vHand: A Human Hand Simulation System

Beifang Yi Frederick C. Harris, Jr. Sergiu M. Dascalu

Department of Computer Science & Engineering
University of Nevada, Reno
Reno, NV 89557
{b_yi, fredh, dascalus}@cse.unr.edu

Abstract

This paper introduces a real time human hand simulation system. A lifelike hand model is constructed and some of the human hand constraints are applied to it. Natural hand gestures and animations can be generated rapidly. This is accomplished by interfacing the hand model with the implemented built-in hand gesture and animation data structures and combining all the operations in a graphical user interface. This system has provided ground truth data for research on human hand gesture analysis and recognition. It can also be incorporated into virtual human body to generate body gestures and to convey some American Sign Language signals.

Keywords: Hand modeling, constraints, gestures; User interface design; Animation; HCI; ASL.

1. Introduction

In human-computer interaction (HCI) applications, the human hand has been considered one of the most promising natural HCI media [12]. Vision-based HCI studies on hand gesture analysis and recognition require large amount of ground truth data (a variety of hand gestures) as input and a virtual hand as output for displaying results. The construction of a virtual hand and the creation of natural hand gestures would improve HCI research on the human hand. On the other side, in the study of American Sign Language (ASL) through virtual human body, a digital ASL interpreter would greatly enhance the communication between deaf and hearing people [8], and a lifelike virtual hand would be the principal element of a digital ASL interpreter.

In this paper, we introduce a real time human hand simulation system, *vHand*. First, a lifelike hand model is constructed and hand constraints are applied to it. Then, with design and implementation of a built-in hand gesture and animation data structure and combination of all the operations in a graphical user interface (GUI), natural hand gestures and animations can be generated rapidly.

There are sophisticated algorithms and implementations in hand modeling and animation: hand modeling with underlying anatomical structure [1], data driven algorithm in hand animation [4], and example-based (from medical images) deformable modeling [5]. Compared with these hand models, our virtual hand is simpler, with less deformation consideration, but looks still realistic with the consideration of human hand constraints. Moreover, the hand model is only part of the simulation system. The advantages of vHand include its simplicity in the modeling algorithm, time efficiency in rendering, the emphasis on the interactions between the hand model and its users, and a convenient GUI that makes all the operations on the finger movements very easily and effectively. Figure 1 presents a screen shot of vHand: the upper half is used for display areas and the lower half for control (here, for adjusting fingers' parameters).

2. Modeling the human hand

2.1. The human hand

Although the hand is a highly complex biological organ, we can think of the hand as a mechanical machine and benefit by applying mechanical principles to studies on the hand. In this view, the hand involves three elements: muscles serve as the motor for providing driving force, tendons and bones and joints transmit the force, and skin and pulp tissues apply the force [2].

Furthermore, the hand motion is often described with the movements of the hand bones. A linkage system of rigid segments of hand bones allows us to describe and analyze hand motions, with hand joints between hand bones as motion (rotation) points. Figure 2 provides a simplified illustration of the hand joints:

- Finger joints
  - DIP: Distal interphalangeal joint
  - PIP: Proximal interphalangeal joint

