# How to Create a Virtual Life ?

**Daniel Thalmann, Hansrudi Noser, Zhiyong Huang**
*Computer Graphics Lab, Swiss Federal Institute of Technology*

**Abstract**

Virtual Life is a new area dedicated to the simulation of life in virtual worlds, human-virtual interaction and immersion inside virtual worlds. Virtual Life cannot exist without the growing development of Computer Animation techniques and corresponds to the most advanced concepts and techniques of it. In this chapter, we present current research developments in the Virtual Life of autonomous synthetic actors. After a brief description of the perception action principles with a few simple examples, we emphasize the concept of virtual sensors for virtual humans. In particular, we describe in details our experiences in implementing virtual vision, tactile and audition. We then describe perception-based locomotion, a multisensor based method of automatic grasping and vision-based ball games. We also discuss problems of integrating autonomous humans into virtual environments.

## 1 Introduction

As a virtual world is completely generated by computer, it expresses itself visually, with sounds and feelings. Virtual worlds deal with all the models describing physical laws of the real world as well as the physical, biological, and psychological laws of life. Virtual Life is linked to problems in artificial life but differs in the sense that these problems are specific to virtual worlds. It does not deal with physical or biological objects in real life but only with the simulation of biological virtual creatures. Virtual Life is at the intersection of Virtual Reality and Artificial Life (Magnenat Thalmann and Thalmann 1994), it is an interdisciplinary area strongly based on concepts of real-time computer animation, autonomous agents, and mobile robotics. Virtual Life cannot exist without the growing development of Computer Animation techniques and corresponds to the most advanced concepts and techniques of it.

In this chapter, we first review the state-of-the-art in this emerging field of Virtual Life, then we emphasize the aspect of Virtual Life of Synthetic Actors, relating the topics to the previous chapters. The objective is to provide autonomous virtual humans with the skills necessary to perform stand-alone role in films, games (Bates et al. 1992)

and interactive television (Magnenat Thalmann and Thalmann 1995). By autonomous we mean that the actor does not require the continual intervention of a user. Our autonomous actors should react to their environment and make decisions based on perception, memory and reasoning. With such an approach, we should be able to create simulations of situations such as virtual humans moving in a complex environment they may know and recognize, or playing ball games based on their visual and tactile perception. We also describe techniques to interact with these virtual people in Virtual Environments.

## 1.1 State-of-the-Art in Virtual Life

This kind of research is strongly related to the research efforts in behavioral animation as introduced by Reynolds (1987) to study the problem of group trajectories: flocks of birds, 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 decides its own trajectory without animator intervention. Reynolds introduces a distributed behavioral model to simulate flocks. 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 behaviors 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. In a module of behavioral animation, positions, velocities and orientations of the actors are known from the system at any time. The animator may control several global parameters: e.g. weight of the obstacle avoidance component, weight of the convergence to the goal, weight of the centering of the group, maximum velocity, maximum acceleration, minimum distance between actors. The animator provides data about the leader trajectory and the behavior of other birds relatively to the leader. A computer-generated film has been produced using this distributed behavioral model: *Stanley and Stella*.

Haumann and Parent (1988) describe behavioral simulation as a means to obtain global motion by simulating simple rules of behavior between locally related actors. Lethebridge and Ware (1989) propose a simple heuristically-based method for expressive stimulus-response animation. Wilhelms (1990) proposes a system based on a network of sensors and effectors. Ridsdale (1990) proposes a method that guides lower-level motor skills from a connectionist model of skill memory, implemented as collections of trained neural networks.

We should also mention the huge litterature about autonomous agents (Maes 1991) which represents a background theory for behavioral animation. More recently, genetic algorithms were also proposed by Sims (1994) to automatically generate morphologies for artificial creatures and the neural systems for controlling their muscle forces. Tu and Terzopoulos (1994) described a world inhabited by artificial fishes

## 1.2 Principle of Behavioral Animation

A simulation is produced in a synchronous way by a behavioral loop such as:

```
t_global = 0.0
        code to initialize the animation environment
while (t_global < t_final) {
        code to update the scene
        for each actor
            code to realize the perception of  the environment
            code to select actions based on sensorial input, actual state and specific
            behavior
        for each actor
        code executing the above selected actions
        t_global += t_interval}
```

The global time t_global serves as synchronization parameter for the different actions and events. Each iteration represents a small time step. The action to be performed by each actor is selected by its behavioral model for each time step. The action selection takes place in three phases. First, the actor perceives the objects and the other actors in the environment, which provides information on their nature and position. This information is used by the behavioral model to decide the action to take, which results in a motion procedure with its parameters: e.g. grasp an object or walk with a new speed and a new direction. Finally, the actor performs the motion.

The control of the motion is based on stimulations from the environment. A stimulation consists of information about the nature of the stimulation source and its position. The stimulation source can be an actor, an object or a position to reach. A transfer function, applied on the stimulation, computes an acceleration which is used to control the motion (Braitenberg 1984, Wilhelms 1990). For example, the transfer function of a collision avoidance behavior increases the intensity of the acceleration and modifies the direction as a function of the distance in order to avoid the obstacle.

A complex behavior generally emerges from the interaction of simpler behaviors. A set of behaviors can be sorted in a precedence order (Maes 1991b). The behavior of a walking human in a crowded street results from the interaction between a forward walking behavior, and a collision avoidance behavior with other humans. The second behavior will take precedence over the first when there is a risk of collision. A behavior can also be described with a sequence of simpler behaviors. Therefore, the behavioral model is based on a hierarchical decomposition in two layers (Hügli et al. 1994, Tyrell 1993). The first layer decomposes a behavior in behavioral units and the second layer decomposes a behavioral unit to elementary behaviors.

An elementary behavior is stimulated by a specific category of actors or objects and applies a transfer function on the perceived stimulation to control a walking action. A road crossing behavior can be stimulated by incoming cars and the transfer function computes a positive acceleration in direction of the other side of the road only when there are no incoming cars.

A behavioral unit is a part of a complex behavior and is composed of one or several elementary behaviors performed in sequence. A finite-state machine is used to represent this sequence. Every state of the machine is linked with an elementary behavior and has a boolean transition function. Only one state of the machine is active at a time and this state is the elementary behavior currently performed. The boolean transition function evaluates if the elementary behavior is finished and when it is the case the transition to the next state is achieved. The transition condition can carry on the intensity of the stimulation or a maximum delay for performing the action linked to the behavior. An active behavioral unit can also inhibit another unit if its behavior takes precedence over the other unit behavior. For this purpose, a unit has a degree of activation and a degree of inhibition. The degree of activation is determined by the intensity of the stimulation perceived by the active elementary behavior of the unit and the degree of inhibition is modified by another preceding unit. The behavioral unit with the highest degree of activation and the lowest degree of inhibition is selected as the active behavior and the motion is produced by its current elementary behavior.

We implemented simple behaviors using this approach. Figure 1 shows actors following a leader. Figure 2 shows actors in a virtual office with a distinct queue for each counter.

Figure 1. Virtual actors following a leader

Figure 2. Virtual actors in a virtual office with a distinct queue for each counter

# 2 Virtual Sensors

## 2.1 Perception through Virtual Sensors

The problem of simulating the behavior of a synthetic actor in an environment may be divided into two parts: 1) provide to the actor a knowledge of his environment, and 2) to make react him to this environment.

