical methods.
- Dynamics-based motions are too regular.

Methods based on parameter adjustment are the most popular approach to dynamics-based animation and correspond to **non-constraint methods** . 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.

In dynamic-based simulation, there are also two problems to be considered: the **forward dynamics** problem and the **inverse-dynamics** problem. 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.

2. 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 [29].

The **Newton-Euler formulation** [30] 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 equations of motion for robots can be derived through the application of the **Lagrange's equations** of motion for nonconservative systems. Based on this theory, Armstrong et al. [31] use a recursive method to design a near real-time dynamics algorithm and implement it in a prototype animation system.

Wilhelms and Barsky [32] use the **Gibbs–Appel formulation** for their animation system Deva; however, the cost of solving for accelerations is prohibively expensive (cost of $O(n^4)$).

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 [33] use the D'Alembert formulation in their DYNAMO system. Also, Arnaldi et al. [28] have produced a dynamics-based animation sequence consisting of an actress' arm, where the hand reaches a point from a rest position and then successively draws letters O and M from this point.

3. Constraint-based Methods

Isaacs and Cohen [34] discuss a method of constraint simulation based on a matrix formulation. Joints are configured as **kinematic constraints**, and either accelerations or forces can be specified for the links. Isaacs and Cohen also propose an integration of direct and inverse kinematics specifications within a mixed method of forward and inverse dynamics simulation. More generally, an approach to imposing and solving geometric constraints on parameterized models was introduced by Witkin et al. [35] using **energy constraints**. Using **dynamic constraints**,

Barzel and Barr [36] build objects by specifying geometric constraints; the models assemble themselves as the elements move to satisfy the constraints. Once a model is built, it is held together by constraint forces. Platt and Barr [37] extend dynamic constraints to flexible models using reaction constraints and optimization constraints. Witkin and Kass [38] propose a new method, called **Spacetime Constraints**, for creating character animation. 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.

D. Surface body animation and deformation

Once the motion of the skeleton is designed, the realism of motion needs to be improved not only from the joint point-of-view, but also in relation to the deformations of bodies during animation. For animating rigid bodies like robots, a simple mapping between the surface and the skeleton is needed. For living beings, the surfaces should be transformed according to the wire-frame model ensuring an automatic continuity between the different surfaces and automatic deformations during the animation process.

The mapping of surfaces onto the skeleton may be based on various techniques. Chadwick et al. [39] propose an approach which constrains the control points of geometric modeling deformations by an underlying articulated robotics skeleton. These deformations are tailored by the animator and act as a muscle layer to provide automatic squash and stretch behaviour of the surface geometry. Komatsu [40] describes the synthesis and the transformation of a new human skin model using the Bézier surfaces. Magnenat-Thalmann and Thalmann [41] introduced the concept of **Joint-dependent Local Deformation** (JLD) operators, which are specific local deformation operators depending on the nature of the joints. These JLD operators control the evolution of surfaces and may be considered as operators on these surfaces. The value of the operator itself is determined as a function of the angular values of the specific set of joints defining the operator. The case of the hand is especially complex [42], as deformations are very important when the fingers are bent, and the shape of the palm is very flexible.

IV. Introduction to Synthetic Actors

A. Introduction

A **synthetic actor** is defined as a human-like autonomous actor completely generated by computer. Applications of synthetic actors are unlimited: in the near future, any human being, dead or alive, may be recreated and placed in any new situation, with no dependence on any live action model. Digital scene simulation will be possible for landscapes with human beings, cinema or theater actors, and spatial vessels with humans; any human behaviour may be simulated in various situations, scientific as

well as artistic. From a user point-of-view, TV announcers may be simulated, or people may walk inside their dream house before the house is built. In the biomedical world, applications are also numerous: deformations of the back and the impact of bad postures, simulation of language dysfunctions and visual dysfunctions. Even in sports education, good positions may be shown as well as the effect of acting muscles. Human beings in dangerous situations may be also simulated: car accidents, airplane crashes, fires, explosions, etc.

The first computerized human models were created 20 years ago by aeroplane and car manufacturers. The main idea was to simulate a very simple articulated structure for studying problems of ergonomics. In the seventies, researchers developed methods to animate human skeletons, mainly based on interpolation techniques. Bodies were represented by very primitive surfaces like cylinders, ellipsoids or spheres. At the same time, the first experimental facial animation sequences appear. The *Juggler* (1982) from the former Information International Inc. (III) was the first realistic human character in computer animation; the results were very impressive; however, the human shape was completely digitized, the body motion had been recorded using 3D rotoscopy and there was no facial animation. In the mid-eighties, researchers started to use the laws of physics. Based on dynamic simulations, it was possible to generate complex motions with a great deal of realism. Even an ordinary human activity like walking is however too complex to be simulated by the laws of dynamics alone. Currently, dynamic simulation is used to portray a few motions of robots or worms as in the film *The Audition* from Apple Computer [43]. Moreover, dynamic simulation always generates the same motions, unrealistic behavior for humans. Two people, even with the same physical characteristics, do not move in the same way. Even one individual does not move in the same way all the time. A behavioral approach to human animation will be eventually necessary to lend a certain credibility to such simulations.

