Creating and Presenting Real and Artificial Visual Stimuli for the Neurophysiological Investigation of the Observation/Execution Matching System

Marco Agus, Fabio Bettio, and Enrico Gobbetti

CRS4
Center for Advanced Studies, Research, and Development in Sardinia
VI Strada Ovest, Z.I. Macchiareddu, C.P. 94, I-09010 Uta (CA), Italy
E-mail: {magus,fabio,gobbetti}@crs4.it,
WWW: http://www.crs4.it/vvr

Abstract. Recent neurophysiological experiments have shown that the visual stimuli that trigger a particular kind of neurons located in the ventral premotor cortex of monkeys and humans are very selective. These *mirror neurons* are activated when the hand of another individual interacts with an objects but are not activated when the actions, identical in purpose, are made by manipulated mechanical tools. A Human Frontiers Science Program project is investigating which are the parameters of the external stimuli that mirror neurons visually extract and match on their movement related activity. The planned neurophysiological experiments will require the presentation of digital stimuli of different kinds, including video sequences showing meaningful actions made by human hands, synthetic reproductions of the same actions made by realistic virtual hands, as well as variations of the same actions by controlled modifications of hand geometry and/or action kinematics. This paper presents the specialized animation system we have developed for the project.

1 Introduction

In spite of its fundamental role for human and animal behavior, very little is known on how individuals recognize actions performed by others. It has been often proposed that to recognize motor actions a common code should exist between the observed events and an internally produced motor activity. Neurophysiological evidence in favor of this putative common code was, however, until recently lacking. A possible breakthrough in this field was the discovery in the monkey of a set of neurons with surprising properties. These neurons, that are located in the ventral premotor cortex, discharge both when the monkey makes a hand action directed towards an object and when it observes another individual making a similar action ("mirror neurons"). In many instances the similarity between the active action and the observed action that activates the neurons is very tight. An action made by others is represented, therefore, in the premotor cortex. If one admits that an individual, when making an action, may predict its outcome, it appears likely that he/she will recognize an action made by others because it evokes a discharge in those same neurons that fire when he/she makes the identical action.

In the context of an international project supported by the Human Frontiers Science Program, CRS4 is developing a computerized system to support investigation of the mirror system (i.e. an observation-execution matching system) in humans, including the role of mirror neurons in creating a link between individuals.

Previous experiments have shown that the visual stimuli that trigger mirror neurons are very selective. These neurons are activated when the hand of another individual interacts with an objects but are not activated when the actions, identical in purpose, are made by tools. One of the goal of the project is to investigate which are the parameters of the external stimuli that mirror neurons visually extract and match on their movement related activity. The planned neurophysiological experiments will require the presentation of visual stimuli of different types, including video sequences showing meaningful actions made by human hands, synthetic reproductions of the same actions made by realistic virtual hands, as well as variations of the same actions by controlled modifications of hand geometry and/or action kinematics.

To simplify the creation and handling of a catalog of digitized stimuli to be used in the experiments, we are creating a specialized animation system that enables animators to create short animation sequences which closely follow a video-taped action, and to later modify the sequences to obtain all the required variations. This report briefly describes the current system prototype.

The rest of the paper is organized as follows. Section 2 provides a general overview of the system, section 3 concentrates on video acquisition and playback, section 4 details the video analysis and animation synthesis subsystem. The paper concludes with a discussion of the results obtained and a view of future work.

2 System Overview

In order to study the encoding of visual stimulus parameters for hand action recognition in monkeys, single neurons will be recorded from a sector of the premotor cortex (F5) and, subsequently, from inferior parietal lobe of monkeys trained to fixate a spot light on a computer screen and to detect a dimming of it. During fixation, digitized stimuli will be presented on the same screen showing hand actions performed in 3D space.

Two different kinds of stimuli are important for the planned experiments:

1. **Direct reproductions of "real" sequences**, without manipulation;
2. **Elaborations of the "real" sequences**, both in terms of modification of visual appearance and/or behavior (i.e. same action with variations in the kinematics and/or in the geometry and material of the performing object).

