VLNET: a Virtual Life Network for telecooperative applications

Nadia Magnenat Thalmann[1], Tolga K. Capin[2]

Igor-Sunday Pandzic[1], Daniel Thalmann[2]

[1]MIRALAB-CUI
University of Geneva
24 rue de Général-Dufour
CHI 211 Geneva 4, Switzerland
{Nadia.Thalmann, Igor.Pandzic}@cul.unige.ch
http://cuisg13.unige.ch:8100/HomePage.html

[2]Computer Graphics Laboratory
Swiss Federal Institute of Technology
CH1015 Lausanne, Switzerland
{capin,thahnann}@lig.di.epfl.ch
http://ligwww.epfl.ch

Abstract

In order to enhance the sense of *presence* within networked Virtual Environments it is important to increase the quality of physical or social interaction of participants with each other and with the environment. We believe that this increase of quality can be achieved by realistic modeling and animation of the human body representing the participant, together with the natural behaviors of the objects in the environment. The representation of *self* as a realistic-looking virtual human body with natural movement helps participants not only to perceive each other and thus feel *together,* but also to interact with the environment in a straightforward and natural fashion. This property is further enhanced if the objects in the environment behave in a natural, expected way. In the Virtual Life Network (VLNET) system we use realistic human body representation together with a set of motor functions giving behaviors to actors and objects to reach the mentioned goals.

Keywords: networked virtual environments, virtual humans, virtual life, computer animation, multimedia

1. Introduction

In the past few years we have seen an increasing number of research efforts for building networked Virtual Environments, and solutions were proposed for building toolkits for communication in networked virtual worlds [Amselam95] [Carlsson93] [Macedonia95] [Singh95], and special-purpose applications [Maxfield95] [Stansfield95] [Gisi94] [Broll95].

Networked virtual environments share problems with single-user environments, but they also need to consider other factors. A virtual environment should give the users a feeling of *presence* within this environment providing better interaction and an intuitive interface. Presence requires that the participant become part of the environment, and interact with the environment using natural ways. The degree of presence is expected to increase with an increasing level of physical or social interaction with appropriate reactions from the environment. Therefore, a multi-user virtual environment system should provide efficient and accurate representation and interaction of the objects with realistic animation, as well as efficient communication management. A user feels a better degree of presence if other participants within the same environment believe that she is present and active in the same environment, and they show it. This property is likely to increase collaboration and interaction among participants within the VE. As the body movements help to show intentions and real actions more clearly, hence decreasing ambiguity in interaction, it is important to represent the participants by virtual human bodies in shared environments.

There has been similar research to represent virtual humans in. virtual environments [Granieri95] [Yoshida95]. In the VLNET (Virtual Life Network) system that we have developed [Capin95], we attempt to provide a more realistic representation through the use of motor functions, combined with interaction with the environment. The motor functions encompass more than inverse kinematics or displaying previously-recorded key frames, as they take into consideration other parameters specific to the motion. Also, they are based on fast heuristics.

Typically the VEs are created by bringing together different models, possibly with different scalings and even different formats. Unlike CAD models, these models lack any corresponding interaction information concerning other objects. This makes it difficult to manipulate the scene, as when, for example placing an object comfortably in the right location without it floating in air. Therefore, realistic goal-oriented methods have to be provided for animating the objects depending on the user input. We propose different classes of motor functions that can be combined for this problem. In this paper, we present these motor functions of the VLNET system, and we discuss the issues and problems in building body-centered networked VEs with environment interaction.

2. User Representation

To improve the interaction among the participants in a multi-user virtual environment we employ the full-body participant representation using virtual actors. The body of a virtual actor should realistically represent the real participant's body and the body animation should be naturally correlated to the user actions. However, realistic modeling and animation of virtual humans is not a straightforward task.

The user representation in VLNET is based on the HUMANOID articulated body model [Boulic95], At the core of this model there is a skeleton structure resembling the anatomical structure of a real skeleton. The skeleton structure consists of a 3D articulated hierarchy of joints, each with realistic limits of movement. It comprises 74 degrees of freedom without the hands, with an additional 30 degrees of freedom for each hand.