The first problem consists of creating an information flow from the environment to the actor. This synthetic environment is made of 3D geometric shapes.

One solution is to give the actor access to the exact position of each object in the complete environment database corresponding to the synthetic world. This solution could work for a very " small world ", but it becomes impracticable when the number of objects increases. Moreover, this approach does not correspond to reality where people do not have knowledge about the complete environment.

Another approach has been proposed by Reynolds (1987): the synthetic actor has knowledge about the environment located in a sphere centered on him. Moreover, the accuracy of the knowledge about the objects of the environment decreases with the distance. This is of course a more realistic approach, but as mentioned by Reynolds, an animal or a human being has always around him areas where his sensitivity is more important. Consider, for example, the vision of birds (birds have been simulated by Reynolds), they have a view angle of 300º and a stereoscopic view of only 15º. The sphere model does not correspond to the sensitivity area of the vision. Reynolds goes one step further and states that if actors can see their environment, they will improve their trajectory planning.

This means that the vision is a realistic information flow. Unfortunately, what is realistic to do for an human being walking in a corridor seems unrealistic to do for a computer. However, using hardware developments like the **graphic engine** (Clark

1982), it is possible to give a geometric description of 3D objects together with the viewpoint and the interest point of a synthetic actor in order to get the vision on the screen. This vision may then be interpreted like the synthetic actor vision. This is our approach as described in this chapter. More generally, in order to implement perception, virtual humans should be equipped with visual, tactile and auditory sensors. These sensors should be used as a basis for implementing everyday human behaviour such as visually directed locomotion, handling objects, and responding to sounds and utterances. For synthetic audition, in a first step, we model a sound environment where the synthetic actor can directly access to positional and semantic sound source information of a audible sound event. Simulating the haptic system corresponds roughly to a collision detection process. But, the most important perceptual subsystem is the vision system. A vision based approach for virtual humans is a very important perceptual subsystem and is for example essential for navigation in virtual worlds. It is an ideal approach for modeling a behavioral animation and offers a universal approach to pass the necessary information from the environment to the virtual human in the problems of path searching, obstacle avoidance, and internal knowledge representation with learning and forgetting. In the next sections, we describe our approach for the three types of virtual sensors: vision, audition and haptic.

## 2.2 Virtual vision

Although the use of vision to give behavior to synthetic actors seems similar to the use of vision for intelligent mobile robots (Horswill 1993, Tsuji and Li 1993), it is quite different. This is because the vision of the synthetic actor is itself a synthetic vision. Using a synthetic vision allow us to skip all the problems of pattern recognition and distance detection, problems which still are the most difficult parts in robotics vision. However some interesting work has been done in the topic of intelligent mobile robots, especially for action-perception coordination problems. For example, Crowley (1987), working with surveillance robots states that "most low level perception and navigation tasks are algorithmic in nature; at the highest levels, decisions regarding which actions to perform are based on knowledge relevant to each situation". This remark gives us the hypothesis on which our vision-based model of behavioral animation is built.

We first introduced (Renault et al. 1990) the concept of synthetic vision as a main information channel between the environment and the virtual actor. Reynolds (1993, 1994) more recently described an evolved, vision-based behavioral model of coordinated group motion, he also showed how obstacle avoidance behavior can emerge from evolution under selection pressure from an appropriate measure using a simple computational model of visual perception and locomotion. The Genetic Programming is used to model evolution. Tu and Terzopoulos (1994, 1994b) also use a kind of synthetic vision for their artificial fishes.

In (Renault et al. 1990), each pixel of the vision input has the semantic information giving the object projected on this pixel, and numerical information giving the distance to this object. So, it is easy to know, for example, that there is a table just in front at 3 meters. With this information, we can directly deal with the problematic question: "what do I do with such information in a navigation system?" The synthetic actor perceives his environment from a small window of typically 30x30 pixels in which the environment is rendered from his point of view. As he can access z buffer values of the pixels, the color of the pixels and his own position he can locate visible objects in his 3D environment. This information is sufficient for some local navigation.

We can model a certain type of virtual world representation where the actor maintains a low level fast synthetic vision system but where he can access some important information directly from the environment without having to extract it from the vision image. In vision based grasping for example, an actor can recognize in the image the object to grasp. From the environment he can get the exact position, type and size of the object which allows him to walk to the correct position where he can start the grasping procedure of the object based on geometrical data of the object representation in the world. This mix of vision based recognition and world representation access will make him fast enough to react in real time. The role of synthetic vision can even be reduced to a visibility test and the semantic information recognition in the image can be done by simple color coding and non shading rendering techniques. Thus, position and semantic information of an object can be obtained directly form the environment world after being filtered. Figure 3 shows an example of of local navigation using synthetic vision.

## 2.3 Virtual Audition

In real life, the behavior of persons or animals is very often influenced by sounds. For this reason, we developed a framework for modeling a 3D acoustic environment with sound sources and microphones. Now, our virtual actors are able to hear (Noser and Thalmann 1995). Any sound source (synthetic or real) should be converted to the AIFF format and processed by the sound renderer. The sound renderer takes into account the real time constraints. So it is capable to render each time increment for each microphone in "real time" by taking into account the final propagation speed of sound and the moving sound sources and microphones. So, the Doppler effect, for example, is audible.

In sound event generation, we integrated in our L-system-based software (Noser et al. 1992, Noser and Thalmann 1993) a peak detector of a force field which allows to detect collision between physical objects. These collisions can be coupled to sound emission events. For example, tennis playing with sound effects (ball-floor and ball-racket collision) has been realized.

Figure 3. Vision-based local navigation

The acoustic environment is composed of sound sources and a propagation medium. The sound sources can produce sound events composed of a position in the world, a type of sound, and a start and an end time of the sound. The propagation medium corresponds to the sound event handler which controls the sound events and transmits the sounds to the ears of the actors and/or to a user and/or a soundtrack file. We suppose an infinite sound propagation speed of the sound without weakening of the signal. The sound sources are all omnidirectional, and the environment is non reverberant.

## 2.4 Virtual tactile

One of our aims is to build a behavioral model based on tactile sensory input received at the level of skin from the environment. This sensory information can be used in tasks as touching objects, pressing buttons or kicking objects For example at basic level, human should sense physical objects if any part of the body touches them and gather sensory information. This sensory information is made use of in such tasks as reaching out for an object, navigation etc. For example if a human is standing, the feet are in constant contact with the supporting floor. But during walking motion each

foot alternately experiences the loss of this contact. Traditionally these motions are simulated using dynamic and kinematic constraints on human joints. But there are cases where information from external environment is needed. For example when a human descends a stair case, the motion should change from walk to descent based on achieving contact with the steps of the stairway. Thus the environment imposes constraints on the human locomotion. We propose to encapsulate these constraints using tactile sensors to guide the human figure in various complex situations other than the normal walking.