The first situation is handled by components that support recording, storage, and playback of stereoscopic movies, while the second requires components that create artificial sequences starting from the acquired stereo movies. Since the type of operations required (variations in kinematics and/or hand shape) are not easily obtainable by image manipulation, the system needs to work directly in 3D space. In our approach, the original sequence is first reconstructed by animating the degrees of freedom (DOFs) of a virtual 3D hand model. The resulting 3D animation, interactively constructed exploiting a combination of artificial vision and standard key-framing techniques, is then refined, modified, and rendered to produce all the desired variations.

Figure 1 provides a general overview of the system, depicting the main data processing components and the flow of data among them.

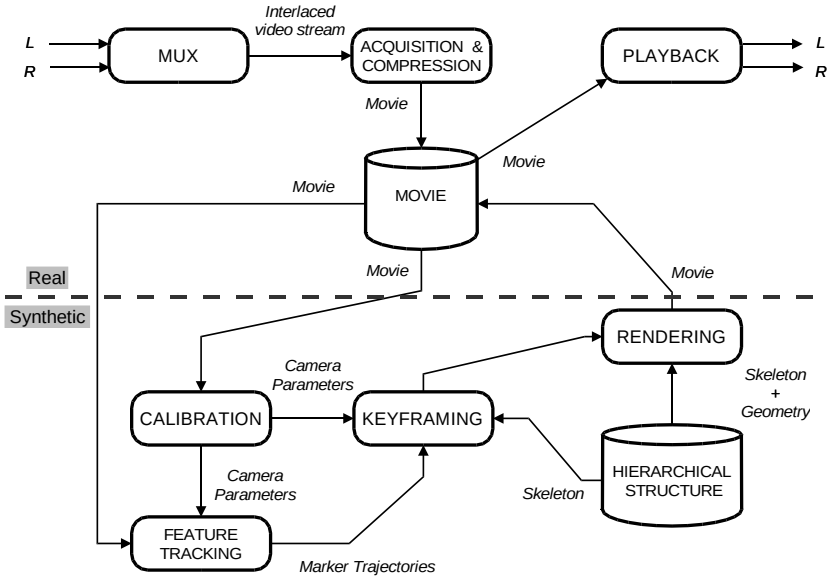

Fig. 1. System overview. Main data processing components and flow of data among them.

3 Stereoscopic Video Acquisition and Video Playback

3.1 Acquisition

This part of the system, built using off-the-shelf components, deals with the procedures for the acquisition of a stereoscopic video and its storage on a disk. The selected technical solution uses two video cameras placed on a tripod - and connected through a special metal band - so that the same scene will be recorded from two different points of view. The two inputs are synchronized and multiplexed in a single PAL or NTSC stream, which is then fed to the graphics workstation for recording or real-time display (see figure 2). A digital compression card is used for real-time signal compression, enabling direct-to-disk recording of sequences. Using M-JPEG compression, a typical stereoscopic sequence of 10 s requires about 20 Mb of storage (PAL format, 50 fields per second, 768x288 pixels per eye).

We have successfully run our applications on a Silicon Graphics IMPACT connected to a VREX Cam3C system and on a Silicon Graphics Octane connected to a TD003 3D Multiplexer with two CCD cameras for input.