A synthetic actor may be animated. As it is a special case of articulated body, the body motion control has been already discussed in Section III. But another important aspect is the facial animation.

B. Facial animation

The face is a small part of a human, but it plays an essential role in the communication. People look at faces for clues to emotions or even to read lips. It is a particular challenge to imitate these few details. An ultimate objective therefore is to model human facial anatomy exactly including its movements to satisfy both structural and functional aspects of simulation. However, this involves many problems to be solved concurrently. The human face is a very irregular structure, which varies from person to person. The problem is further compounded with its interior details such as muscles, bones and tissues, and the motion which involves complex interactions and deformations of different facial features. Facial animation of synthetic actors is not an easy task, it corresponds to the task of an impersonator. Not only the actors should be realistic in static images, but their motion should be as natural as possible, when a series of images is displayed under the form of a film.

1. The Basic Facial Animation Models

The complexity of facial models leads to what is commonly called facial expressions. The properties of these facial expressions have been studied for 25 years by Psychologist Ekman, who proposed a parameterization of muscles with their relationships to emotions: the **Facial Action Coding System** (FACS) [44]. FACS describes the set of all possible basic actions performable on a human face.

There has been extensive research done on basic facial animation and several models have been proposed. In the early models proposed by Parke [45 46], he has used a combination of digitized expressions and linear interpolation of features such as eyelids and eyebrows and rotations for jaw. The set of facial parameters is based on observation and the underlying structures that cause facial expression. The face is modeled as a collection of polygons manipulated through the set of parameters which may be interpolated.

Platt and Badler [47] have presented a model of a human face and developed a notational system to encode actions performable on a face. The notation drives a model of the underlying muscle structure which in turn determines the facial expression. The model simulates points on the skin, muscle, and bone by a set of interconnected 3D network of points using arcs between selected points to signify relations.

Waters [48] proposes a muscle model which is not specific to facial topology and is more general for modifying the primary facial expression. In this model, muscles are geometric deformation operators which the user places on the face in order to simulate the contraction of the real muscles. Two types of muscles are created linear/parallel muscles that pull and sphincter muscles that squeeze. These muscles are independent of the underlying bone structure, which makes the muscle model independent of specific face topology. The control parameters are based on FACS.

Magnenat Thalmann et al. [49] introduced a way of controlling the human face based on the concept of **abstract muscle action** (AMA) procedures. An AMA procedure is a specialized procedure which simulates the specific action of a face muscle. AMA procedures work on certain regions of the human face which must be defined when the face is constructed. Each AMA procedure is responsible for a facial parameter corresponding approximately to a muscle, for example, vertical jaw, close upper lip, close lower lip, lip raiser etc. A facial expression is considered as a group of facial parameter values obtained by the AMA procedures in different ways.

Terzopoulos and Waters [50] have extended the Waters model, using three layered deformable lattice structures for facial tissues. The three layers correspond to the skin, the subcutaneous fatty tissue, and the muscles. The bottom surface of the muscle layer is attached to the underlying bone. The model uses physically-based technique.

Parke [51] reviews different parameterization mechanism used in different previously proposed models and introduces the future guidelines for ideal control parameterization and interface. Ideal parameterization is in fact a

universal parameterization which would enable all possible individual faces with all possible expressions and expression transitions.

Recently several authors have provided new facial animation techniques based on the information derived from human performances [52 53 54]. The information extracted is used for controlling the facial animation. These performance driven techniques provide a very realistic rendering and motion of the face. Kurihara and Arai [55] introduced a new transformation method for modeling and animating the face using photographs of an individual face. The transformation method enables the movement of points in the skin mesh to be determined by the movement of some selected control points. Texture mapping is used to render the final image.

Kalra et al. [56] propose the simulation of muscle actions based on **Rational Free Form Deformations** (RFFDs). Free form deformation (FFD) is a technique for deforming solid geometric models in a free form manner [57]. It can deform surface primitives of any type or degree, for example, planes, quadrics, parametric surface patches or implicitly defined surfaces. Physically, FFD corresponds to deformations applied to an imaginary parallelepiped of clear, flexible plastic in which are embedded the object(s) to be deformed. The objects are also considered to be flexible so that they are deformed along with the plastic that surrounds them.