The body envelope (skin) is attached to the skeleton in the form of 16 deformable surfaces representing the body parts: head, pelvis, thorax, abdomen, left and right upper leg, lower leg, foot, upper arm, lower arm and hand. As the skeleton is animated, these surfaces follow the movement and are deformed appropriately at the seams to form a realistic-looking deformed body. The body can be represented in three levels of detail ranging from 2000 to 40000 triangular facets.

The high quality visual representation is only one step towards a believable body model; at least as important is the natural body movement corresponding to the user actions.

This could be best achieved by using a large number of sensors to track all degrees of freedom in the real body. However, this is generally not possible or not practical. Normally, only a few degrees of freedom will be tracked, and the rest has to be interpolated using the behavioral human animation knowledge and different motion generators.

Each user sees the virtual environment through the eyes of her body, and can control the movement of the body by various sensor devices (varying from spaceball and dataglove, to numerous sensors attached to body). In addition to her eye position, the user also has control of her virtual hand to interact with the environment (pick and reposition objects). We selected these two modes of control, as most conventional input devices sense position and orientation of the head (e.g. head-mounted displays) and the hand (e.g. dataglove).

In the VLNET system, we provide a set of motor functions that are responsible for different human motion: walking motor for *navigation,* and arm motor for *manipulation* of objects. These motor functions are more powerful than playing previously-recorded motions: they are based on approximations coming from biomechanical experiments, and they attempt to consider different parameters of the motion they are responsible for, in order to give parameterized motion (for example step length in walking as a function of velocity).

When the user navigates through the environment, the walking motor is used to perform a natural walking motion. The participant uses input devices (e.g. spaceball, dataglove with gesture interpretation) to update the eye position of the virtual actor. Based on this control, the incremental change of the eye position is computed and the rotation and velocity of the body center is estimated. The walking motor uses the instantaneous velocity to compute the length and duration of the walking cycle, from which it computes the joint angles of the body. The walking motor is based on the HUMANOID walking model [Boulic90], guided interactively by the user or automatically generated from the given trajectory. Figure I (see Appendix) shows an example of the walking motion in real time.

For object picking and the arm motion in general, the arm motor has to compute the joint angles of the arm based on the 6 degrees of freedom of the hand determined by user input. Figure 2 illustrates the complexity of the degrees of freedom of the joints in the arm. There are multiple solutions of joint angles reaching the same hand position, and the most realistic one has to be chosen. At the same time the joint constraints have to be taken into account. These considerations make the arm motor much more complicated then a simple inverse kinematics problem. For the arm motor we use the captured data obtained using sensors and stored into a precomputed table of arm joints. This table divides the normalized volume around the body into a discrete number of subvolumes (e.g. 4x4x4) and stores the mapping from subvolumes into joint angles of the right arm. Figure 3 (see Appendix) shows an example of arm motion produced by this mechanism.

The users can also select a posture for the upper body for different emotions: tiredness, happiness, paying attention, etc. Currently, the user explicitly selects one emotion, using commands similar to smileys that are used commonly in text messages to express different emotions. The emotion motor sets the body Joints at the vertebrae ending at the shoulders based on this input. There is a need to define an emotion motor function that automatically recognizes the appropriate motion using data sensed from the real user. This developed motor function is an introductory step to building an automatic emotion motor. Figure 4 (see Appendix) shows body postures for some example emotions set by this motor.