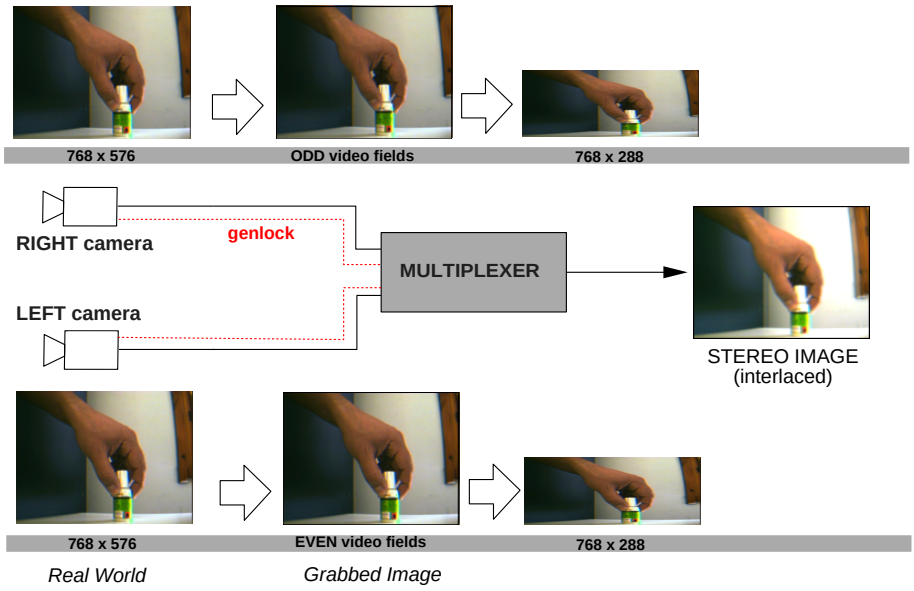

Fig. 2. Stereo acquisition. The two inputs are synchronized and multiplexed in a single PAL or NTSC stream, which is then fed to the graphics workstation for recording or real-time display.

3.2 Playback

This part of the system deals with the decoding of compressed stereoscopic movies and the stereo visualization of the video acquired. In order to reach high-level performances for stereo visualization, we have decided to completely load and decompress video sequences in memory prior to visualization. During visualization, left and right images are copied to the frame buffer in the format needed for time-multiplexed stereo presentation and the graphics card is suitably configured. Shutter glasses are used for image presentation. Both stereo-in-a-window and full-screen presentations are possible. Once decompressed, a typical stereoscopic movie of 10 s requires around 440 MB of memory. Real-time image zooming during display, combined with image resizing at load time, allows users to trade image resolution with sequence length.

4 Video Analysis and Animation Synthesis

The video analysis and animation synthesis subsystem provides tools for creating animations of articulated 3D models that closely follow the actions of a real hand depicted in a stereoscopic movie.

In order to reach this goal, we have combined in a single interactive environment the basic features of artificial vision systems, to extract meaningful information from the stereo movie, with the concept of key-framed animation. In our approach, the original sequence is reconstructed by estimating the projection and viewing matrices of the cameras, tracking the 3D trajectory of interesting hand features, and animating the degrees of freedom of a virtual 3D hand model so as to obtain the best match with the recorded movie. The resulting 3D animation is then refined, modified, and rendered using standard editing features to produce all the desired variations.

The system has been implemented on a PC platform running Windows NT and requires consumer-level graphics boards (e.g. NVIDIA GeForce based systems).

4.1 Camera Registration

The camera transformation matrices contain all the geometric information (intrinsic and extrinsic camera parameters) which is necessary to calculate 3D coordinates from correspondences between two stereo perspective images of a scene.

Since a metric reconstruction of the environment is necessary, we have decided to implement a strong calibration system that estimates camera parameters by observing a calibration object whose geometry is known.

We currently use a semi-automatic approach in which the user selects a set of at least seven matching points on a camera view and associates them with their known 3D coordinates (see figure 3). The process is applied independently for the left and right camera. Figure 4 illustrates the results obtained.

Given a set of 2D points $\mathcal{U} = \{\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_n\}$ and a set of corresponding 3D points $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, the camera projection matrix $\mathbf{P}(\theta, x_s, y_s, x_c, y_c)$ and the camera viewing matrix $\mathbf{V}(r_x, r_y, r_z, t_x, t_y, t_z)$ are estimated by minimizing the following functional:

$$\sum_{i=1}^{n} \left\| \begin{bmatrix} \frac{\mathbf{t}_1 \cdot \mathbf{x}_i}{\mathbf{t}_3 \cdot \mathbf{x}_i} \\ \frac{\mathbf{t}_2 \cdot \mathbf{x}_i}{\mathbf{t}_3 \cdot \mathbf{x}_i} \end{bmatrix} - \mathbf{u}_i \right\|^2$$

where:

- $\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3$ are the rows of $\mathbf{T} = \mathbf{P}(\theta, x_s, y_s, x_c, y_c)\mathbf{V}(r_x, r_y, r_z, t_x, t_y, t_z)$;
- θ is the horizontal field of view;
- x_s, y_s are the dimensions of the image in pixels;
- x_c, y_c are the pixel coordinates of the center of projection;
- r_x, r_y, r_z are the Euler angles specifying the camera orientation;
- t_x, t_y, t_z are the coordinates of the camera position.