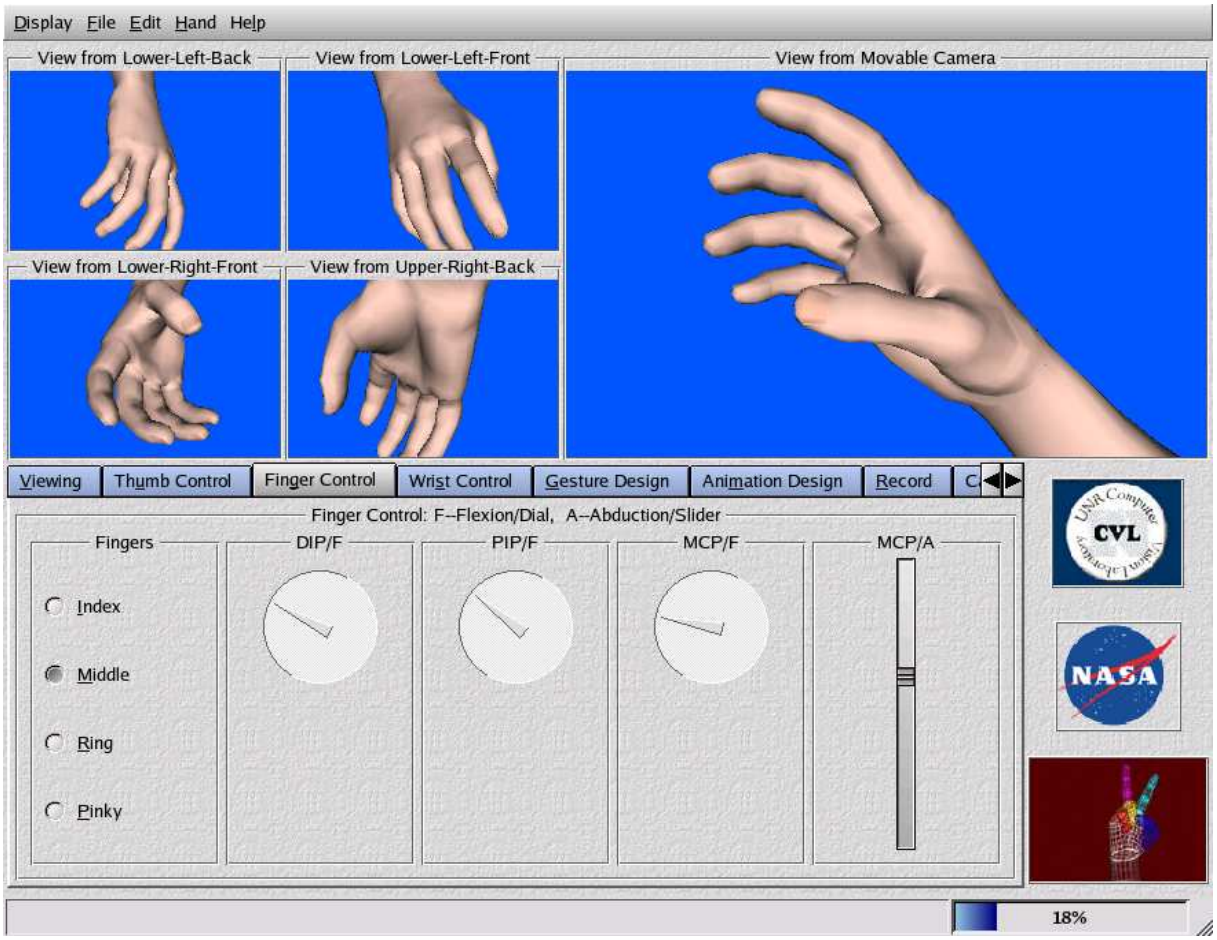


Figure 1: A screen shot of vHand.

  - MCP: Metacarpophalangeal joint

- Thumb joints
  - IP: Interphalangeal joint
  - MCP: Metacarpophalangeal joint
  - CMC: Carpometacarpal joint

Hand motions are complex combinations of the movements (rotations) of different bones at various hand joints. While medical researchers and biomechanical scientists have used sophisticated methods and measurements to describe and quantize hand motions [3, 9], computer workers and engineering professionals prefer to use a more abstract hand model [7].

These hand motions are described as the combination of rotations at various joints around different axes. There is only one motion, flexion-extension (bending-extension, or simply called flexion) at DIP and PIP of each finger and the thumb IP. In addition to flexion, the thumb MCP and CMC joints and the finger MCP joint have side-to-side movement, called abduction-adduction.

The wrist bones have the most complicated movements. Because the palm bones have tendency to converge to a point in wrist bones, for simplicity, only one point is used to represent the wrist joint. There are six movements at this joint: one for bending (flexion), one for side-side movement, one for rotation (supination-pronation), and three for displacement in 3D space.

Even though a highly articulated organic structure, the human hand cannot generate any arbitrary gestures and is constrained: there are limitations on the natural movements of hand parts at their rotation joints [6, 7]. Typical constraints are:

- There exists a dependency in a finger between the flexion of the DIP joint and that of the PIP joint. More specifically, when you bend your finger at the PIP joint for θ degrees, your finger tip will automatically bend for $\frac{2}{3}\theta$ at its DIP joint.