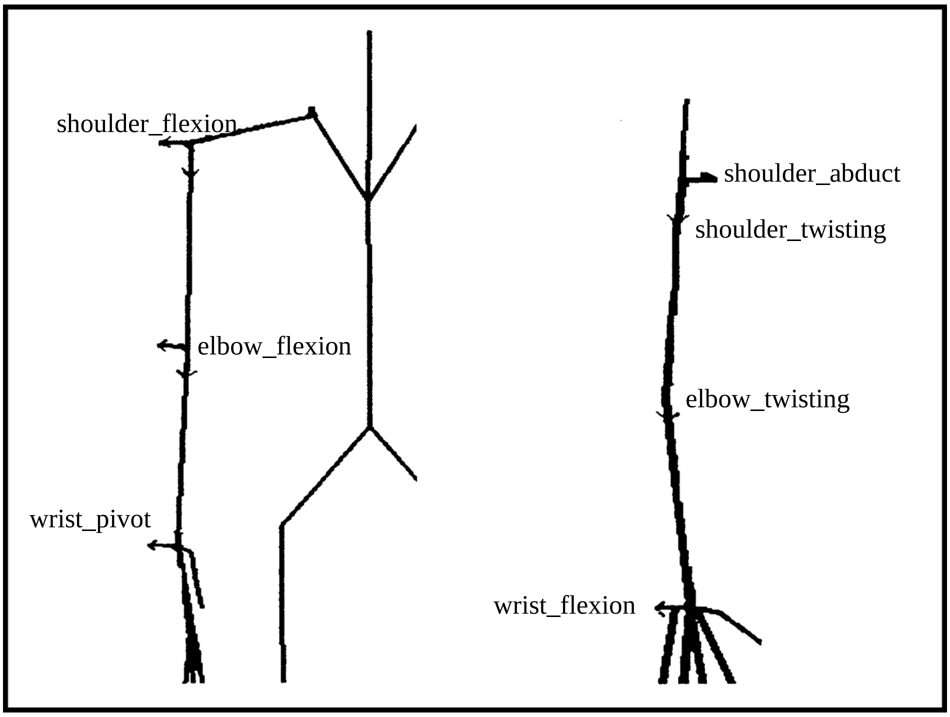

Figure 2: Degrees of Freedom for the Right Arm

Facial expressions are among the most important means of human communication, expressing intentions, thoughts and feelings. Therefore we include the facial communication in our multi-user virtual environment to enhance the communication between the users [Pandzic94].

We implement the facial communication by capturing the user's face using a camera and distributing it in real time to other users to be texture-mapped on the face of the virtual actor. Thus the virtual actor has the real moving face of the remote user.

The original images captured by the camera are first processed to extract the subset of the image containing the user's face. This processing is based on a comparison with an initial background image (the requirement is that the background is static). The extracted facial image is compressed at each frame and distributed to other users. At the receiving side, an additional service process is charged with the receipt and decompression of the images. The main application gets the decompressed images through shared memory from the service process, decoupling the facial video frame rate from the application frame rate.

The facial images are texture-mapped on a simplified model of a human head with attenuated features. This is a compromise between mapping on a simple shape (e.g. box, ellipsoid) which would give unnatural results and mapping on a full-featured human head model where more precise image feature alignment would be necessary. The texture mapping is illustrated in figure 5 (see Appendix).

The main drawback of the facial module is that it is not possible to incorporate it with users wearing HMD, as the face cannot be captured. However, it provides a good medium of interaction with shutter glasses or in the absence of these devices.

3. Interaction with Virtual Environment and Object Behaviors

It is expected that the participants feel a higher degree of presence if the environment reacts to their actions in a realistic way. For example, the user should be able to interact with the environment, reposition objects by picking them up with her virtual hand, and releasing them, making them fall. In order to pick up an object, the user moves her hand near the object and explicitly requests picking (e.g. by clicking spaceball button, closing **dataglove).** The objects stay picked until released explicitly by the user.

Typically the VEs are created by bringing together different models, possibly with different scalings and even different formats. Unlike CAD models, these models lack any corresponding interaction information between objects. This makes it difficult to manipulate the scene. A dynamics simulation with collision response would solve this problem. However for medium-sized environments this is a time-consuming solution, resulting in unwanted delays in the simulation. Therefore, we adopt a solution which compromises between realistic appearance and goal-oriented behaviors. We extend the view-dependent *object associations* framework proposed by Bukowski and Sequin [Bukoswki95] and we propose three classes of motor functions that can be attached to the objects, and include efficient communication schemes. We present the classification in this section, and network issues for executing these motor functions in multi-user VEs, in the next section.

A set of behaviors can be associated dynamically with any object in the environment. The object behaviors are implemented as different motor functions which give them a means of interacting with the users and the other objects. The types of motor functions can be divided into 3 classes:

- *continuous motor functions:* these functions require transformation update of the object regularly, within a specific period of time without any delay. For example, hands of a clock to show the time are in this category.

- *user-dependent motor functions:* these functions depend on the user input. This can be an explicit user input (for example, request for changing servers, see below); or implicit input (for example, automatic door behavior driven by position of the user).

- *environment-dependent motor functions:* these functions are dependent on the environment as well as the object itself We define different built-in motor functions corresponding to this category: magnet, vertical displacement, horizontal displacement, axis alignment. Magnet allows to attach different objects to each other with a predetermined transformation matrix (e.g. the watch body and bracelet are always attached with one trans format ion). Vertical displacement is called when the object is released; and is used for making the objects fall until it collides with an object, simulating gravity.