As already mentioned, simulating the haptic system corresponds roughly to a collision detection process. In order to simulate sensorial tactile events, a module has been designed to define a set solid objects and a set of sensor points attached to an actor. The sensor points can move around in space and collide with the above mentioned solid objects. Collisions with other objects out of this set are not detected. The only objective of collision detection is to inform the actor that there is a contact detected with an object and which object it is. Standard collision detection tests rely on bounding boxes or bounding spheres for efficient simulation of object interactions. But when highly irregular objects are present, such tests are bound to be ineffective. We need much 'tighter' bounding space than a box or a sphere could provide. We make use (Bandi and Thalmann 1995) of a variant of a digital line drawing technique called DDA (Fujimoto et al. 1986) to digitize the surface of such objects to get very tight fitting bounds that can be used in preliminary collision detection. The digitization of these objects can be stored in octree structures which permits optimum use of memory space. When we deal with 'static' objects, the digitized objects can be permanently stored in memory for the duration of simulation, thus avoiding the trouble of digitizing them at every simulation step.

# 3 Perception-based Actions

## 3.1 Action level

Synthetic vision, audition and tactile allow the actor to perceive the environment. Based on this information, his behavioral mechanism will determine the actions he will perform. Actions may be at several degrees of complexity. An actor may simply evolve in his environment or he may interact with this environment or even communicate with other actors. We will emphasize three types of actions: navigation and locomotion, grasping and ball games.

Actions are performed using the common architecture for motion control developed in the European projects HUMANOID (Boulic et al. 1995) and HUMANOID-2. HUMANOID has led to the development of a complete system for animating virtual actors (see Fig.4) for film production. HUMANOID-2 currently extends the project for

realtime applications and behavioral aspects as described in this chapter. The heart of the HUMANOID software is the motion control part which includes 5 generators: keyframing, inverse kinematics, dynamics (see Fig.5), walking and grasping and high-level tools to combine and blend them. An interactive application TRACK (Boulic et al. 1994) has been also developed to create films and sequences to be played in realtime playback for multimedia applications.

Figure 4. Synthetic actors

### 3.2 Perception-based navigation and locomotion

When the actor evolves in his environment, a simple walking model is not sufficient, the actor has to adapt his trajectory based on the variations of terrain by bypassing, jumping or climbing the obstacles he meets. The bypassing of obstacles consists in changing the direction and velocity of the walking of the actor. Jumping and climbing correspond to more complex motion. These actions should generate parameterized motion depending on the height and the length of the obstacle for a jump and the height and location of the feet for climbing the obstacle. These characteristics are determined by the actor from his perception.

Figure 5. Dynamics-based motion

The actor can be directed by giving his linear speed and his angular speed or by giving a position to reach. In the first case, the actor makes no perception (virtual vision). He just walks at the given linear speed and turns at the given angular speed. In the second case, the actor makes use of virtual vision enabling him to avoid obstacles. The vision based navigation can be local or global. With a local navigation, the agent goes straight on to his goal and it is possible that he cannot reach it. With a global navigation, the actor first tries to find a path to his goal and if the path exists, the actor follows it until he reaches the goal position or until he detects a collision by his vision. During global navigation the actor memorizes his perceived environment by voxelizing it (as explained in Section 11.3.3), based on his synthetic vision. Next Section will give more details on local and global navigation.

We developed a special automata for walking in complex environments with local vision based path optimization. So an actor continues walking even if he detects a future collision in front of him. By dynamically figuring out a new path during

walking he can avoid the collision without halting. We also proposed a system for the automatic derivation of a human curved walking trajectory (Boulic et al. 1994b) from the analysis provided by its synthetic vision module. A general methodology associates the two low-level modules of vision and walking with a planning module which establishes the middle term path from the knowledge of the visualized environment. The planning is made under the constraint of minimizing the distance, the speed variation and the curvature cost. Moreover, the planning may trigger the alternate walking motion whenever the decreasing in curvature cost is higher than the associated increasing in speed variation cost due to the corresponding halt and restart. The Analysis of walking trajectories on a discrete environment with sparse foothold locations has been also completed (Boulic et al. 1993b) regarding the vision-based recognition of footholds, the local path planning, the next step selection and the curved body trajectory. The walking model used is based on biomechanical studies of specific motion pattern (Boulic et al. 1990). Figure 6 shows an example of walking from the film *Still Walking* (Magnenat Thalmann and Thalmann 1991).

Figure 6. Biomechanical model for walking from the film *Still Walking*

### 3.3 Global and local navigation

The task of a navigation system is to plan a path to a specified goal and to execute this plan, modifying it as necessary to avoid unexpected obstacles (Crowley 1987). This task can be decomposed into global navigation and local navigation. The global navigation uses a prelearned model of the domain which may be a somewhat simplified description of the synthetic world and might not reflect recent changes in the environment. This prelearned model, or map, is used to perform a path planning algorithm.

The local navigation algorithm uses the direct input information from the environment to reach goals and sub-goals given by the global navigation and to avoid unexpected obstacles. The local navigation algorithm has no model of the environment, and doesn't know the position of the actor in the world.

Once again to make a comparison with a human being, close your eyes, try to see the corridor near your room, and how to follow it. No problem, your were using your "visual memory," which corresponds to the global navigation in our system. Now stand up and go to the corridor near your room, then close your eyes and try to cross the corridor... There the problems begin (you know that there is a skateboard in front of your boss's door but...). This is an empirical demonstration of the functionalities of the local navigation as we define it in our system.

The global navigation needs a model of the environment to perform path-planning. This model is constructed with the information coming from the sensory system. Most navigation systems developed in robotics for intelligent mobile robots are based on the accumulation of accurate geometrical descriptions of the environment. Kuipers et al. (1988) give a nearly exhaustive list of such methods using quantitative world modeling. In robotics, due to low mechanical accuracy and sensory errors, these methods have failed in large scale area. We don't have this problem in Computer Graphics because we have access to the world coordinates of the actor, and because the synthetic vision or other simulations of perception systems are more accurate. We develop a 3D geometric model, based on grid, implemented as an octree. Elfes (1990) proposed a 2D geometric model based on grid but using a Bayesian probabilistic approach to filter non accurate information coming from various sensor positions. Roth-Tabak (1989) proposed a 3D geometric model based on a grid but for a static world.

In the last few years, research in robot navigation has tended towards a more qualitative approach to world modeling, first to overcome the fragility of purely metrical methods, but especially, because humans do not make spatial reasoning on a continuous map, but rather on a discrete map (Sowa 1964). Kuipers et al. (1988) present a topological model as the basic element of the cognitive map. This model consists of a set of nodes and arcs, where nodes represent distinctively recognizable places in the environment, and arcs represent travel edges connecting them. Travel edges corresponding to arcs are defined by local navigation strategies which describe

how a robot can follow the link connecting two distinct places. These local navigation strategies correspond to the Displacement Local Automata (DLA) implemented in the local navigation part of our system. These DLAs work as a black box which has the knowledge to create goals and sub-goals in a specific local environment. They can be thought of as low-level navigation reflexes which use vision, reflexes which are automatically performed by the adults.

**The octree as visual memory representation**

Noser et al. (1995) use an octree as the internal representation of the environment seen by an actor because it offers several interesting features. With an octree we can easily construct enclosing objects by choosing the maximum depth level of the subdivision of space. Detailed objects like flowers and trees do not need to be represented in complete detail in the problem of path searching. It is sufficient to represent them by some enclosing cubes corresponding to the occupied voxels of the octree. The octree adapts itself to the complexity of the 3D environment, as it is a dynamic data structure making a recursive subdivision of space. Intersection tests are easy. To decide whether a voxel is occupied or not, we only have to go to the maximum depth (5-10) of the octree by some elementary addressing operations. The examination of the neighborhood of a voxel is immediate, too.