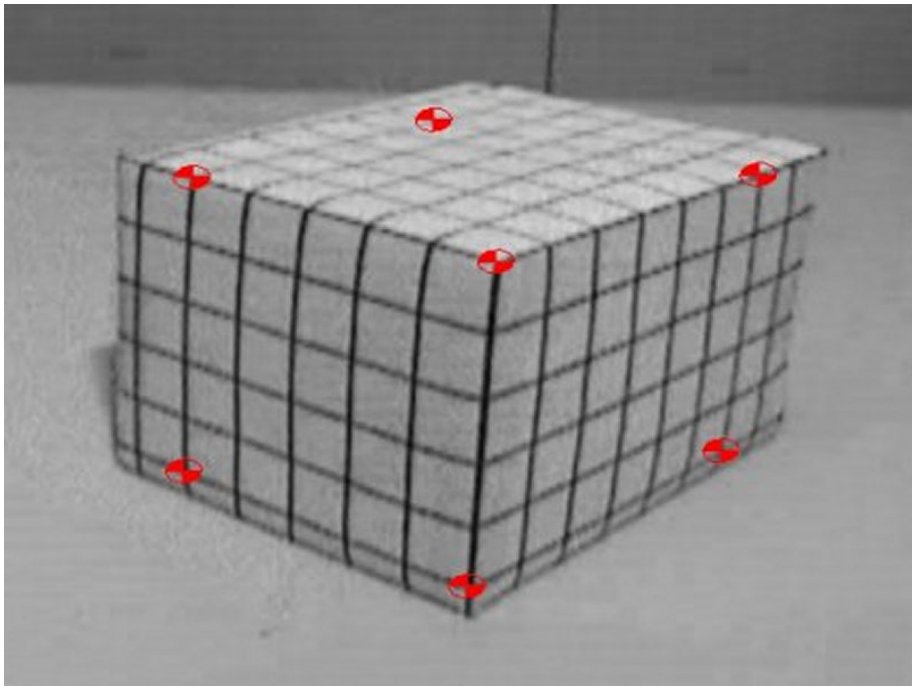

Fig. 3. Camera calibration. The camera viewing and projection matrices are determined from the correspondences between selected left and right calibration points and their 3D coordinates.

In our implementation, we use a gradient descent algorithm to search for the optimal solution. A series of different starting points for the viewing parameters are generated on the sphere containing the bounding box of the 3D data set. The minimization algorithm is applied for each starting point and the minimum error solution is chosen as the optimal one.

4.2 Feature Tracking

3D reconstruction Once the camera transformation matrices are known, the system has all the information to calculate the 3D coordinates of a point from point matches on the left and right images. This is used to help users track interesting features (e.g. finger tips) over time and use them to guide the keyframing process.

Reconstructing the 3D position of a point $\mathbf{x}$ from its two projections $\mathbf{u}^{(l)}$ and $\mathbf{u}^{(r)}$ requires the solution of an homogeneous system. Let $\mathbf{T}^{(l)}$ be the left camera transformation matrix and let $\mathbf{T}^{(r)}$ be the right camera transformation matrix. The 3D coordinates of the point $\mathbf{x}$ are the solution of the following equation:

$$\mathbf{A}\mathbf{x} = \mathbf{0} \tag{1}$$

where $\mathbf{A}$ is a 4x4 matrix given by:

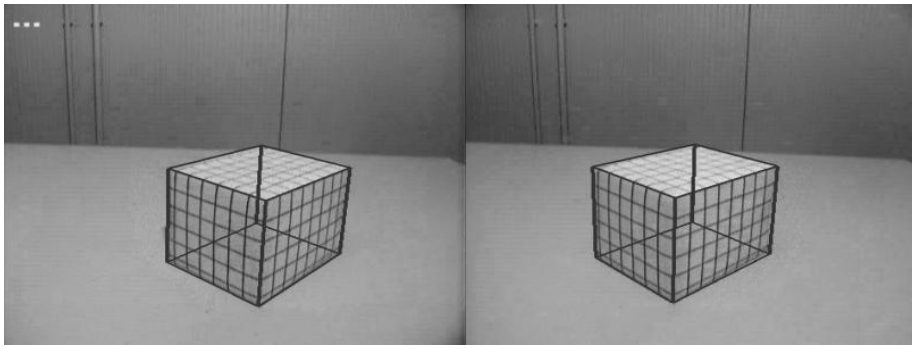

Fig. 4. Camera calibration test. A synthetic 3D box is overlayed over the captured images to check the accuracy of the calibration process.

$$\mathbf{A} = \begin{bmatrix} \mathbf{t}_1^{(l)} - u_1^{(l)}\mathbf{t}_3^{(l)} \\ \mathbf{t}_2^{(l)} - u_2^{(l)}\mathbf{t}_3^{(l)} \\ \mathbf{t}_1^{(r)} - u_1^{(r)}\mathbf{t}_3^{(r)} \\ \mathbf{t}_2^{(r)} - u_2^{(r)}\mathbf{t}_3^{(r)} \end{bmatrix} \tag{2}$$

This problem is solved using an iterative linear least-squares method [8].