A subset of these behaviors can be added optionally to the objects during the scene creation. Different motor functions can be appended together to obtain more complex behaviors. For example, when an object is released; the vertical displacement motor is activated until a collision occurs with another object (e.g. table); after that the axis-alignment motor, that orients the object in a vertical position with respect to the collided object, becomes active.

A new motor function can be added only by programming. However, we have established a well defined step-by-step procedure for adding new motor functions, enabling a fairly experienced programmer to implement a new motor function easily and without knowledge of the rest of the application.

A motor function is attached dynamically to an object through a pointer to the motor function structure. This structure contains the necessary internal data of a particular motor function and the pointers to the subroutines to be executed in defined situations (e.g. the Update subroutine. is executed in each time step, the Save subroutine is activated when the user saves the scene configuration to a file). When the subroutines are executed, they get as a parameter the pointer to the object to which the motor function is attached. Using this mechanism the user can dynamically attach motor functions to objects, change their parameters or detach them.

4. Network Structure

The communication is based on a client/server model as illustrated in figure 6. The server is designed as a lightweight applications making it possible to run the server continuously in background on any host without putting a noticeable strain on the CPU. This actually provides permanent virtual worlds to which the VLNET clients can connect, each being a kind of virtual meeting place.

When the client establishes connections with the server, the server first provides the scene description to the new client, including all the object files necessary to build and visualize the virtual environment. All the other clients are informed that a new user entered the virtual world. The user representation information (body description, face) is exchanged between all the users, passing through the server. This insures that each user can provide his own body and face description and thus be recognized by others.

Once this initial information exchange is finished, all information exchange is done through the server using uniformly sized packets which are not more than the Maximum Transfer Unit of the protocol being used. The content of each packet is interpreted according to its type - new transformation of an object, body skeleton angles, grouping/ungrouping information, entry/exit messages etc. The packet is a data structure comprising a header which contains the message type and the sender id, and the body which is a union of data structures - one for each message type. All geometrical information is sent in absolute, rather then incremental values, insuring the coherence of the shared virtual environment even if a packet is lost.

When a user quits her VLNET session, the server cancels her from the client list and informs all other clients, thus insuring that this user disappears from the environment.

Links to other servers (i.e. other virtual worlds) can be established in a similar fashion as VRML or WWW links, These links can be attached to any object using a specialized motor function. When the user approaches such an object (it makes sense to make it look like a door), she is disconnected from the current server and connected to another one following the link. This gives the user the impression of "walking into another world". The user can take any objects with her when going to a different world. The linking mechanism, providing the possibility to carry objects through different worlds and allowing virtual actors to walk freely through the worlds, actually provides a hyper-world consisting of multiple servers scattered across the network.

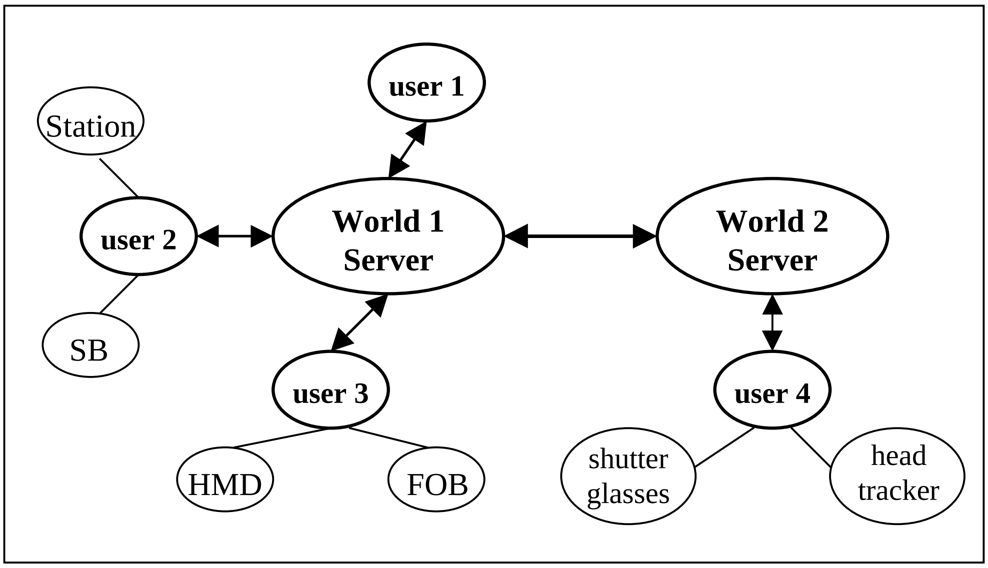

Figure 6: Communication Architecture of the VLNET System is based on a client/server model with links between the servers

The motor functions can be handled in different ways in the networked application.