Another interesting property of the octree is the fact that it represents a graph of a 3D environment. We may consider, for example, all the empty voxels as nodes of a graph, where the neighbors are connected by edges. We can apply all the algorithms of graph theory directly on the octree and it is not necessary to change the representation.

Perhaps the most interesting property of the octree is the simple and fast transition from the 2D image to the 3D representation. All we have to do is take each pixel with its depth information (given by the z-buffer value) and calculate its 3D position in the octree space. Then, we insert it in the octree with a maximum recursion depth level. The corresponding voxel will be marked as occupied with possible additional information depending on the current application.

The octree has to represent the visual memory of an actor in a 3D environment with static and dynamic objects. Objects in this environment can grow, shrink, move or disappear. In a static environment (growing objects are still allowed) an *insert* operation for the octree is sufficient to get an approximate representation of the world. If there are moving or disappearing objects like cars, other actors, or opening and closing doors, we also need a *delete* operation for the octree. The *insert* operation is simple enough. The *delete* operation however, is more complicated. Our approach follows.

At a given instant, each pixel is inserted into the octree and the corresponding voxel is marked with the actual time stamp. After the insertion of all image pixels, all the voxels in the vision volume are tested whether they have disappeared or not. The principle of such a test is shown in Figure 7.

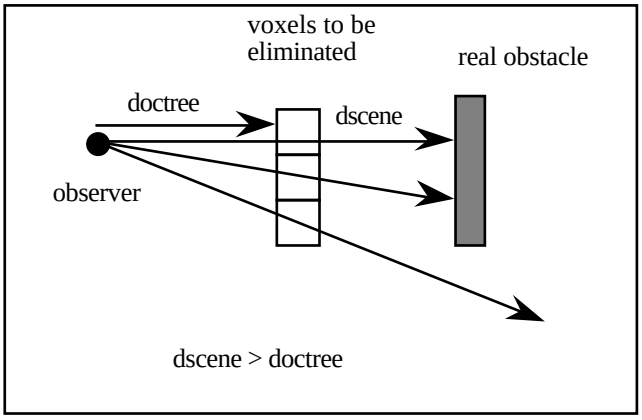



Figure 7. Object removal in the octree.

An occupied voxel has to be deleted only if there is no real object either at its position or in front of it. This condition is expressed by equation (1).

$d_{scene} > d_{octree}$ (distances from the observer) (1)

The distance of the voxel of the octree from the observer has to be smaller than the distance of a real object or the background. To get the distance of the voxel from the observer you have to transform the observer into the octree coordinate system and apply the corresponding modeling and perspective transformation to the voxel. Doing this with all voxels in the vision volume, we get the image of the memory in normalized coordinates and can compare it directly with the z buffer values of the 2D synthetic world image .

The algorithm of path finding is based on path searching in a graph. Each free voxel not occupied by an obstacle is interpreted as a node of the graph. All the neighbor voxels are considered to be connected by an edge. So, the octree represents a graph with nodes and edges. The algorithm of path searching uses the principle of backtracking and memorizes all tested nodes in a sorted list. With this list of already tested nodes, circuits can be avoided, and situations without a path from a given source to a given destination can be detected. In a first approach, a path is represented by a sequence of free nodes. To avoid a combinatorial explosion of possibilities in graph searching, we use a heuristic depth first search. The first type of heuristic is characterized by the choice of the neighbors of the current voxel. For example, if we know that the search is done in a plane (2D), we will only examine the neighbors in that plane. So, we can reduce the numbers of new voxels to be tested from 26 to 8. The second type of heuristic is determined by the order of the new neighbors to be tested. If we are searching a path from the current position to an aim, we will sort the list of the new neighbors according to their distance from the aim and we will continue the depth

first search with the nearest neighbor to the aim. The third type of heuristic is determined by some additional conditions on the new neighboring voxels to be examined. If we want, for example, that the actor is bound to the ground ( if he/she cannot fly) in a 3D environment with stairs, ramps, bridges, holes, etc., we can use only neighbors which have an occupied voxel beneath themselves. With these simple condition, our actor is now able to avoid holes, use bridges and mount or descend stairs and ramps.

The path finding procedure is a mental process of the actor, which is based on the contents of his visual memory (octree). This means that during his reasoning on a possible path, he does not move. Very often, though, he is placed in an unknown environment, which he has still not seen and memorized. In this case, he cannot find a path using, for example, some conditional heuristic. So, he is forced to explore his environment guided by his vision and a heuristic. This exploring is an active process and the actor has to walk and memorize what he sees. In this case, an heuristic depth first search step can be used to guide the actor to guarantee that he finds a way if one exists.

If the actor, for example, is enclosed in a house, there will be no path to a destination outside the house. In this case, he will explore the parts of the interior accessible to him according to the heuristic. He will finish his search by having memorized the interior of the house and the conclusion that he is enclosed. He should avoid turning infinitely in a loop and he can recognize that he has checked all possibilities to find an exit.

To illustrate the capabilities of the synthetic vision system, we have developed several examples. First, an actor is placed inside a maze with an impasse, a circuit and some animated flowers. The actor's first goal is a point outside the maze. After some time, based on 2D heuristic, the actor succeeds in finding his goal. When he had completely memorized the impasse and the circuit, he avoided them. After reaching his first goal, he had nearly complete visual octree representation of his environment and he could find again his way without any problem by a simple reasoning process. We have also implemented a more complex environment with flowers and butterflies; the complex flowers were represented in the octree memory by enclosing cubes.

**Local Navigation System**

The local navigation system can be decomposed into three modules. The *vision* module, conceptually the perception system, draws a perspective view of the world in the vision window, constructs the vision array and can perform some low level operation on the vision array. The *controller* module, corresponding to the decision system, contains the main loop for the navigation system, and decides on the creation of the goals and administrates the DLAs. The *performer*, corresponding to the task execution system, contains all the DLAs.

- **The Vision module**
  We use the hardware facilities of the Silicon Graphics IRIS to create the synthetic vision, more precisely we use the flat shading and z-buffer drawing capabilities of the graphic engine. The vision module has a modified version of the drawing routine traveling the world; instead of giving the real color of the object to the graphic engine, this routine gives a code, call the *vision_id*, which is unique for each object and actor in the world. This code allows the image recognition and interpretation. Once the drawing is done, the window buffer is copied into a 2D array. This array contains the vision_id and the z-buffer depth for each pixel. This array is referred as the *view*.

- **The Controller module**
  In local navigation there are two goals. These two goals are geometrical goals, and are defined in the local 2D coordinate system of the actor. The actor itself is the center of this coordinate system, one axis is defined by the direction "in front", the other axis is defined by the "side" direction. The global goal, or final goal, is the goal the actor must reach. The local goal, or temporary goal, is the goal the actor creates to avoid the obstacles encountered in the path towards the global goal. These goals are created by the Displacement Local Automata (DLA), or given by the animator or by the global navigation system. The main task of the controller is to create these goals created and to make the actor reach them.
  Goal creation and actor displacement are performed by the DLAs. The controller selects the appropriate DLA either by knowing some internal set-up of the actor, or by visual by analyzing the environment. For example, if the actor has a guide, the controller will choose the DLA *follow_the_guide*. Otherwise, from a 360 look-around, the controller will determine the visible objects and then determine the DLA corresponding to these objects. No real interpretation of the topology of the environment (as in Kuipers et al. 1989) has yet been implemented. The choice of the DLA is hardcoded by the presence of some particular objects, given by their *vision_id*.
  The actor has an internal clock administrated by the controller. This clock is used by the controller to refresh the global and local goal at regular intervals. The interval is given by the *attention_rate*, a variable set-up for each actor that can be changed by the user or by the controller. This variable is an essential parameter of the system: with a too high attention rate the actor spends most of his time analyzing the environment and real-time motion is impossible; with a too low attention rate, the actor starts to act blindly, going through objects. A compromise must be found between these two extremes.