Tracking features over time To construct the 3D trajectory of a feature during the animation, the user determines its position in a small set of interesting frames and lets the system build an interpolating 3D spline that provides the position of the feature as a function of time (see figure 5).

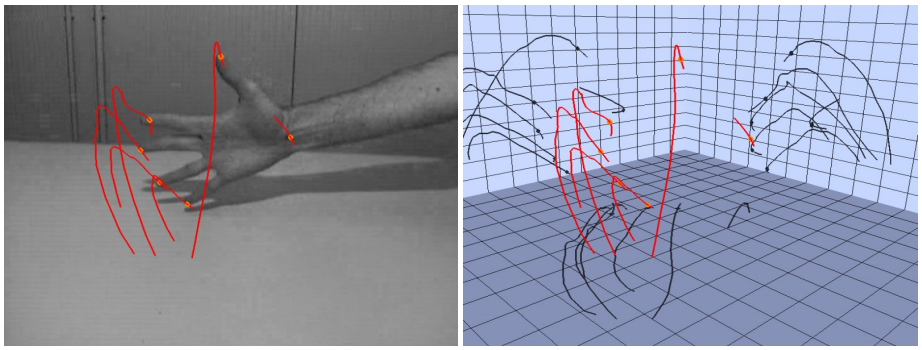

Fig. 5. Tracking features over time. Lines represent the trajectories of selected features for the duration of the movie. Dots represent the position of the markers in this particular frame. On the left image the current movie frame is on the background, while on the right image the 3D trajectories are projected on three orthogonal planes.

4.3 Virtual Hand Positioning and Key-Framing

The key-framing animation system aims at generating synthetic animations by positioning a virtual hand model, specifying the joint values (DOFs) for some time values and interpolating for the others. The appearance and structure of the virtual hand are fixed for a single animation sequence. The model is encoded using a VRML node graph [3]. The structure of the standard hand used for creating the reference animation that matches the video recorded one is based on the one described in the MPEG4 Version 2 (PDAM1) specification [1] (see figure 6). The system, however, is not restricted to this model and is able to animate objects with arbitrary structure.