- The MCP joint of the middle finger displays little adduction.

- There exist various constraints among the fingers. For example, when you bend your ring finger at

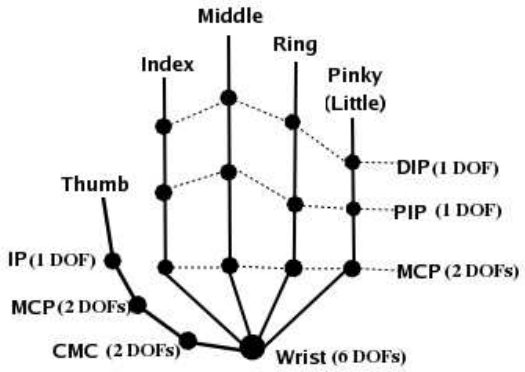


Figure 2: The simplified representation of the hand structure.

the MCP joint, your middle finger and pinky will bend at their MCP joints to different degrees.

2.2. Constructing the virtual hand

We implemented our first version of the hand model on an SGI workstation with Open Inventor and then transformed all the code to Linux with an Open Source Version of Open Inventor, Coin3d Inventor (`http://www.coin3d.org`). For the hand model to be more realistic, we include a portion of the forearm.

First, sixteen hand parts are identified: three for the thumb and three for each of the fingers and one for the palm. Then, the hand joints (finger's DIP, PIP, and MCP, thumb's IP, MCP, and CMC, and wrist joint) are calculated from the boundaries. Finally, a coordinate system is attached to each of the 16 joints. These coordinate systems serve as the object (local) coordinate systems for hand parts: one hand part belongs to one coordinate system and at the initial step is aligned with that coordinate system. Each coordinate system is attached to another coordinate system just as its corresponding hand part is attached to another hand part. For example, the index finger has three hand parts in their corresponding coordinates systems with origins at index finger's DIP, PIP, and MCP joints. The coordinate system at the DIP joint is attached that at the PIP joint, which is attached to that at the MCP joint.

The anatomical features of the hand are maintained by keeping the length of any part's bone (that is, the displacement (distance) of two adjacent coordinate systems is constant during hand motions). The center of the boundary of two adjacent hand parts is used as the origin of coordinate system. This approach is similar to the method for measuring hand motions used by medical and biomechanical professionals [3]. There is flexion at all the hand joints; abduction at each finger's MCP joint, each thumb's MCP and CMC joints, and at the wrist joint. Other movements are added to the wrist joint: a rotation from the forearm's rotation and 3D translation as a result of the shoulder's motion. The hand parts are primarily rigid with the exception of the components around the boundaries of two adjacent hand parts. When a hand part rotates at a joint, the mesh points that are close to the hand joint on both adjacent hand parts produce the most deformations. Deformation weights are distributed according to point closeness and position.

3. Generating natural gestures

The hand, when in motion under constraints, generates what we call natural hand gestures. One example of a natural gesture is that when you bend your middle finger at its MCP joint: your index and ring fingers will automatically follow the middle finger's movement and even your pinky will to a lesser degree. Although people display differences in the magnitude of hand constraints, all the natural movements observe the same pattern: the related fingers rotate the same way, only with different degrees.

The application of the hand constraints in our model is based on the constraint values obtained by medical researchers [3, 9]. First, the static hand constraints are embedded into the hand model, setting up motion (rotation) ranges for hand joint movements. Specifically, there are limitations on flexions at all the hand joints, on abductions at each finger's MCP joint, the thumb's MCP and CMC joints, at the wrist, and on the rotation of the hand at the wrist.

The intra-finger dynamic constraints (how a hand joint in one finger affects other joints in the same finger) are applied by following the principle of two-thirds: the flexion angle at the DIP joint is two thirds of that at the PIP joint, and *vice versa*. Test and correction in the implementation of the principle on the hand model was necessary.

Finally, the inter-finger dynamic constraints at fingers' MCP joints (how one finger moving at its MCP joint affects the other fingers) is implemented.