The simplest mechanism is to execute the motor function locally on each host. This is appropriate for motor functions that do not put much strain on the CPU and where there is no danger of loosing the coherence of the shared environment. Time dependent motor functions are generally handled in this way.

The second way to handle the motor functions in a networked environment is to execute the function only on the host on which it has been triggered, and distribute the object position updates to other hosts. The standard communication packets are used for this distribution. This approach has two advantages: it distributes the processing, which is convenient for the more power-consuming motor functions, and it guarantees the coherence of the shared environment. This approach is used in general by the user- and environment-dependent motor functions (although some of them can be handled by the first, simple approach).

An extension of this second approach is provided for the motor functions that need to communicate some function-specific data. To this end a general-purpose communication packet can be used by the motor function. As an example of this approach, we have implemented a virtual slide show. When a user changes the slide, the slide show motor function distributes the slide number to other hosts, insuring that everybody sees the same slide.

5. Results

We have tested the VLNET system over the ATM network between Geneva and Singapore, provided during the Telecom'95 exhibition in Geneva. Our results showed that the ATM network is suitable for guaranteeing quality of service for small-sized packets between the server and the clients. We will make another test with larger numbers of users located in Geneva, Belgium, and Singapore.

We have built experimental worlds for different applications such as teleshopping, game-playing and medical education, and have made tests between multiple users located in Switzerland and Japan over the Internet network and Swiss ATM Pilot network. Snapshots from some of these sessions are presented in figure 7 (see Appendix).

6. Conclusion and future work

In this paper we have presented the VLNET system which provides a shared environment with virtual humans and their interactions. The motor functions provide powerful and efficient tools for increasing realism of body-centered interactions. In addition, they allow parallel animation of objects in the multiple-user VEs, improving the speed of interaction.

Future work remains for including deformable objects in the shared environment. There is also a need to build an emotion motor that automatically recognizes the emotion of the real participant, and updates the body realistically corresponding to this emotion. Currently the motor functions are coded in software. However further research will continue building general motor functions by combining low-level motors.

Acknowledgments

The research was partly supported by ESPRIT project HUMANOID (P 6079), Swiss National Foundation for Scientific Research, Silicon Graphics, Federal Office of Education and Science, and Department of Economy of City of Geneva. We would like to thank assistants of LIG and MIRALAB for the models and libraries.

References

[Amselam95] Amselem D., "A Window on Shared Virtual Environments", Presence: Teleoperators and Virtual Environments, Vol. 4, No. 2, 1995.

[Boulic95] Boulic R., Capin T., Huang Z., Kalra P., Lintermann B., Magnenat-Thalmann N., Moccozet L., Molet T., Pandzic L, Saar K., Schmitt A., Shen J., Thalmann D., "The Humanoid Environment for Interactive Animation of Multiple Deformable Human Characters", *Proceedings of Eurographics '95, 1995.*

[Boulic90] Boulic R., Magnenat-Thalmann N. M.,Thalmann D. "A Global Human Walking Model with Real Time Kinematic Personification", *The Visual Computer,* Vol.6(6),1990.

[Broll95] Broll W., "Interacting in Distributed Collaborative Virtual Environments", *Proceedings of IEEE VRAIS'95, 1995.*

[Bukowski95] Bukowski R.W., Sequin C.H., "Object Associations: A Simple and Practical Approach to Virtual 3D Manipulation", *Proceedings of ACM Symposium on Interactive 3D Graphics,* Monterey, California, *1995.*

[Capin95] Capin T.K., Pandzic I.S., Magnenat-Thalmann N., Thalmann, D., "Virtual Humans for Representing Participants in Immersive Virtual Environments", *Proceedings of FIVE '95,* London, *1995* (to appear).