2. Speech, Emotion and Synchronization

Most of the facial movements result from either speech or the display of emotions; each of these has its own complexity. However, both speech and emotions need a higher level specification of the controlling parameters. The second level parameterization used in speech animation is usually in terms of phonemes. A phoneme is a particular position of the mouth during a sound emission. These phonemes in turn control the lower level parameters for the actual deformations. Similarly, emotion is a sequence of expressions, and each expression is a particular position of the face at a given time. In order to have a natural manipulation of the speech and emotions there is a need of some synchronization mechanism.

Efforts for **lip synchronization** and to automate the speech initiated with the first study of Lewis and Parke [58]. In their approach, the desired speech is spoken and recorded, the recording is then sampled and analyzed to produce a timed sequence of pauses and phonemes. Hill et al. [59] have introduced an automatic approach to animate speech using speech synthesized by rules. Magnenat-Thalmann et al. [50] have used lip synchronization based on AMA procedures. Let us have an example: For the phoneme "I" (as in "it"), the teeth are slightly open and the commissures are horizontally pulled towards the outside (risorius muscle); this corresponds to 10% of the AMA procedure VERTICAL_JAW, 50% of the AMA procedure LEFT_RISORIUS and 50% of the AMA procedure RIGHT_RISORIUS. At script level, a script in facial animation is a collection of multiple tracks, where each track is a chronological sequence of keyframes for a given facial parameter. On each track, a percentage of a facial parameter or the facial expression may be fixed for a given time. For example, "KID" will be pronounced by a character, indicating that the

phoneme "K" is used at a given time, the phoneme "I" a short time later, then the phoneme "D". Then the software will progressively transform the facial expression corresponding to the phoneme "K" in order to obtain the facial expression corresponding to the phoneme "I", then to the phoneme "D".

In another system, Kalra et al. [60] introduce a **multi-layer approach** where, at each level, the degree of abstraction increases. The high level layers are the most abstract and specify "what to do", the low level layers describe "how to do". This results in a system where complexity is relatively low from the point of view of the animator. The defined entities correspond to intuitive concepts such as phonemes, expressions, words, emotions, sentences and eye motion, which make them natural to manipulate. A manipulation language, HLSS, is provided to ensure synchronization while manipulating these entities.

V. Task-level and behavioral animation

A. Task-level Animation

As stated by Zeltzer [61], a **task-level animation system** must schedule the execution of motor programs to control characters, and the motor program themselves must generate the necessary pose vectors. To do this, a knowledge base of objects and figures in the environment is necessary, containing information about their position, physical attributes, and functionality. With task-level control, the animator can only specify the broad outlines of a particular movement and the animation system fills in the details. Task-level motor control is a problem under study by roboticists.

Similarly to a robot task-level system, actions in a task level animation system [23] are specified only by their effects on objects. According to Lozano-Perez's [62] description, **task planning** may be divided into three phases:

1) **World modelling**: it consists mainly of describing the geometry and the physical characteristics of the objects and the object.
2) **Task specification**: a task specification by a sequence of model states using a set of spatial relationships has been described by Popplestone et al. [63]. In this approach, each state is given by the configuration of all the objects in the environment. The specification by a sequence of commands or a natural language interface is the most suitable and popular [64 65].
3) **Code Generation**: several kinds of output code are possible: series of frames ready to be recorded, value of parameters for certain keyframes, script in an animation language or a command-driven animation system.

In each case, the correspondence between the task specification and the motion to be generated is very complex. In the next sections, we consider two essential tasks for a synthetic actor: walking and grasping.

B. Walking

To generate the motion corresponding to the task "WALK from A to B", it is necessary to take into account the possible obstacles, the nature of the terrain and then evaluate the trajectories which consist of a sequence positions, velocities and accelerations. Given such a trajectory, as well as the forces to be exerted at end effectors, it is possible to determine the torques to be exerted at the joints by inverse dynamics and finally the values of joint angles may be derived for any time. In summary, the task-level system should integrate the following elements: obstacle avoidance, locomotion on rough terrains, trajectory planning, kinematics and dynamics.

For many years there has been a great interest in natural gait simulation. According to Zeltzer [66], the **gait cycle** is usually divided into a stance phase, during which the foot is in contact with the ground, and a swing phase, where the leg is brought forward to begin the stance phase again. Each arm swings forward with the opposite leg and swings back while the opposite leg is in its stance phase. For implementing such a cycle walk, Zeltzer describes a walk controller invoking eight **local motor programs** (LMP): left swing, left stance, right swing, and right stance, which control the actions of the legs, hips, and pelvis; and four other LMPs that control the swinging of the arms.