In the case of abductions, the middle finger can have slight abduction-adduction, and the ring finger follows the pinky in abduction when the pinky moves far away from the ring in abduction. For the case of flexions at fingers' MCP joints, the following issues are considered:

- The flexion angle range at a finger's MCP joint is divided into three parts: low, middle, high flexion subranges.
- When a finger (the active finger) bends at its MCP joint, it exerts influence on the MCP flexions of other fingers (the passive fingers). The

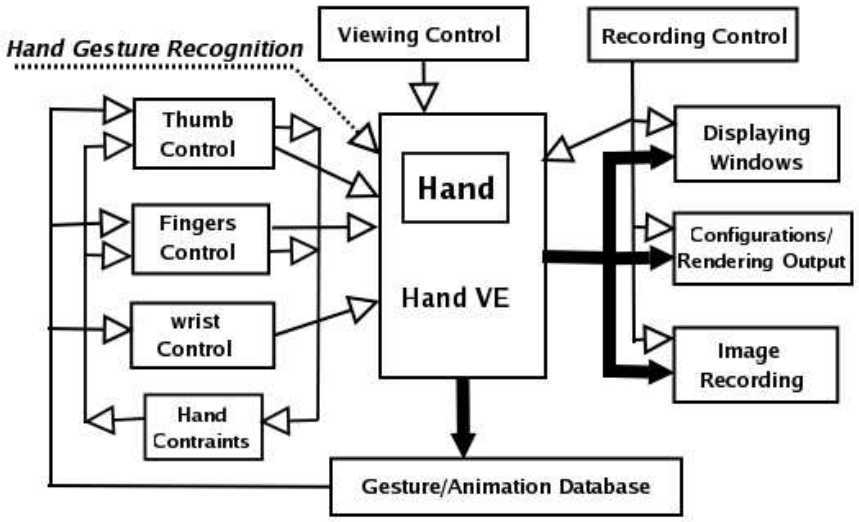


Figure 3: The architecture of vHand.

influence varies depending on the active finger's flexion angle value (on which of the low, middle, and high flexion subranges it is in) and on the closeness of the passive finger to the active finger. For example, when bending at the MCP joint, the index finger has the strongest influence on the middle finger, the moderate on the ring finger, and the least on the pinky (the influence becomes stronger as the index finger bends from the low subrange to the high subrange).

- The influence factors are not measured in a multiplying factor (by which the active flexion angle is multiplied to give the passive flexion angles) but are included in the maximum difference between the active flexion angle and the passive flexion angle. If the flexion angle difference between the active flexion angle and the passive flexion angle is larger than the maximum value, the passive flexion angle should be adjusted so that the difference is equal to the influence factor (the maximum value). For example, suppose that the index finger (active finger) bends to θ_I (active flexion angle) in one of the subranges and that the middle and ring fingers (two passive fingers) are at θ_M and θ_R. Also, suppose that the index in this subrange has the influence factor of of 20^o on the middle finger and the influence factor of 35^o on the ring finger. There should exist inequalities: $\theta_M \leq \theta_I \pm 20^o$ and $\theta_R \leq \theta_I \pm 35^o$. If either inequality is not satisfied, we adjust θ_M or θ_R such that $\theta_M = \theta_I \pm 20^o$ or $\theta_R = \theta_I \pm 35^o$. The positions of the active finger and the passive finger determine the choice of the sign "+" or "−".
- All the influence factors among all four fingers are estimated through experiments on our hand model and on real human hands.

We have recorded some animation sessions of the constraint applications for demonstration (see `http://www.beifang.info/RActivities/HCI/vHand/`).

4. Interfacing the virtual hand

A graphical user interface was designed and implemented to facilitate the process of generating hand gestures and animations. The major operations were identified: displaying the virtual hand rendered from different viewpoints (with various virtual cameras), controlling the virtual hand, creating hand gestures and animations, and recording hand image/rendering sequences. This interface was extended by creating a virtual environment for the hand model and designing the effective operations in such a way that this GUI should play the role of an interactive interface between the virtual hand and its users. The hand model, and the interface consists of a hand simulation, and its design architecture is shown in Figure 3.

4.1. Theoretical and practical considerations

The theoretical guidelines for designing this interactive simulation system, vHand, are: (1) Usability goals [10], which require a system to be effective, efficient, easy to learn, and easy to remember how to use; (2) User experience goals [10], which require that

a system should be satisfying, esthetically pleasing, and supportive of creativity; and (3) The conformity of the program model to the user model [11] which means that the program should be designed such that it behaves in the way the users think it should do.