[Carlsson93] Carlsson C., Hagsand 0., "DIVE - a Multi-User Virtual Reality System", *Proceedings of IEEE VRAIS '93,* Seattle, Washington, *1993.*

[Gisi94] Gisi M.A., Sacchi C., "Co-CAD: A Collaborative Mechanical CAD System", *Presence: Teleoperators and Virtual Environments, Vol. 3, No. 4, 1994.*

[Gobbetti93] Gobbetti E., Balaguer J.F., Thalmann D., "VB2: An Architecture for Interaction in Synthetic Worlds", *Proceedings ofACM UIST93,* Atlanta, *1993.*

[Granieri95] Granieri J.P., Becket W., Reich B.D., Crabtree J., Badler N.I., "Behavioral Control for Real-Time Simulated Human Agents", *Proceedings of ACM Symposium on Interactive 3D Graphics,* Monterey, California, *1995.*

[Macedonia94] Macedonia M.R., Zyda M.J., Pratt D.R., Barham P.T., Zestwitz, "NPSNET: A Network Software Architecture for Large-Scale Virtual Environments", *Presence: Teleoperators and Virtual Environments, Vol. 3, No. 4, 1994.*

[Maxfield95] Maxfield J., Fernando T., Dew P., "A Distributed Virtual Environment for Concurrent Engineering", *Proceedings of IEEE VRAIS '95, 1995.*

[Pandzic94] Pandzic I.S., Kalra P., Magnenat-Thatmann N., Thalmann D., "Real-Time Facial Interaction", *Displays, Vol 15,* No *3, 1994.*

[Singh95] Singh G., Serra L., Png W., Wong A., Ng H., "BrickNet: Sharing Object Behaviors on the Net", *Proceedings of IEEE VWS '95, 1995.*

[Stansfield95] Stansfield S., 1,vfiner N., Shawver D., Rogers D., "An Application of Shared Virtual Reality in Situational Training", *Proceedings of IEEE VRAIS '95, 1995.*

[Yoshida95] Yoshida M., Tijerino Y., Abe S., Kishino F., "A Virtual Space Teleconferencing System that Supports Intuitive Interaction for Creative and Cooperative *Work", Proceedings of A CM Symposium on Interactive 3D Graphics,* Monterey, California, *1995.*