Girard and Maciejewski [24] use inverse kinematics to interactively define gaits for legged animals. Boulic et al. [67] 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. Such an approach also allows a personification of the walking action in an interactive real-time context in most cases.

Although Girard's model [25] also incorporates some dynamic elements for adding realism, it is not a truly dynamic approach. Also Bruderlin and Calvert [68] 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.

McKenna and Zeltzer [69] describe an efficient forward dynamic simulation algorithm for articulated figures which has a computational complexity linear in the number of joints.

C. Grasping

To generate the motion corresponding to the task "PICK UP the object A and PUT it on the object B", the planner must choose where to grasp A so that no collisions will result when grasping or moving them. Then grasp configurations should be chosen so that the grasped object is stable in the hand (or at least seems to be stable); moreover contact between the hand and the object should be as natural as possible [70]. Once the object is grasped, the system should generate the motions that will achieve the desired goal of the operation. A free motion should be synthesized; during this motion the principal goal is to reach the destination without collision, which implies obstacle avoidance. In this complex process, joint evolution

is determined by kinematics and dynamics equations. In summary, the task-level system should integrate the following elements: path planning, obstacle avoidance, stability and contact determination, kinematics and dynamics.

D. Impact of the environment

1. Introduction

Synthetic actors are moving in an **environment** comprising models of physical objects. Their animation is dependent on this environment and the environment may be modified by these actors. Moreover several synthetic actors may interact with each other. Several very complex problems must be solved in order to render three-dimensional animation involving actors in their environment. They may be classified into the following categories:

- reaching or avoiding obstacles
- contacts and collisions between rigid objects
- contacts and deformations of deformable objects
- group behaviour (this problem will be discussed in Section 4).

2. Obstacle avoidance

Consider, for example, the problem of walking without collision among obstacles. One strategy is based on the Lozano-Perez algorithm [71]. The first step consists of forming a **visibility graph**. Vertices of this graph are composed of the vertices of the obstacles, the start point S and the goal point G. Edges are included if a straight line can be drawn joining the vertices without intersecting any obstacle. The shortest collision-free path from S to G is the shortest path in the graph from S to G. Lozano-Perez and Wesley describe a way of extending this method to moving objects which are not points. Schröder and Zeltzer [72] introduced Lozano-Perez algorithm into their interactive animation package BOLIO. Breen [73] proposes a technique employing cost functions to avoid obstacles. These functions are used to define goal-oriented motions and actions and can be defined so that the variables are the animated parameters of a scene. These parameters are modified in such a way to minimize the cost function.

3. Contacts and collisions of rigid objects

The reaction of an actor to the environment may also be considered using dynamic simulation in the processing of interactions between bodies. The interaction is first identified and then a response is 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 [74] modelled simultaneous collisions as a slightly staggered series of single collisions and used non-analytical methods to deal with bodies in resting contact. Hahn [75] prevented bodies in resting contact as a series of frequently occuring collisions. Baraff [76] 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 also described. Baraff [77] 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. [78] developed a collision algorithm for time-dependent parametric surfaces. Hahn [77] 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. Gourret et al. [3] develop a finite element method for simulating deformations of objects and the hand of a synthetic character during a grasping task. When a deformable object is grasped, the contact forces that act on it and on the fingertips will lead both to deformation of the object and of the fingertips, giving reacting forces which provide significant information about the object and more generally about the environment of the synthetic human body.

4. Deformable and flexible objects

Terzopoulos and Fleischer [79] developed **deformable models** capable of perfectly elastic and inelastic behaviour, viscoelasticity, plasticity, and fracture. The models recently developed by Terzopoulos et al. [80] are for example implemented using the Finite Difference Method, and collisions between elastic objects are simulated by creating potential energy around each object, i.e. intersections between deformable bodies are avoided by surrounding the object surfaces with a repulsive collision force. This is a **penalty method**.

A specific, but important case is cloth animation. In recent years **Cloth Animation** has become an important subject in computer animation, and many efforts have been made in this field. Cloth animation includes not only the modelling of garment on the human body, but also such things as flags, curtains, tablecloths and stage screens. A geometric approach is suitable for representing single pieces of the objects or clothes with simple shapes, which are easily computed, but geometric flexible models like Weil's model (a best looking model that can create realistic folds) [81] or Hinds and McCartney's model [82] have not incorporated concepts of quantities varying with time, and are weak in representing physical properties of cloth such as elasticity, anisotropy, and viscoelasticity. Only physical models like Terzopoulos' model [82] and Aono's model [83] may correctly simulate these properties. Another interesting approach by Kunii and Gotoda [84] incorporates both the kinetic and geometric properties for generating garment wrinkles.