- **The Performer module**
  This module contains the DLAs. There are three families of DLA: the DLAs creating the global goal (follow_the_corridor, follow_the_wall,

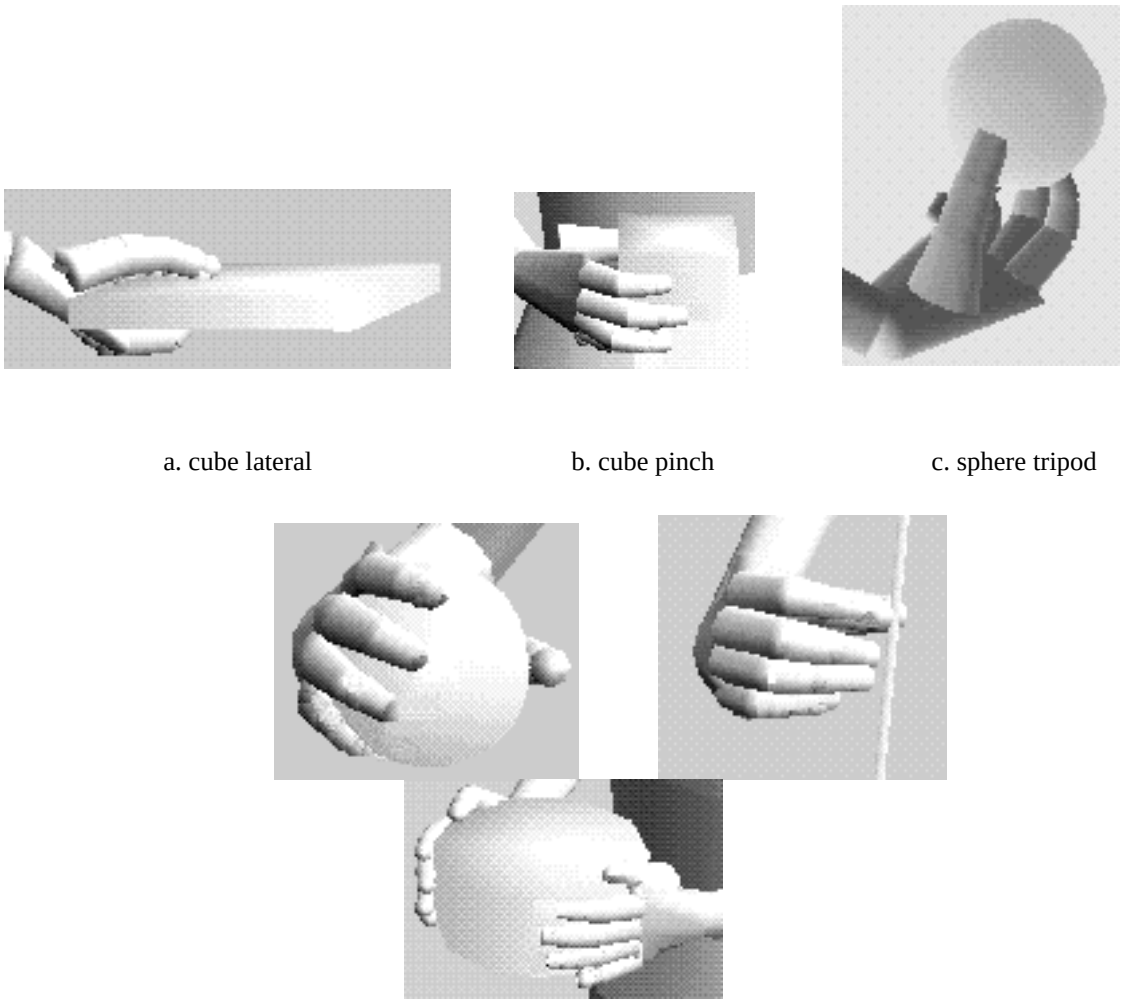
follow_the_visual_guide), the DLAs creating the local goal (avoid_obstacle, closest_to_goal), and the DLAs effectively moving the actor (go_to_global_goal). The DLAs creating goals only use the vision as input. All these DLAs have access to a library of routines performing high level operations on the vision. A detailed algorithm of the use of vision to find avoidance goal is described by Renault et al. (1990) .

### 3.4 Perception-based grasping

With the advents of virtual actors in computer animation, research in human grasping has become a key issue in this field. Magnenat Thalmann et al. (1988) proposed a semi-automatic way of grasping for a virtual actor interacting with the environment. A knowledge-based approach is suitable for simulating human grasping, and an expert system can be used for this purpose (Rijpkema and Girard 1991). Kunii et al. (1993) have also presented a model of hands and arms based on manifold.

Our approach is based on three steps:

- Inverse kinematics to find the final arm posture
- Heuristic grasping decision. Based on a grasp taxonomy, Mas and Thalmann (1994) proposed a completely automatic grasping system for synthetic actors. In particular, the system can decide to use a pinch when the object is too small to be grasped by more than two fingers or to use a two-handed grasp when the object is too large. Figure 8 shows several examples.
- **Multi-sensor hand.** Our approach (Huang et al. 1995) is adapted from the use of proximity sensors in Robotics (Espiau and Boulic 1985), the sensor-actuator networks (van de Panne and Fiume 1993) and recent work on human grasping (Mas and Thalmann 1994). In our work, the sphere multi-sensors have both tactile and length sensor properties, and have been found very efficient for synthetic actor grasping problem. Multi-sensors are considered as a group of objects attached to the articulated figure. A sensor is activated for any collision with other objects or sensors. Here we select sphere sensors for their efficiency in collision detection (Figure 9).



a. cube lateral b. cube pinch c. sphere tripod

d. sphere wrap e. cylinder pinch f. cylinder wrap

Figure 8. The different grasping ways for different objects

Each sphere sensor is fitted to its associated joint shape with different radii. This configuration is important in our method because when a sensor is activated in a finger, only the articulations above it stop moving, while others can still move. By doing this way, all the fingers are finally positioned naturally around the object, as shown in Figure 11.10.

To summarize: after the hand center frame is aligned with the object frame, the fingers are closed according to the different strategies, e.g. pinch, wrap, lateral, etc., while sensor-object and sensor-sensor collisions are detected. When one sensor is activated, all articulations above it are blocked. The grasping is completed when the remaining sensors are activated or the joints reach their limit.