The practical considerations of implementing vHand include: (1) Using a cross-platform GUI toolkit (Qt) and a graphical library kit; (2) Building fixed and movable virtual cameras; (3) Creating several display windows for virtual hands rendered from various cameras; (4) Using interface metaphors and affordances as described in [11]; and (5) Constructing a built-in database for storing hand gestures and animation sequences.

Figure 1 showed a screen shot of the layout of the current version of vHand. Menus are positioned on the top. The five display windows for displaying renderings from different view points are on the upper half of the screen. On the lower half of the screen are Qt tab dialog boxes for specifying operations on the virtual hand and the virtual environment. In Figure 1, the finger control panel has been selected. The lower-right of the screen shows logos and a small display area for demonstrating an animated hand image sequence.

4.2. Viewing the virtual hand

The virtual hand resides in a virtual rectangular box where there are eight virtual cameras at its corners and a movable camera with initial position at the center of the front side. All the cameras are facing the hand. The virtual environment is controlled though the "Viewing" control panel shown in Figure 4. The operations in this panel include setting the background color, coordinating display windows with virtual cameras, and adjusting camera parameters.

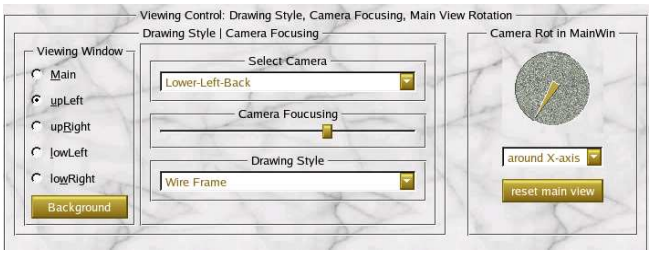


Figure 4: The viewing control panel.

There are five display windows and nine cameras, and the windows are displaying rendered hand images from the cameras. To set the camera and display, first use the radio-buttons on the left part of the panel to select display area. Then choose a camera from the "Select camera" combo box on the middle part of panel for the selected display window. A camera's focus can be adjusted by sliding the "Camera focusing" sliderbar. The rendering effect will be shown on the display window connected with that camera. The "Drawing style" combo box is used to render the virtual hand in the form of solid, lines, or points in the selected display area.

When the "Background" button is clicked, a color palette will pop up where a color can be selected or created for the background for the virtual environment. The camera can rotate around three axes (*via* input from the combo box) and rotate to any angle (*via* the input from the "Camera rot" dial).

Figure 5 shows some hand gestures rendered in different drawing styles, camera focuses, lighting models, and backgrounds.

4.3. Controlling the virtual hand

There are three control panels, named "Thumb control," "Finger control," and "Wrist control," to control the motions at hand joints. The lower part of Figure 1 shows the finger control panel. Operations on the thumb and wrist are similar to those on finger control panels.

To adjust a finger's motion, first choose the finger on the leftmost of the panel, then specify the motion of the finger joints: the finger tip (DIP) on the left, the middle joint (PIP) on the middle, and the finger base joint (MCP) on the right. Also, the dial allows rotation (flexion-extension) and the sliderbar controls side-side movement (abduction-adduction)

For controlling the thumb's motion, there are three dials and two sliderbars for the thumb's flexions at its three joints and abductions at its lower two joints. In the case of wrist movement, three dials and three sliderbars are used for bending, side-to-side rotation, twisting, and displacement in 3D space.

The hand constraints can be switched on, which is a menu item under the "Hand" menu, to increase the speed of creating a hand gesture through adjusting the hand joint motions. This increases the speed because the hand constraints entail related hand joint motions. Furthermore, turning on hand constraints guarantees the generation of natural hand gestures.

4.4. Creating hand gestures and animations

A data structure was designed and implemented within the simulation system for storing, retrieving, and editing hand gestures which can be constructed by adjusting hand joint motions as discussed in the previous section.

Some basic hand gestures have been constructed and stored in the database they can help the needs of serious vHand users in constructing particular hand gestures. Figure 6 shows the operational panel for this gesture database.