Lafleur et al. [85] 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 recently, Carignan et al. [86] discuss the use of physics-based models for animating clothes on synthetic actors in motion. In their approach, cloth pieces are first designed with polygonal panels in two dimensions, and are then seamed and attached to the actor's body in three dimensions. They describe a new approach to the problem of handling collisions among the cloth elements themselves, or between a cloth element and a rigid object like the human body.

E. Behavioral animation

Motion of 3D characters is not simply a matter of mechanics: you cannot walk exactly the same way from the same bar to home twice. Mechanics-based motions 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. Behaviour and personality of the human beings are also an essential cause of the observable differences. Behavioral animation corresponds to modeling the behavior of characters, from path planning to complex emotional interactions between characters. In an ideal implementation of a behavioral animation, it is almost impossible to exactly play the same scene twice. For example, in the task of walking, everybody walks more or less the same way, following more or less the same laws. This is the "more or less" which will be difficult to model. And also a person does not walk always the same way everyday. If the person is tired, or happy, or just got some good news, the way of walking will appear slightly different. So in the future, another big challenge is open for the computer animation field: to model human behavior taking into account social differences and individualities.

Reynolds [87] studied in details the problem of group trajectories: bird flocks, herds of land animals and fish schools. This kind of animation using a traditional approach (keyframe or procedural laws) is almost impossible. In the Reynolds approach, each bird of the flock decide itself its trajectory without animator intervention. Reynolds introduces a distributed behavioural model to simulate flocks of birds, herds of land animals, and schools of fish. The simulated flock is an elaboration of a particle system with the simulated birds being the particles. A flock is assumed to be the result of the interaction between the behaviours of individual birds. Working independently, the birds try both to stick together and avoid collisions with one another and with other objects in their environment. The animator provides data about the leader trajectory and the behaviour of other birds relatively to the leader (e.g. minimum distance between actors). A computer-generated film has been produced by symbolic using this distributed behavioural model: *Breaking the ice.*

Haumann and Parent [88] describe behavioural simulation as a means to obtain global motion by simulating simple rules of behaviour between locally related actors. Lethebridge and Ware [89] propose a simple heuristically-based method for expressive stimulus-response animation. They model stimulus-response relationships using "behaviour functions"

which are created from simple mathematical primitives in a largely heuristic manner.

For solving the problem of a synthetic actor crossing a room with furniture (table, chairs etc.), the use of an algorithm like the Lozano-Perez algorithm will certainly provide a trajectory avoiding the obstacle. But this trajectory won't be "natural". No human would follow such a path! The decision of where to pass is based on our vision and we require a certain additional room for comfort. We try to keep a "security distance" from any obstacle. This is a typical behavioral problem that cannot be solved by graph theory or mathematical functions. Moreover, walking depends on our knowledge of the location of obstacles and it is only when we see them that we start to include them in our calculations for adapting the velocity. Renault et al. [90] propose a way of giving to the synthetic actor a vision of his environment. The synthetic environment chosen for these trials is a corridor containing several obstacles of various sizes. The synthetic actor may be placed at any location of the corridor and with any look direction; he will move along the corridor avoiding the obstacles. The system is able to avoid collisions with movable objects, what is not possible with well-known robotics algorithms of path-planning. The model is based on the concept of Displacement Local Automata (DLA), which is an algorithm that can deal with a specific environment. Two typical DLAs are called *follow-the-corridor* and *avoid-the-obstacle*. Vision simulation is the heart of this system. This has the advantage of avoiding all the problems of pattern recognition involved in robotic vision. As input, we have a database containing the description of 3D objects: the environment, the camera characterized by its eye and interest point. As output, the view consists of a 2D array of pixels. each pixel contains the distance between the eye and the point of the object for which this is the projection.

Behavioral communication systems should be the most important in the future, simply because this is the most common way of communicating between real people. In fact, our behavior is generally based on the response to stimuli [91] from other people. In particular, we participate in linguistic and emotional exchange with other people. Thus the facial expressions of an actor may be in response to the facial expressions of another actor. For example, the actress Marilyn may smile just because the actor Bogey smiles.

VI. References

1 Halas J, Manwell R (1968) The Technique of Film Animation, Hastings House, New York

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

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

[4] Balaguer F, Mangili A (1991) Virtual Environments in: N.Magnenat-Thalmann and D.Thalmann (Eds) New Trends in Animation and Visualization, John Wiley and Sons, pp.91-106.