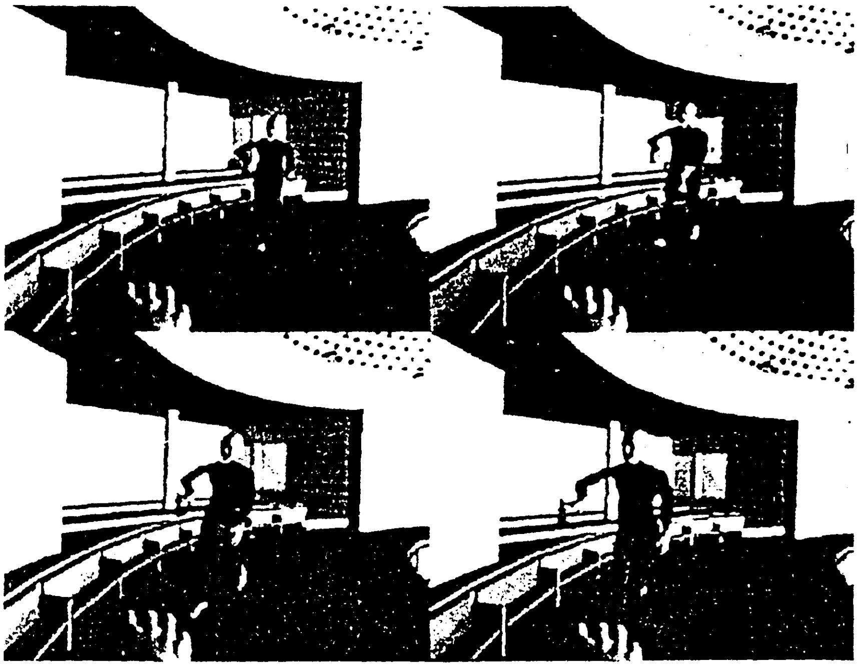

Figure 1: An example of real-time walking sequence

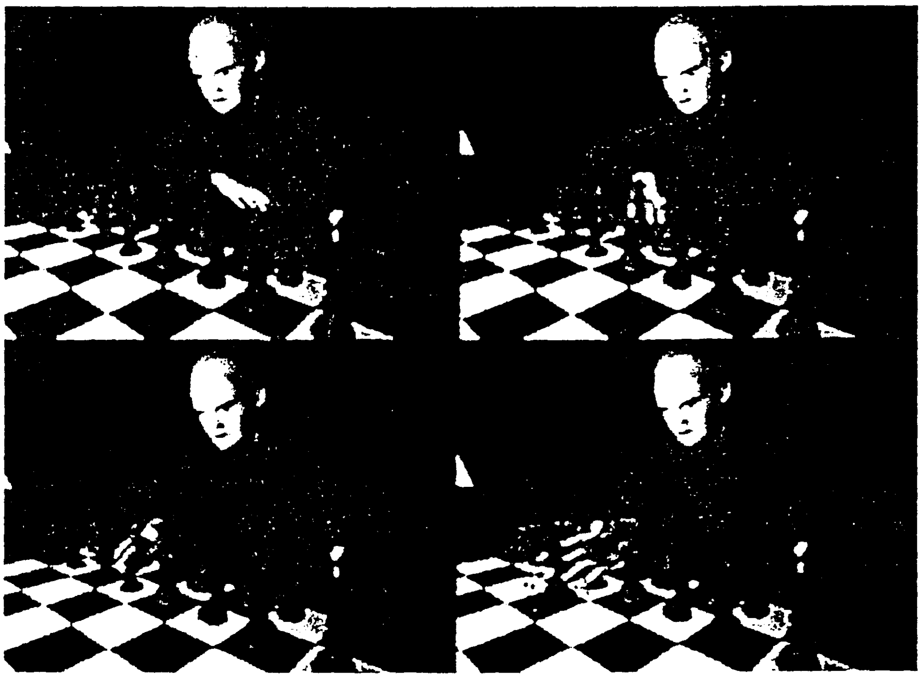

Figure 3: An example real-time grasping sequence

Figure 4: Emotion Motor updates the joints on the upper part of the body depending on the user's input. Some of the possible emotion representations: a) paying attention, b) tired, c) surprised.

Figure 5: Mapping of the face to the 3D virtual actor. Usage of simple head provides a compromise between 3D geometry and texture quality.

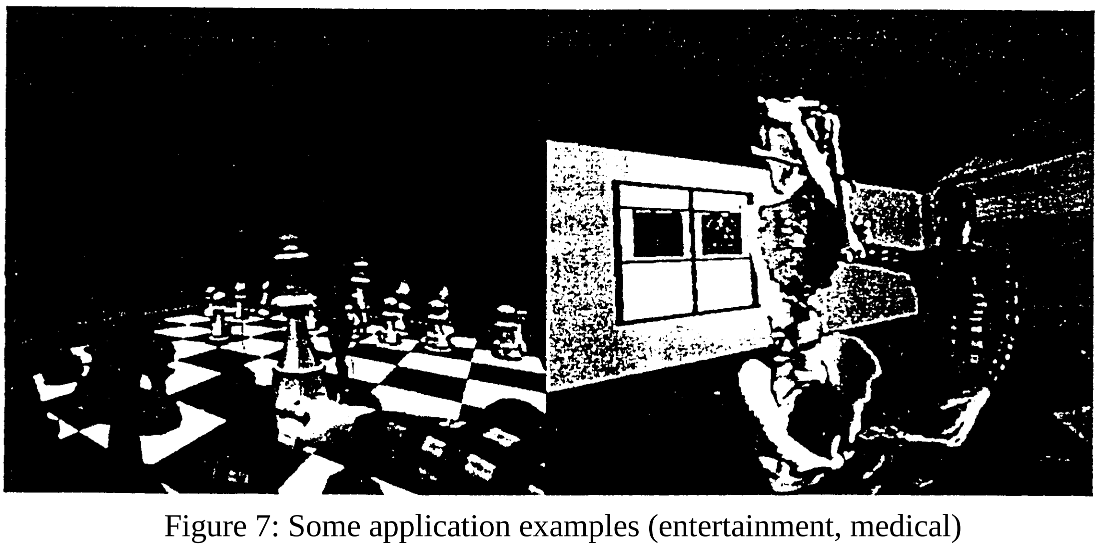

Figure 7: Some application examples (entertainment, medical)