To create a particular hand gesture, the users of vHand can load from the gesture database a hand ges-

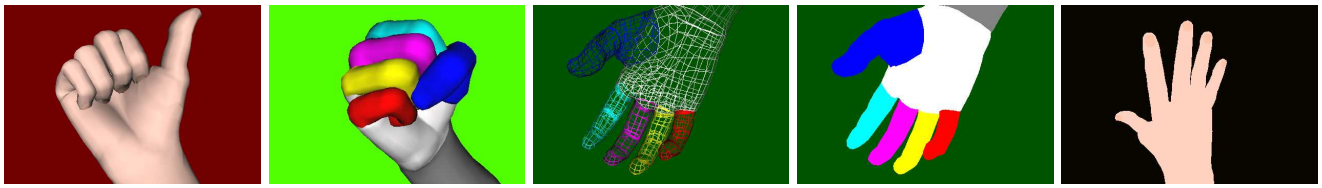


Figure 5: The hand rendered in different drawing styles and environments.

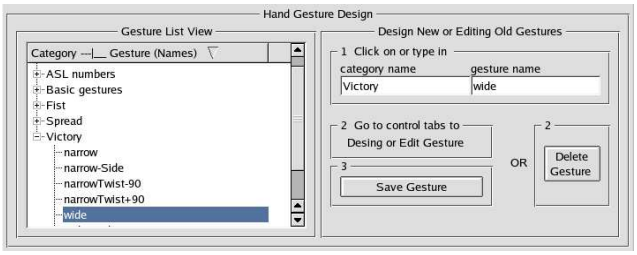


Figure 6: Control panel for gesture management.

ture (by clicking on one gesture name in the "Gesture list view" area on the panel) that is close to the particular gesture they desire. The hand joint motions can fine tuned through the thumb, finger, and wrist control panels. For viewing the immediate output effect, the users can adjust the unfixed camera and connect it to the larger display window and connect some other cameras to the other display windows.

After a gesture has been constructed, users can store it in a particular gesture group (category). They can name the gesture and the gesture group at their convenience. Of course, gestures can be deleted and edited. Editing a gesture involves first retrieving (loading) and then modifying the gesture, just as described in the above process.

Hand animation consists of a sequence of hand gestures, and a built-in data structure was implemented in vHand for creating and storing and editing hand animations. The hand animation control panel is shown in Figure 7.

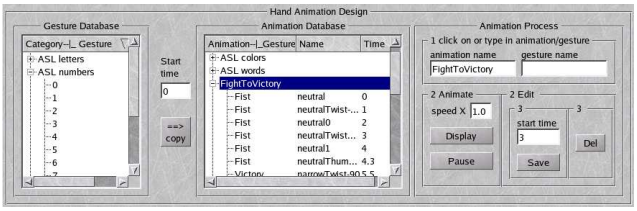


Figure 7: Control panel for animation management.

To create a new hand animation, the users first give the animation a name. Next, choose a hand gesture from the gesture database and type in its starting time. Click the "Copy" button and the gesture will go to the right place in the animation database. They can immediately see the animation by clicking on the "Display" button and can save the sequence by clicking on the "Save" button.

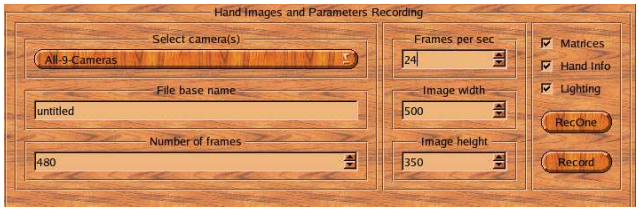


Figure 8: Control panel for recording hand images and rendering parameters.

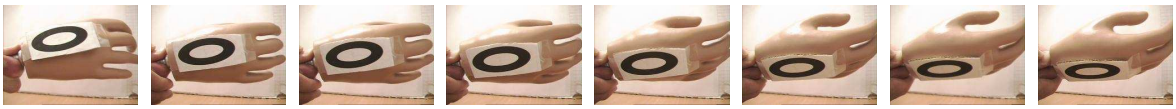
The default animation speed is set at 20 frames per second. During the rendering of an animation sequence, vHand automatically generates new hand configurations based on two adjacent hand gestures (and their starting time) in the animation sequence and the animation speed. New hand configurations are thus interpolated, and vHand renders these newly calculated hand gestures according the rendering speed. The rendering speed can be changed by typing a multiplying factor in the input area "Speed X".