[5] Brooks F.P. Jr (1986) Walkthrough - A Dynamic Graphics System for Simulating Virtual Buildings Proceedings 1986 Workshop on Interactive 3-D Graphics ACM, pp.9-22

[6] Fisher S.S., McGreevy M., Humphries J., Robinett W.,(1986), "Virtual Environment Display System", Proceeding 1986 Workshop on Interactive 3-D Graphics, ACM, pp 77-87

[7] Burtnyk N, Wein M (1971) Computer-generated Key-frame Animation, Journal SMPTE, 80, pp.149-153.

[8] Baecker R (1969) Picture-driven Animation, Proc. AFIPS Spring Joint Comp. Conf., Vol.34, pp.273-288

[9] Burtnyk N, Wein M (1976) Interactive Skeleton Techniques for Enhancing Motion Dynamics in Key Frame Animation, Comm. ACM, Vol.19, No10, pp.564-569.

[10] Reeves WT (1981) Inbetweening for computer animation utilizing moving point constraints. Proc. SIGGRAPH '81, Computer Graphics, Vol.15, No3, pp.263-269

[11] Steketee SN, Badler NI (1985) Parametric Keyframe Interpolation Incorporating Kinetic Adjustment and Phrasing Control, Proc. SIGGRAPH '85, pp. 255-262.

[12] Hong T.M., R.Laperrière, D.Thalmann, A General Algorithm for 3-D Shape Interpolation in a Facet-Based Representation, Proc. Graphics Interface'88, Edmonton, 1988

[13] Kochanek D and Bartels R (1984) Interpolating Splines with Local Tension, Continuity and Bias Tension, Proc. SIGGRAPH '84, Computer Graphics, Vol.18, No3, pp.33-41.

[14] Reynolds CW (1982) Computer Animation with Scripts and Actors, Proc. SIGGRAPH'82, Computer Graphics, Vol.16, No3, pp.289-296.

[15] Magnenat-Thalmann N, Thalmann D (1983) The Use of High Level Graphical Types in the MIRA Animation System, IEEE Computer Graphics and Applications, Vol. 3, No 9, pp. 9-16.

[16] Magnenat-Thalmann N, Thalmann D, Fortin M (1985) MIRANIM: An Extensible Director-Oriented System for the Animation of Realistic Images, IEEE Computer Graphics and Applications, Vol.5, No 3, pp. 61-73.

[17] Magnenat-Thalmann N, Thalmann D (1985) 3D Computer Animation: More an Evolution Problem than a Motion Problem, IEEE Computer Graphics and Applications Vol. 5, No10, pp. 47-57

[18] Magnenat-Thalmann N, Thalmann D (1986) Special cinematographic effects using multiple virtual movie cameras, IEEE Computer Graphics and Applications 6(4):43-50

[19] Ware C, Osborne S, “Exploration and Virtual Camera Control in Virtual Three Dimensional Environments”, Computer Graphics, 24 (2), pp. 175-183.

[20] Turner R, Balaguer F, Gobbetti E, Thalmann D (1991), Physically-Based Interactive Camera Motion Control Using 3-D Input Devices, in: Patrikalakis N (ed.) Scientific Visualization of Physical Phenomena, Springer, Tokyo, pp.135-145.

[21] Badler NI and Smoliar SW (1979) Digital Representation of Human Movement, ACM Computing Surveys, March issue, pp.19-38.

[22] Hollerbach JM, Sahar G (1983) Wrist-Partitioned, Inverse Kinematic Accelerations and Manipulator Dynamics, Intern. Journal of Robotics Research, Vol.2, No4, pp.61-76.

[23] Badler NI, Korein JD, Korein JU, Radack GM and Brotman LS (1985) Positioning and Animating Figures in a Task-oriented Environment, The Visual Computer, Vol.1, No4, pp.212-220.

[24] Girard M and Maciejewski AA (1985) Computational Modeling for Computer Generation of Legged Figures, Proc. SIGGRAPH '85, Computer Graphics, Vol. 19, No3, pp.263-270

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

[26] Korein J, Badler NI (1982) Techniques for Generating the Goal-directed Motion of Articulated Structures, IEEE Computer Graphics and Applications, Vol.2, No9, pp.71-81

[27] Badler NI et al. (1986) Multi-Dimensional Input Techniques and Articulated Figure Positioning by Multiple Constraints, 1986 Workshop on Interactive 3D Graphics, Chapel Hill, North Carolina

[28] 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.

[29] 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.

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

[31] Armstrong WW, Green M and Lake R (1987) Near real-time Control of Human Figure Models, IEEE Computer Graphics and Applications, Vol.7, No 6, pp.28-38

[32] Wilhelms J and Barsky B (1985) Using Dynamic Analysis to Animate Articulated Bodies such as Humans and Robots, in: N.Magnenat-Thalmann and D.Thalmann (Eds) Computer-generated Images, Springer, pp.209-229.