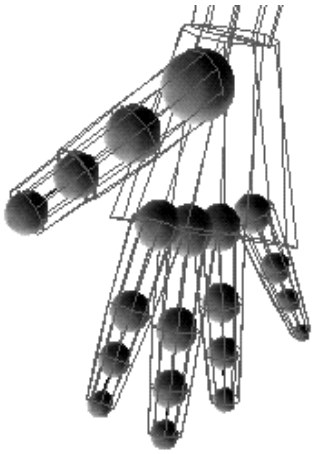


Figure 9. The hand with sphere sensors at each joint

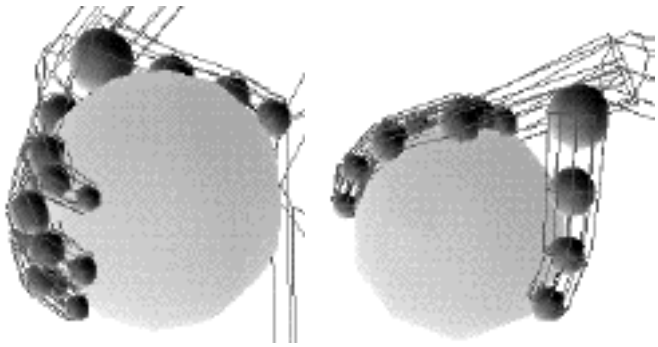


Figure 10. One example shown sensors in grasping

All the grasping functions mentioned above have been implemented and integrated into the TRACK system (Boulic et al. 1994). The examples shown in this section were performed on SGI Indigo 2 in real time. The first example shows an actor using two hands for grasping a cube and a cylinder. The strategies are different according to the types of the objects (Fig.11). The second example (Fig.12) shows two actors grasping a frustum. The first actor uses two hands to grasp the thick part, while the second only uses one hand. In Figure 13, we extend the grasping method to interactions between two actors.

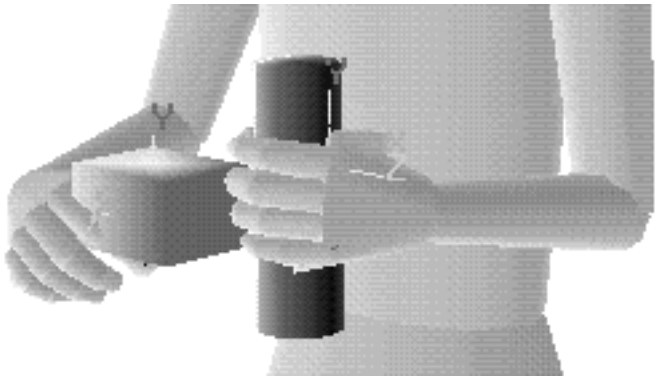


Figure 11.11. Grasp different objects with two hands

Figure 11.12. Two actors grasp a frustum at different positions with different ways

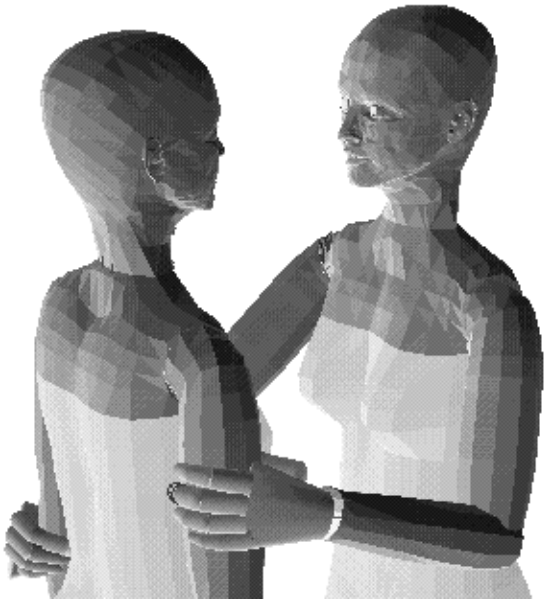


Figure 13 Interaction between two actors modeled by triangle meshes

### 3.5 Vision-based Tennis Playing

Tennis playing is a human activity which is severely based on the vision of the players. In our model, we use the vision system to recognize the flying ball, to estimate its trajectory and to localize the partner for game strategy planning. The geometric characteristics of the tennis court however, make part of the players knowledge. For the dynamics simulation of the ball, gravity, net, ground and the racquet we use the force field approach developed for the L-system animation system. The tracking of the ball by the vision system is controlled by a special automata. A prototype of this automata is already able to track the ball, to estimate the collision time and collision point of ball and racquet and to perform successfully a hit with given force and a given resulting ball direction. In a first step, we have a prototype where only two racquets with synthetic vision can play against each other, in order to develop, test and improve game strategy and the physical modeling. Figure 14 shows the prototype system. The integration of the corresponding locomotor system of a sophisticated actor is under development as seen in Fig.15.

Figure 14. Vision-based tennis players

Figure 15. Marilyn playing tennis

In the navigation problem each colored pixel is interpreted as an obstacle. No semantic information is necessary. In tennis playing however, the actor has to distinguish between the partner, the ball and the rest of the environment. The ball has to be recognized, its trajectory has to be estimated and it has to be followed by the vision system. At the beginning of a ball exchange, the actor has to verify that its partner is ready. During the game the actor needs also his partner's position for his play strategy.

To recognize objects in the image we use color coding. The actor knows that a certain object is made of a specific material. When it scans the image it looks for the corresponding pixels and calculates its average position and its approximate size. Thus each actor can extract some limited semantic information from the image.

Once the actor has recognized the ball, it follows it with his vision system and adjusts at each frame his field of view. To play tennis each partner has to estimate the future racket-ball collision position and time and to move as fast as possible to this point. At each frame (1/25 sec) the actor memorizes the ball position. So, every n-th frame the actor can derive the current velocity of the ball. From this current velocity and the current position of the ball it can calculate the future impact point and impact time. We suppose that the actor wants to hit the ball at a certain height h.

In the next phase the actor has to play the ball. Now he has to determine the racket speed and its orientation to play the ball to a given place. Before playing the ball the actor has to decide where to play. In our simulation approach he looks where his partner is placed and then he plays the ball in the most distant corner of the court.

All the above features are coordinated by a specialized "tennis play" automata. First an actor goes to his start position. There he waits until his partner is ready. Then he looks for the ball, which is thrown into the game. Once the vision system has found the ball, it always follows it by adjusting the field of view angle. If the ball is flying towards the actor, it starts estimating the impact point. Once the ball has passed the net, the actor localizes his partner with his vision system during one frame. This information is used for the game strategy. After playing the ball, the actor goes back to his start point and waits until the ball comes back to play it again.

# 4 Virtual Environments

## 4.1 Presence and Immersion

**Presence** is the fact or condition of being present and it is something (as a spirit) felt or believed to be present. This spirit is essential in Virtual Reality. As stated by Slater and Usoh (1994), immersion may lead to a sense of presence. This is an emergent psychological property of an immersive system, and refers to the participant's sense of "being there" in the world created by the Virtual environment

system. Astheimer et al. (1994) define an immersive system as follows: if the user cannot tell, which reality is "real", and which one is "virtual", then the computer generated one is immersive.

We are currently creating interactive and immersive real-time simulations of our smart virtual actors. These actors will be able:

- to move from one place to another by walking, bypassing, jumping or climbing obstacles.
- to move objects in the Virtual Space

The simulation will be performed in Virtual Environments allowing the participant (real human) to move the objects in the Virtual Space using VR-devices.

In the next section, we analyze the problems of processing virtual sensors in the case of Virtual Environments.