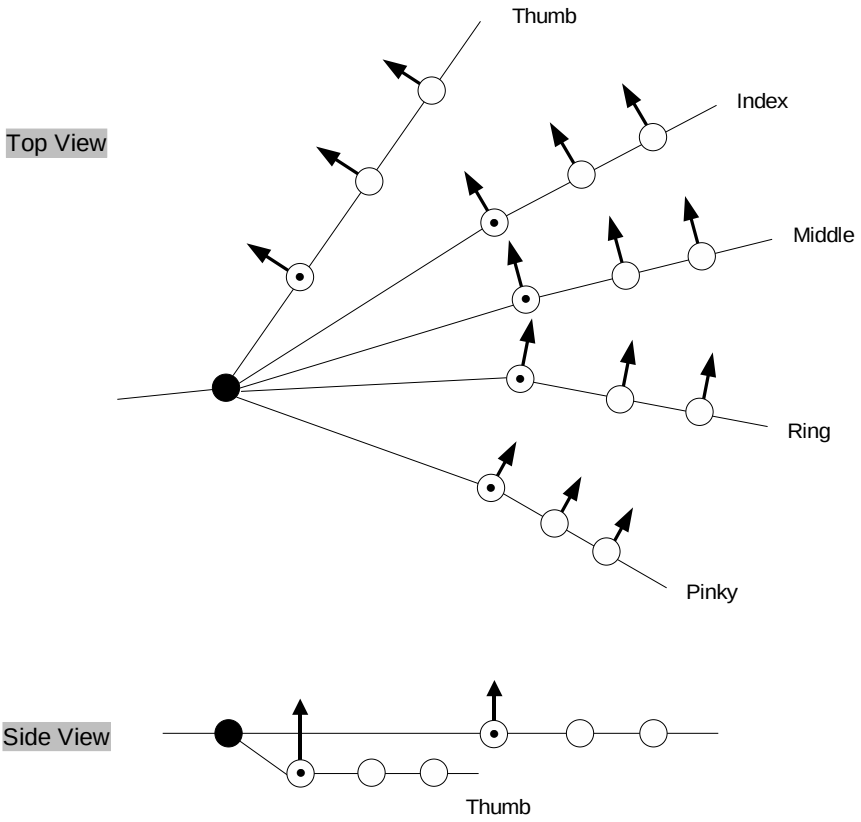

Fig. 6. Standard right hand structure. Top and side view of the mobilities of the right hand in rest position. The model has 20 DOFs associated to finger joints and 6 DOFs associated to the wrist.

In our system, the model position at a key frame is specified with a 6DOF input device (Magellan SpaceMouse), and it is also possible to specify each joint angle using both direct and inverse techniques.

Direct specification of key joint values A number of visual elements guide the animator in specifying the DOF values so as to obtain an animation as similar as possible to the original grasping sequence. These are the following:

- the **original movie** in background, so that the animator may compare the virtual hand pose with the original pose in the movie;
- the **trajectories and markers** obtained from movie, that specify the goal position of interesting features (e.g. finger tips);
- other **auxiliary visual elements**, such as orthogonal projections on a box, and segments that join virtual finger tips to marker goals.

Directly determining all DOF values using this direct technique requires a great deal of input from the animator.

Inverse specification of key joint values using constrained manipulation This subsystem takes as input the marker trajectories (e.g. the tip positions of the hand) and computes the optimal DOF values that permit to reach these goal positions, helping the animator in modeling the key-frames and obtaining more realistic animations.

The approach that we used to reach this goal is to let animators freely manipulate the root position using the 6DOF input device, while the system concurrently selects the optimal joint values by minimizing a cost function. This very general approach has proven very helpful in a number of animation contexts [2]. Reducing the number of degrees of freedom under direct animator control in the generation of hand animations is also motivated by the fact that the mechanical structure of the hand introduces constraints that reduce the ability to control hand joints independently (see [6] for a task-level computer animation application).

In our case, we use the following cost function:

$$Cost(\varphi) = \sum_i \|P^i_{tip}(\varphi) - P^i_{goal}\|^2 + \sum_j barrier(\varphi_j) + \sum_j \frac{1}{\sigma_j^2}\|\varphi_j - \hat{\varphi}_j\|^2 \quad (3)$$

where:

- $\varphi = \{\varphi_1, \varphi_2, ..., \varphi_n\}$ is the joint value vector to determinate;
- P^i_{goal} is the goal position for the i-th finger tip;
- $P^i_{tip}(\varphi)$ is the current position of the i-th finger tip, function of DOF vector φ.

The cost function in equation 3 contains various components. The first component drives the tips of the hand fingers (or other selected points in the local reference frame of the hand) towards the associated goal positions (markers). The second component implements joint value constraints by summing the penalty functions associated to joint limits for each of the joints. The limit function increases as joint angle φ_j approaches either its maximum or minimum limit and is zero otherwise. This strongly inhibits each joint from bending beyond its prescribed limits. One such function which enforces joint limits is the following:

$$barrier(\varphi_j) = \begin{cases} -\frac{\varphi_j - (\varphi_j^{\min} + \alpha)}{\alpha} + \ln\left(\frac{\delta}{\alpha}\right), & \varphi_j < \varphi_j^{\min} + \alpha \\ \ln\left(\frac{\delta}{\varphi_j - \varphi_j^{\min}}\right), & \varphi_j^{\min} + \alpha < \varphi_j < \varphi_j^{\min} + \delta \\ 0, & \varphi_j^{\min} + \delta < \varphi_j < \varphi_j^{\max} - \delta \\ \ln\left(\frac{\delta}{\varphi_j^{\max} - \varphi_j}\right), & \varphi_j^{\max} - \delta < \varphi_j < \varphi_j^{\max} - \alpha \\ \frac{\varphi_j - (\varphi_j^{\max} - \alpha)}{\alpha} + \ln\left(\frac{\delta}{\alpha}\right), & \varphi_j > \varphi_j^{\max} - \alpha \end{cases} \quad (4)$$

where φ_j is the angle of the current j-th joint , δ is the angular distance from the limits at which the limit function becomes non-zero, and α is the angular distance at which the logarithmic barrier is substituted with a linear barrier for numerical reasons. The third component drives each joint value φ_j towards a rest position $\hat{\varphi}_j$ and is weighted by a sensitivity parameter σ_j. This parameter approximates how much the j-th joint variations influences the correspondent tip position changes.