An animation process can also be edited. To delete a gesture in the animation sequence, simply click on the gesture name within the sequence in the animation database and then click on the "Del" button. To insert a gesture in the sequence, choose the inserted gesture in the gesture database, select in the sequence the gesture after which the inserted gesture will reside, give the starting time, and click the "Copy" button. To change the starting time for a gesture in an animation sequence, first choose that gesture, modify the starting time that is displayed in the input area under the "Edit" frame, and click on the "Save" button.

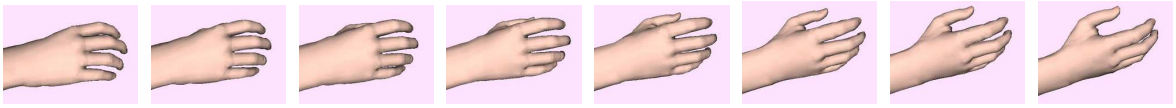
4.5. Recording rendering processes

vHand records hand gestures and animations in image files and saves their corresponding hand configurations and rendering parameters (such as OpenGL rendering matrices) in text files. This is done through the operations on the "Recording" panel, as shown in Figure 8.

The recording process begins by choosing the cameras (the rendered image sources). With eight fixed



(a) Inputs from hand gesture analysis system.



(b) The matched outputs in vHand for the inputs in (a).

Figure 9: vHand as an output platform for hand gesture analysis and recognition system.

cameras and one movable camera, images can be recorded from nine different viewing points for a single hand rendering configuration. An image size is chosen and file name is specified. If more than one camera is chosen, vHand will generate suffixes and append them to the given file name to indicate the image source. Recording animation sequences requires the animation recording parameters to be specified: the total number of frames and the number of frames per second. Within vHand, an animation sequence renders images according to the parameters in its database. Because writing images to a hard disk takes time, when vHand decides that it is time to record a frame (with the result calculated from the animation recording parameters), it *stops* the inner timing clock of the rendering mechanism in order to write the rendered images for the current hand configuration. And after finishing the writing process for the current hand configuration, it switches on the inner timing clock and continues the rendering and recording process.

The virtual hand can be rendered not only with Phong lighting model but also with a base object color model (only diffuse lighting). A special colored hand is possible with fingers in different colors. vHand can save on disk the rendered images with the rendering parameters such as OpenGL projection and modelview matrices. It can also keep on file the hand configurations such as hand orientation, position, and hand joint angles. These operations are available through the recording panel.

All the hand images in this paper were recorded with the use of the recording panel with the exception of the vHand's layout shot (Figure 1) and the control panels (Figure 4, Figure 6, Figure 7, Figure 8).

5. Imparting expressive content

This hand simulation system, vHand, can be used to generate meaningful hand gestures and animation sequences. These sequences provide hand gesture and animation data as input to the hand gesture analysis and recognition system, display corresponding hand configurations as the output platform for such a gesture analysis system, and convey rich expressions as a pioneer prototype for an ASL word generator.

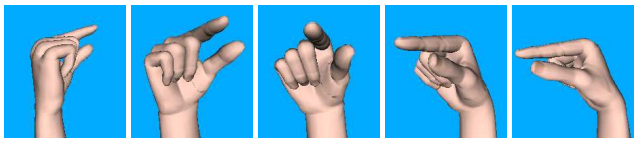
Advantages to using vHand to provide hand ground truth data for computer vision research on human hand analysis and recognition include: vHand can produce natural, demonstrative hand gestures; the system generates gestures and animations rapidly (more than real time) from any view points at the same time; the virtual hand will not move in the virtual environment; the hand configuration, virtual cameras, and background can be set up quantitatively, and these values, together with rendering information, can be outputted accurately; and to facilitate gesture analysis the hand can be rendered in different styles (with various environments): a lifelike hand, a colored hand, a wire frame, and a colored hand and a lifelike hand with only a diffused lighting model. Figure 5 presents these options in the same order. Some exemplar hand data can be found at `http://www.beifang.info/RActivities/HCI/vHand/`.