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

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

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

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

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

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

[39] Chadwick J, Haumann DR, Parent RE (1989) Layered Construction for Deformable Animated Characters, Proc. SIGGRAPH '89, Computer Graphics, Vol. 23, No3, pp.234-243

[40] Komatsu K (1988) Human Skin Model Capable of Natural Shape Variation, The Visual Computer, Vol.3, No5, pp.265-271

[41] Magnenat-Thalmann N, Thalmann D (1987), The Direction of Synthetic Actors in the film Rendez-vous à Montréal, IEEE Computer Graphics and Applications, Vol. 7, No. 12, pp. 9-19.

[42] Magnenat-Thalmann N, Laperrière R and Thalmann D (1988) Joint-dependent Local Deformations for Hand Animation and Object Grasping, Proc. Graphics Interface '88

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

[44] Ekman P and Friesen W (1978) Facial Action Coding System, Consulting Psychologists Press, Palo Alto.

[45] Parke FI (1975) A Model for Human Faces that allows Speech Synchronized Animation, Computers and Graphics, pergamon Press, Vol.1, No1, pp.1-4.

[46] Parke FI (1982) Parameterized Models for Facial Animation, IEEE Computer Graphics and Applications, Vol.2, No 9, pp.61-68

[47] Platt S, Badler N (1981) Animating Facial Expressions, Proc. SIGGRAPH '81, Computer Graphics, Vol.15, No3, pp.245-252.

[48] Waters K (1987) A Muscle Model for Animating Three-Dimensional Facial Expression, Proc. SIGGRAPH '87, Computer Graphics, Vol.21, No4, pp.17-24.

[49] Magnenat-Thalmann N, Primeau E, Thalmann D (1988), Abstract Muscle Action Procedures for Human Face Animation, The Visual Computer, Vol. 3, No. 5, pp. 290-297.

[50] Terzopoulos D, Waters K (1990) Physically Based Facial Modeling, Analysis, and Animation, Visualization and Computer Animation, Vol. 1, No. 2, pp. 73-80.

[51] Parke FI (1991), Control Parameterization for Facial Animation, Proc. Computer Animation '91, Geneva, Switzerland, (Eds Magnenat-Thalmann N and Thalmann D), pp. 3-13.

[52] deGraf B (1989) in State of the Art in Facial Animation, SIGGRAPH '89 Course Notes No. 26, pp. 10-20.

[53] Williams L (1990), Performance Driven Facial Animation, Proc SIGGRAPH '90, pp. 235-242.

[54] Terzopoulos D, Waters K (1991) Techniques for Facial Modeling and Animation, in: Magnenat Thalmann N, Thalmann D (eds) Computer Animation '91, Springer, Tokyo, pp.59-74

[55] Kurihara T, Arai K (1991), A Transformation Method for Modeling and Animation of the Human Face from Photographs, Proc. Computer Animation '91 Geneva, Switzerland, (Eds Magnenat-Thalmann N and Thalmann D), pp. 45-57.

[56] Kalra P, Mangili A, Magnenat-Thalmann N, Thalmann D (1992) Simulation of Facial Muscle Actions Based on Rational Free Form Deformations, Proc. Eurographics '92, Cambridge, UK.

[57] Sederberg TW, Parry SR (1986) Free-form Deformation of Solid Geometric Models, Proc.SIGGRAPH'86, Computer Graphics, Vol.20, No4, pp.151-160.

[58] Lewis JP, Parke FI (1987) Automated Lip-synch and Speech Synthesis for Character Animation, Proc. CHI '87 and Graphics Interface '87, Toronto, pp.143-147.

[59] Hill DR, Pearce A, Wyvill B (1988), Animating Speech: An Automated Approach Using Speech Synthesised by Rules, The Visual Computer, Vol. 3, No. 5, pp. 277-289.

[60] Kalra P, Mangili A, Magnenat-Thalmann N, Thalmann D (1991) SMILE: a Multilayered Facial Animation System, Proc. IFIP Conference on Modelling in Computer Graphics, Springer, Tokyo, Japan, pp.189-198.

[61] Zeltzer D (1985) Towards an Integrated View of 3D Computer Animation, The Visual Computer, Vol.1, No4, pp.249-259.

[62] Lozano-Perez (1982) Task Planning in: Brady M (Ed.) Robot Motion: Planning and Control, MIT Press, Cambridge, Mass.