A gradient descent algorithm is used to modify the joint angles of the model in order to drive the equation 3 to a minimum. The algorithm used is adaptive in the sense that computes each descent steps using golden search bracketing and line search algorithms [5]. Figure 7 illustrates key-frame editing using constrained manipulation.

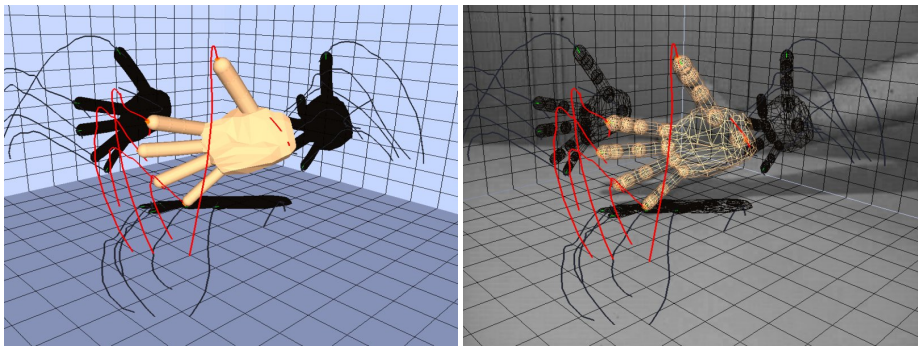

Fig. 7. Constrained manipulation. The optimization algorithm computes optimal values for joint angles, while the user controls the hand wrist using the SpaceMouse. In the left image, orthogonal projections are used to guide interactive positioning, while in the right image the movie frame is also displayed in the background.

Direct and inverse (constrained) techniques are usually iteratively applied until the animator judges the matching between the pose and the original movie sufficiently precise. Once the desired pose is found, all the joint values are stored in the key-framing system so as to participate as control points in the animation. This is repeated for a number of key-frames, until the animator is satisfied with the resulting animation.

4.4 Playback

A simple direct playback system offering the same functionalities as a VCR lets the animator check the animation effectiveness and compare it with the original movie.

4.5 Animation Editing

In order to get meaningful modifications of the grasping sequences, the system makes it possible to interactively manipulate the animation. These manipulations involve:

- **Geometry and Appearance Modification:** the system is able to apply the same animation data to various models, that differ in geometry and appearance.
- **Time warping:** the system provides tools that enables the animator to modify the animations by applying different time functions to joint angles.

Currently, geometry and appearance modification is obtained by re-using the same animation tracks with different articulated structures (i.e. different VRML files) containing the same degrees of freedom, while time warping features are limited to global scaling. More sophisticated motion retargetting techniques (e.g. those presented in [7, 4]) will be explored in the future.

5 Animation Results

A number of movies are being produced to be used as visual stimuli. Figure 8 shows selected frames of three different versions of a grasping sequence.

6 Conclusions and Future Work

We have presented a specialized animation system that enables animators to create short animation sequences that closely follow a video-taped action, and modify the sequences to obtain meaningful variations. The system contains tools that cover the following functionality:

- record and play stereo grasping sequences;
- reconstruct the projection and viewing matrices of left and right camera from stereo movies;
- track the trajectory of interesting features inside stereo movies;
- generate complete animations of virtual models by key-frame editing and interactive positioning;
- semi-automatically generate key-frames by constrained manipulation of the virtual model;
- manipulate animations in the sense of geometry modifications and motion warping.

All of these features are currently implemented, and the first movies to be used as digital stimuli in the neurophysiological experiments are in the production phase. The future work will concentrate on improving the animation manipulation and motion warping features, and in producing higher-quality renderings.