When connected to a hand gesture analysis system, vHand can play the role of an output platform, displaying the hand postures estimated from that system. vHand is coordinated with the gesture analysis system in two steps: (1) Align the coordinate system of vHand's movable camera with that of the analysis system's camera; and (2) Transform vHand's virtual hand such that it matches the position and orientation of the hand in the analysis system. Figure 9 demonstrates some results from the matching process (due to the initial stage of our hand gesture analysis system, a dummy hand is used here).

We believe that vHand can be utilized to help convey ASL meanings. In our current version, vHand is used to demonstrate ASL numbers and letters and some ASL words. We have stored hand gestures corresponding to the ASL numbers and letters and some ASL words in the gesture and animation database. Two ASL signs are given here as examples. Figure 10a shows a part of the session for the ASL sign "Goodbye" which is basically a wave of the fingers while Fig-

ure 10b displays a part of the session for the ASL word "green" which is a rotation of the wrist of the ASL letter "G".

(a). Part of the session of the ASL sign "Good-bye"



(b). Part of the session of the ASL word "Green"

Figure 10: Illustration of the ASL sign "Good-bye" and the ASL word "Green".

6. Current and Future work

Our current hand model cannot represent different hand sizes (thin, thick hands). The calibration of the virtual hand to different representative hand shapes will be part of the future work on this project. We have combined the vHand into a virtual human body and are constructing a virtual gesture production system. A gesture database is designed (using MySQL) for storing and managing the whole body virtual gestures. Also, modeling of basic facial expressions (such as happiness, sadness, surprise, fear, disgust, and anger) is underway.

We are now incorporating the above subsystems (virtual gesture generation, gesture database, and facial express modeling) into a Sign Language simulation system. User-friendly and effective GUIs are implemented for efficient operations on creating virtual gestures, matching the gestures to ASL linguistic parts (meaningful gesture animation sessions), and combining and editing these ASL parts into ASL sentences and paragraphs.

7. Acknowledgment

This project has been partly supported by NASA under grant NCC5-583. Jorge Usabiaga provided the posture data of a dummy hand out of his hand analysis system.

References

[1] Irene Albrecht, Jörg Haber, and Hans-Peter Seidel. Construction and animation of anatomically based human hand models. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 89–109, San Diego, California, July 2003.

[2] Paul W. Brand. *Clinical Mechanics of the Hand*. The C. V. Mosby Company, 1985.

[3] Edmund Y. S. Chao, Kai-Nan An, William P. Cooney III, and Ronald L Linscheid. *Biomechanics of the Hand: A Basic Research Study*. World Scientific Publishing Co. Pte. Ltd., Singapore, 1989.

[4] George ElKoura and Karan Singh. Handrix: animating the human hand. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 110–119, San Diego, California, July 2003.

[5] Tsuneya Kurihara and Matsuki Miyata. Modeling deformable human hands from medical images. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 355–363, Grenoble, France, August 2004.

[6] Jintae Lee and Tosiyasu L. Kunii. Model-based analysis of hand posture. *IEEE Computer Graphics and Applications*, pages 77–86, September 1995.

[7] John Lin, Ying Wu, and Thomas S. Huang. Modeling the constraints of human hand motion. In *in Proc. 5th Annual Federated Laboratory Symposium (ARL2001)*, pages 105–110, Maryland, April 2001.

[8] John McDonald, Jorge Toro, et al. An improved articulated model of the human hand. *The Visual Computer*, 17(3):158–166, May 2001.

[9] American Academy of Orthopaedic Surgeons. *Joint Motion: Method of Measuring and Recording*. Churchill Livingstone, New York, 1988.

[10] Jennifer Preece, Yvonne Rogers, and Helen Sharp. *Interaction Design: Beyond Human-Computer Interaction*. John Wiley & Sons, Inc, 2002.

[11] Joel Spolsky. *User Interface Design for Programmers*. Apress, 2001.

[12] David Joel Sturman. *Whole-hand Input*. PhD thesis, Massachusetts Institute of Technology, Febrary 1992. Available from World Wide Web: `xenia.media.mit.edu/~djs/thesis.ftp.html` [cited November 3, 2005].