## 4.2 Virtual Sensors in Virtual Environments

We have seen that virtual vision can be a powerful tool in modeling virtual autonomous actors in virtual worlds. Such actors in virtual worlds can have different degrees of autonomy and different sensing channels to the environment where they behave in a certain manner. In robotics for example, the agent (the robot) only gets information of the world by his sensors. If he has a vision sensor, he has to extract all the semantic information of the world from an image. This is a very difficult task and thus, according to the actual state of knowledge, his intelligence is very restricted and his behavior is limited to some navigational tasks by avoiding collisions.

In virtual worlds the situation is different as we can provide some extra information to the actors making him more intelligent and faster. Until now, we tried to make the actors completely independent of the virtual worlds' internal representation and they only got the vision image and their position as sensory information. Thus, vision based navigation, collision avoidance, visual memory and tennis playing could be successfully modeled. We now integrate actors in virtual reality (VR) where real time constraints demand fast and intelligent reactions of actors with a set of elementary actions like grasping objects, sitting on chairs, jumping over obstacles, pressing buttons, running, ..... To reach this goal we model a certain type of virtual world representation where the actor maintains a low level fast synthetic vision system but where he can access some important information directly from the environment without having to extract it from the vision image.

A human being can participate in VR by the head-mounted display and the earphones. He cannot get any internal VR information. His only source of knowledge from the VR is communicated by the vision and the sound (and perhaps some tactile sensory information). His behavior is strongly influenced by this sensory input and his

proper intelligence. In order to process the virtual actor vision in a similar way than the vision of the participant, we need to have a different model. In this case, the only information obtained by the virtual actor will be the vision image with the z-buffer values and the shaded and colored pixels (he may also get the sound signal and some tactile sensor information). Such a virtual actor would be independent of each VR representation (as a human too) and he could in the same manner communicate with human participants and other virtual actors.

For virtual audition, we encounter the same problem as in synthetic vision. The real time constraints in VR demand fast reaction to sound signals and fast recognition of the semantic it carries. Thus, we plan in a first step to model a sound environment where the synthetic actor can directly access to positional and semantic sound source information of a audible sound event. This allows him to localize and recognize one or more sound sources in a reliable way and to react immediately.

This access to the sound environment representation, however, makes him dependent of it and lets the communication problem with human participants in VR unresolved. That is why, we try to realize a really independent actor as already mentioned above. This type of actor will get the same sound signal (digitized) as any other human participant in VR through his earphones. From this sound signals (stereo) the actor can estimate the position of a sound source and with an added speech recognition module he should be capable to extract some semantic information of some spoken language. Thus the synthetic actor should be able to understand and speak a reduced set of vocabulary allowing him also to communicate with human participants in VR.

Concerning virtual haptic sense, we have already implemented a case of 3D interaction with VR technology. The participant may place an object into the Virtual Space using the CyberGlove and the virtual actor will try to grasp it and put it on a virtual table for example. The actor interacts with the environment by grasping the object and moving it. At the beginning of interactive grasping, only the hand center sensor is active. The six palm values from CyberGlove are used to move it toward the object. Inverse kinematics is used to update the arm postures from hand center movement. After the sensor is activated, the hand is close enough to the object final frame. The hand center sensor is deactivated and multi-sensors on hand are now used, to detect sensor object collision. The following process is similar to the multi-sensor method discussed in Section 11.3.4. The major difference is that the grasping strategy is defined interactively. One example is shown in Figure 11.16.

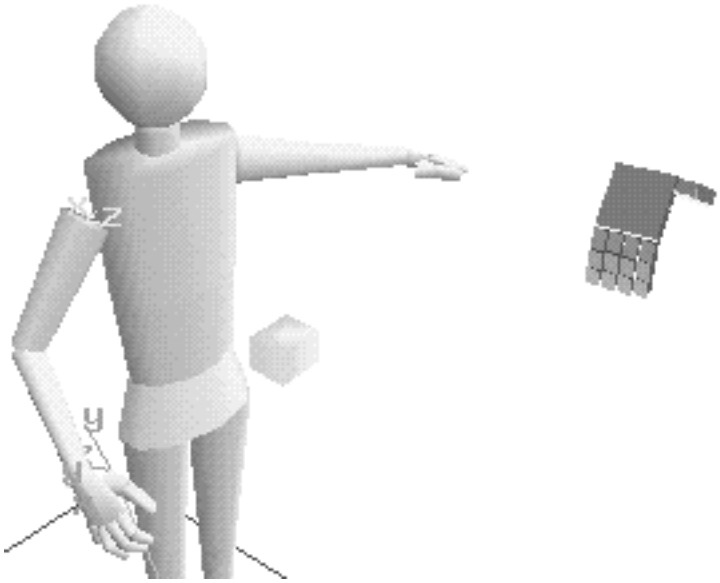


Figure 16. 3D interactive grasping with CyberGlove

## 5 Conclusion

In this chapter, we have presented a new approach to implement autonomous virtual actors in virtual worlds based on perception and virtual sensors. We believe this is an ideal approach for modeling a behavioral animation and offers a universal approach to pass the necessary information from the environment to an actor in the problems of path searching, obstacle avoidance, game playing, and internal knowledge representation with learning and forgetting characteristics. We also think that this new way of defining animation is a convenient and universal high level approach to simulate the behavior of intelligent human actors in dynamics and complex environments including virtual environments.

## 6 Acknowledgments

The author is grateful to the people who contributed to this work, in particular Srikhan Bandi, Pascal Bécheiraz, Ronan Boulic, and Serge Rezzonico. The research was supported by the Swiss National Science Research Foundation, the Federal Office for Education and Science, and is part of the Esprit Project HUMANOID and HUMANOID-2.

# 7 References

Astheimer P., Dai, Gobel M. Kruse R., Müller S., Zachmann G. (1994) “Realism in Virtual Reality”, in: (Magnenat Thalmann and Thalmann, eds) Artificial Life and Virtual Reality, John Wiley, Chichester, pp.189-208.

Bandi S., Thalmann D. (1995) “An Adaptive Spatial Subdivision of the Object Space for Fast Collision Detection of Animated Rigid Bodies”, Proc. Eurographics ‘95 (to appear)

Bates J, Loyall AB, Reilly WS (1992) “An architecture for Action, Emotion, and Social Behavior”, Proc. Fourth Europeans Workshop on Modeling Autonomous Agents in a multi Agents World, S. Martino al Cimino, Italy.

Boulic R., Capin T., Kalra P., Lintermann B., Moccozet L., Molet T., Huang Z., Magnenat-Thalmann N., Saar K., Schmitt A., Shen J. and Thalmann D. (1995) "A system for the Parallel Integrated Motion of Multiple Deformable Human Characters with Collision Detection", EUROGRAPHICS' 95, Maastricht.

Boulic R., Huang Z., Magnenat-Thalmann N., Thalmann D. (1994) ”Goal-Oriented Design and Correction of Articulated Figure Motion with the TRACK System” , Comput. & Graphics, Vol. 18, No. 4, pp. 443-452.