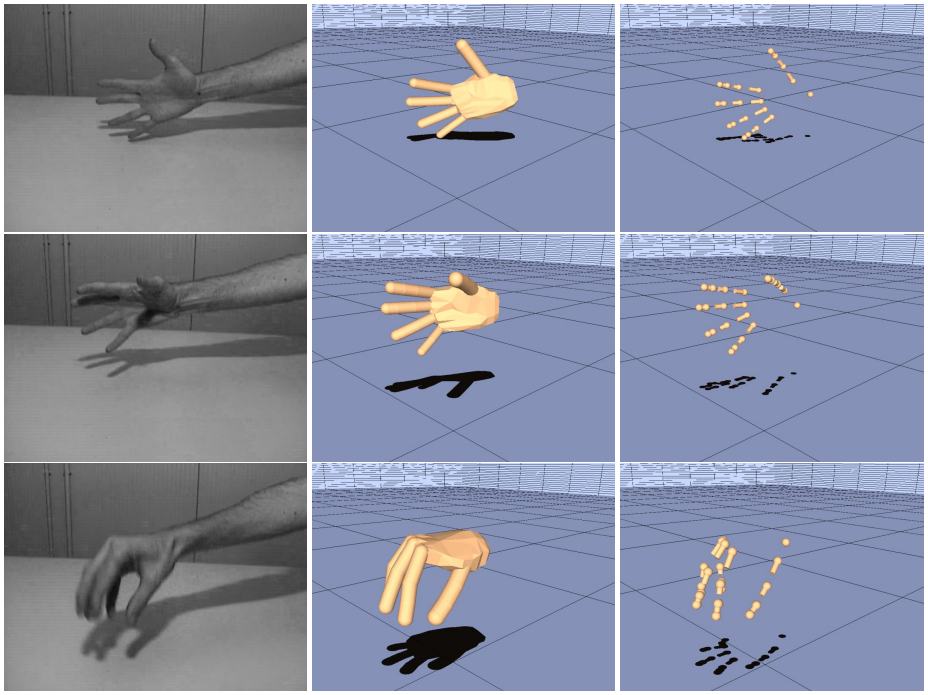

Fig. 8. Animation results. The left images show selected frames of a video-taped grasping sequence. The middle images shows a synthesized animation replicating the same motion. The right images shows the same animation applied to a different geometric structure.

Acknowledgments

This work is carried out within the research project “MIRRORS: Mirror Neurons and the Execution/Matching System in Humans”, sponsored by the Human Frontiers Science Program Foundation. The project partners are: Istituto di Fisiologia Umana, Università di Parma, Italy (Coordinator); CRS4, Cagliari, Italy; Brain Research Unit, Low Temperature Lab. Helsinki University of Technology, Espoo, Finland; Dept. of Behavioral and Brain Sciences, Primate Research Institute, Kyoto University, Inuyama, Japan; Dept. of Basic Sciences, Lab. of Functional Brain Imaging, University of Crete Medical School and IACM FORTH, Heraklion, Greece; Dept. of Neurology, University of California School of Medicine, Los Angeles, USA. We also acknowledge the contribution of Sardinian regional authorities.

References

1. ISO/IEC 14496-1: MPEG-4 PDAM1. Available on the web at the address http://www.cslet.stet.it/mpeg.
2. David E. Breen. Cost minimization for animated geometric models in computer graphics. *The Journal of Visualization and Computer Animation*, 8(4):201–220, 1997. ISSN 1049-9807.

3. Rikk Carey and Gavin Bell. *The VRML 2.0 Annotated Reference Manual*. Addison-Wesley, Reading, MA, USA, January 1997. Includes CD-ROM.
4. Kwangjin Choi and Hyeongseok Ko. On-line motion retargetting. In *Proceedings of the International Pacific Graphics '99*, 1999.
5. W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, UK, second edition, 1992.
6. Hans Rijpkema and Michael Girard. Computer animation of knowledge-based human grasping. *Computer Graphics*, 25(4):339–348, July 1991.
7. Andrew Witkin and Zoran Popović. Motion warping. *Computer Graphics*, 29(Annual Conference Series):105–108, November 1995.
8. Zhengyou Zhang. Determining the epipolar geometry and its uncertainty: A review. Technical Report 2927, Institut National de Recherche en Informatique et en Automatique, Sophia-Antipolis Cedex, France, July 1996.