[63] Popplestone RJ, Ambler AP and Bellos IM (1980) An Interpreter for a Language for Describing Assemblies, Artificial Intelligence, Vol.14, pp.79-107

[64] Drewery K, Tsotsos J (1986) Goal Directed Animation using English Motion Commands, Proc. Graphics Interface '86, pp.131-135

[65] Badler NI (1989) Artificial Intelligence, Natural Language, and Simulation for Human Animation, in: Magnenat-Thalmann N, Thalmann D (Eds) State-of.the-Art in Computer Animation, Springer, Tokyo, pp. 19-32

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

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

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

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

[70] Rijpkema H, Girard M (1991) Computer Animation of Knowledge-based Human Grasping, Proc. Siggraph '91, Computer Graphics, Vol. 25, No3, pp.339-348

[71] Lozano-Perez T, Wesley MA (1979) An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles, Comm.ACM, Vol.22, No10, pp. 560-570.

[72] Schröder P, Zeltzer D (1988) Pathplanning inside Bolio, in: Synthetic Actors: The Impact of Artificial Intelligence and Robotics on Animation, Course Notes SIGGRAPH '88, ACM, pp.194-207.

[73] Breen DE (1989) Choreographing Goal-Oriented Motion Using Cost Functions, in: Magnenat-Thalmann N, Thalmann D, State-of.the-Art in Computer Animation, Springer, Tokyo, pp. 141-152

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

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

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

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

[78] Von Herzen B, Barr AH, Zatz HR (1990) Geometric Collisions for Time-Dependent Parametric Surfaces, Proc. SIGGRAPH'90, Computer Graphics, Vol.24, No.4, pp.39-46

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

[80] Terzopoulos D, Platt J, Barr A, Fleischer K (1987) Elastically Deformable Models, Proc. SIGGRAPH'87, Computer Graphics, Vol. 21, No.4, pp.205-214

[81] Weil J (1986) The Synthesis of Cloth Objects, Proc. SIGGRAPH '86, Computer Graphics, Vol.20, No.4, pp.49-54

[82] Hinds BK, McCartney J, Interactive garment design, The Visual Computer (1990) 6, pp.53-61

[83] Aono M (1990) A Wrinkle propagation Model for Cloth, Proc. Computer Graphics International '90, Springer-verlag, Tokyo, pp.96-115

[84] Kunii TL, Gotoda H (1990) Modeling and Animation of Garment Wrinkle Formation Processes, Proc. Computer Animation'90, Springer, Tokyo, pp.131-147

[85] Lafleur B, Magnenat Thalmann N, Thalmann D, Cloth Animation with Self-Collision Detection, Proc. IFIP Conf. on Modeling in Computer Graphics, Tokyo

[86] Carignan M, Yang Y, Magnenat Thalmann N, Thalmann D (1992) Dressing Animated Synthetic Actors with Complex Deformable Clothes, Proc. SIGGRAPH '92.

[87] Reynolds C (1987) Flocks, Herds, and Schools: A Distributed Behavioral Model, Proc.SIGGRAPH '87, Computer Graphics, Vol.21, No4, pp.25-34

[88] Haumann DR, Parent RE (1988) The Behavioral Test-bed: Obtaining Complex Behavior from Simple Rules, The Visual Computer, Vol.4, No 6, pp.332-347

[89] Lethebridge TC and Ware C (1989) A Simple Heuristically-based Method for Expressive Stimulus-response Animation, Computers and Graphics, Vol.13, No3, pp.297-303

[90] Renault O, Magnenat-Thalmann N, Thalmann D (1990) A Vision-Based Approach to Behavioural Animation, Visualization and Computer Animation Journal, John Wiley, Vol,1, No1 (July 1990)

[91] Wilhelms J (1990) A "Notion" for Interactive Behavioral Animation Control, IEEE Computer Graphics and Applications , Vol. 10, No 3 , pp.14-22

Bibliography

The following book covers all aspects of Computer Animation:

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

The two following books are dedicated to animation of articulated bodies and synthetic actors:

Badler NI, Barsky B, Zeltzer D (eds) (1990) Making Them Move, Morgan Kaufmann, San Mateo, California

Magnenat-Thalmann N, Thalmann D (1990) Synthetic Actors in Computer-Generated Films, Springer-Verlag, Heidelberg

The two following publications are dedicated to research in Computer Animation:

1. The Journal of Visualization and Computer Animation, John Wiley and Sons
2. Proceedings of Computer Animation Conferences. published each year by Springer-Verlag Tokyo since 1989.

Papers about Computer Animation may be often found in the following publications:

1. The Visual Computer, Springer-Verlag, Heidelberg
2. Proceedings SIGGRAPH, ACM
3. IEEE Computer Graphics and Applications