Boulic R., Noser H., Thalmann D. (1993) ”Vision-Based Human Free-Walking on Sparse Foothold Locations", Fourth Eurographics Workshop on Animation and Simulation, Barcelona Spain, Eurographics, pp.173-191

Boulic R., Noser H., Thalmann D. (1994b) ”Automatic Derivation of Curved Human Walking Trajectories from Synthetic Vision”, Computer Animation '94, Geneva, IEEE Computer Society Press, pp.93-103.

Boulic R., Thalmann D, Magnenat-Thalmann N. (1990) "A global human walking model with real time kinematic personification" The Visual Computer, 6(6).

Braitenberg V. (1984) ”Vehicles, Experiments in Synthetic Psychology”. The MIT Press.

Clark J.H. (1982) “The Geometric Engine: A VLSI Geometry System for Graphics”, Proc. SIGGRAPH ‘82, Computer Graphics, Vol. 10, No3, pp.127-133.

Crowley J.L. (1987) “Navigation for an Intelligent Mobile Robot”, IEEE journal of Robotics and Automation, Vol. RA-1, No. 1, pp 31-41.

Elfes A. (1990) “Occupancy Grid: A Stochastic Spatial Representation for Active Robot Perception”, Proc. Sixth Conference on Uncertainty in AI.

Espiau B., Boulic R. (1985) ”Collision avoidance for redondants robots with proximity sensors”, Proc. of Third International Symposium of Robotics Research, Gouvieux, October.

Fujimoto A., Tanaka T., Iwata K. (1986) "ARTS: Accelerated Ray-Tracing System", IEEE CG&A, vol.6, No.4, pp.16-26.

Haumann D.R., Parent R.E. (1988) “The Behavioral Test-bed: Obtaining Complex Behavior from Simple Rules”, The Visual Computer, Vol.4, No 6, pp.332-347.

Horswill I. (1993) ”A Simple, Cheap, and Robust Visual Navigation System”, in: From Animals to Animats 2, Proc. 2nd Intern. Conf. on Simulation of Adaptive Behavior, MIT Press, pp.129-136.

Huang Z., Boulic R., Magnenat Thalmann N., Thalmann D. (1995) ”A Multi-sensor Approach for Grasping and 3D Interaction”, Proc. CGI ‘95 (to appear)

Hügli H., Facchinetti C., Tièche F., Müller J.P., Rodriguez M., Gat Y. (1994) ”Architecture Of An Autonomous System: Application to Mobile Robot Navigation”. In Proceedings of Symposium on Artificial Intelligence and Robotics, pp. 97-110.

Kuipers B., Byun Y.T. (1988) “A Robust Qualitative Approach to a Spatial Learning Mobile Robot”, SPIE Sensor Fusion: Spatial Reaoning and Scene Interpretation, Vol. 1003.

Kunii T.L., Tsuchida Y., Matsuda H., Shirahama M., Miura S. (1993) ”A model of hands and arms based on manifold mappings”, Proceedings of CGI'93, pp.381-398.

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

Maes P. (ed.) (1991) “Designing Autonomous Agents”, Bradford MIT Press.

Maes P. (1991b) ”Bottom-Up Mechanism for Behavior Selection in an Artificial Creature”, Proc. First International Conference on Simulation of Adaptive Behavior, 1991.

Magnenat-Thalmann N., Laperrière R., Thalmann D. (1988) “Joint-dependent local deformations for hand animation and object graspin“, Proceedings of Graphics Interface '88, pp.26-33.

Magnenat Thalmann N., Thalmann D. (1991) “Still Walking”, video, 1 min.

Magnenat Thalmann N., Thalmann D. (1994) “Creating Artificial Life in Virtual Reality” in: (Magnenat Thalmann and Thalmann, eds) Artificial Life and Virtual Reality, John Wiley, Chichester, 1994, pp.1-10

Magnenat Thalmann N., Thalmann D. (1995) “Digital Actors for Interactive Television”, Proc. IEEE, July.

Mas S.R., Thalmann D. (1994) ”A Hand Control and Automatic Grasping System for Synthetic Actors”, Proceedings of Eurographic'94, pp.167-178.

Noser H., Thalmann D. (1993) ”L-System-Based Behavioral Animation”, Proc. Pacific Graphics '93, pp.133-146.

Noser H, Thalmann D, Turner R (1992) ”Animation based on the Interaction of L-systems with Vector Force Fields”, Proc. Computer Graphics International, in: Kunii TL (ed): Visual Computing, Springer, Tokyo, pp.747-761.

Noser H., Renault O., Thalmann D., Magnenat Thalmann N. (1995) ”Navigation for Digital Actors based on Synthetic Vision, Memory and Learning”, Computers and Graphics, Pergamon Press, Vol.19, No1, pp.7-19.

Noser H., Thalmann D. (1995) ”Synthetic Vision and Audition for Digital Actors”, Proc. Eurographics ‘95.

Renault O., Magnenat Thalmann N., Thalmann D. (1990) ”A Vision-based Approach to Behavioural Animation”, The Journal of Visualization and Computer Animation, Vol 1, No 1, pp 18-21.

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

Reynolds C.W. (1993) ”An Evolved, Vision-Based Behavioral Model of Coordinated Group Motion”, in: Meyer J.A. et al. (eds) From Animals to Animats, Proc. 2nd International Conf. on Simulation of Adaptive Behavior, MIT Press, pp.384-392.

Reynolds C.W. (1994) “An Evolved, Vision-Based Model of Obstacle Avoidance Behavior”, in: C.G. Langton (ed.), Artificial Life III, SFI Studies in the Sciences of Complexity, Proc. Vol. XVII, Addison-Wesley.

Ridsdale G. (1990) “Connectionist Modelling of Skill Dynamics”, Journal of Visualization and Computer Animation, Vol.1, No2, 1990, pp.66-72.

Rijpkema H, Girard M. (1991) ”Computer animation of knowledge-based human grasping”, Proceedings of Siggraph'91, pp.339-348.

Roth-Tabak Y. (1989) “Building an Environment Model Using Depth Information”, Computer, pp 85-90.

Sims K. (1994) “Evolving Virtual Creatures”, Proc. SIGGRAPH ‘94, pp. 15-22.

Slater M., Usoh (1994) ”Body centred Interaction in Immersive Virtual environments”, in: (Magnenat Thalmann and Thalmann, eds) Artificial Life and Virtual Reality, John Wiley, Chichester, 1994, pp.1-10

Sowa JF (1964) Conceptual Structures, Addison-Wesley.

Tsuji S, Li S. (1993) “Memorizing and Representing Route Scenes”, in: Meyer J.A. et al. (eds) From Animals to Animats, Proc. 2nd International Conf. on Simulation of Adaptive Behavior, MIT Press, pp.225-232.

Tu X., Terzopoulos D. (1994) ”Artificial Fishes: Physics, Locomotion, Perception, Behavior”, Proc. SIGGRAPH '94, Computer Graphics, pp.42-48.

Tu X., Terzopoulos D. (1994b) ”Perceptual Modeling for the Behavioral Animation of Fishes”, Proc. Pacific Graphics '94, World Scientific Publishers, Singapore, pp.165-178

Tyrell T. (1993) ”The Use of Hierarchies for Action Selection”, in: From Animals to Animats 2, Proceedings of the Second International Conference on Simulation of Adaptive Behavior, 1993, pp. 138-147.

van de Panne M., Fiume E. (1993) ”Sensor-Actuator Network”, Computer Graphics, Annual Conference Series, 1993, pp.335-342.

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