University of Nevada, Reno

Virtual Hand: A HCI Testbed for Computer Vision Research on the Human Hand

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science with a major in Computer Engineering.

by

Beifang Yi

Dr. Frederick C. Harris, Jr., Thesis advisor

August 2003

We recommend that the thesis

prepared under our supervision by

Beifang Yi

entitled

Virtual Hand: A HCI Testbed

for Computer Vision Research on the Human Hand

be accepted in partial fulfillment of the

requirements for the degree of

MASTER OF SCIENCE

Frederick C. Harris, Jr., Ph.D., Advisor

Mircea Nicolescu, Ph.D., Committee Member

George Danko, Ph.D., At-Large Member

Marsha H. Read, Ph.D., Associate Dean, Graduate School

August 2003

Abstract

A virtual hand is constructed as a human computer interaction testbed for the use in computer vision experiments on the human hand gesture analysis and recognition. The hand model is based on the results from the studies on human hand in the areas of anatomy, biomechanics. A graphical user interface is built for the testbed in which the movements of hand joints can be controlled to produce any hand gestures. Hand constraints are considered in the model and animation is possible. The rendered hand configurations and animation can be recorded in both image sequences and text files for the model and its rendering parameters. The dynamic anatomy-based hand model produces in real time more realistic 3D hand than do the hand models used in current computer vision projects. The virtual hand combined interface is more convenient and provides more functions than do those peer interfaces for researches on hand.

Acknowledgments

I would like to thank Professor Harris, my advisor, for his generous help when it is most needed, for his guidance, support, and encouragement throughout the thesis process.

I am very grateful to Professor Nicolescu and Professor Danko for serving on my committee and for their valuable time. Dr. Danko's robotics course and Dr. Bebis' computer vision course have been a big help in my academic studies. In addition, my participation of Dr. Bebis' NASA group meetings has given me wonderful experiences and beneficial incentives in my work.

Finally, I am indebted to the financial support from the NASA project at UNR, which allowed me to have been concentrating on the development of this testbed and the writing of this thesis.

Last but not least, special thanks should be poured out to my family for their understanding of and patience with my career change from an engineer to a student.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In human computer interaction (HCI), traditional controlling and navigating devices, such as keyboard, mouse, and joystick, become cumbersome and unsuitable in the communication between man and machine. One of the most promising natural HCI media, as a substitute for the improper mechanical input tools is the use of human hand [61, 73, 81]. In order to take advantage of this new input modality, emphasis has been on the vision-based studies on hand modeling, hand gesture analysis and recognition. Although the philosophies and methodologies on the study of hand vary widely, the construction of a hand model is required either for the use as an convenient input channel or representation of the output results, or both. An appropriate graphical user interface is desirable from the view of HCI.

In this paper, a testbed, a software package *Virtual Hand*, is introduced which can help experiment on human hand in the area of computer vision domains. It includes a 3D dynamic human hand model and a graphical user interface based on the model. Results from anatomical and biomechanical studies on human hand are considered in the construction of virtual hand and the movements at hand joints. The hand model can produce any (natural) hand gestures by controlling rotational angles at hand joints.

The controlling process is very convenient through the graphical design interface. The model can also provide some animation in real time. Hand configurations in the generated gestures and animation can be recorded as rendered image, the configuration parameters such as hand joint angles, whole hand box size in space, OpenGL rendering matrices and camera parameters.

The testbed has potential applications in hand pose estimation, hand gesture analysis and recognition, and hand (inverse) kinematics studies. The virtual hand looks more realistic than do those hand models used in the current computer vision studies on the hand. The virtual hand combined interface is more convenient and provides more functions than do most of those peer interfaces for research on the hand.

The organization of this thesis is as follows: Chapter 2 provides the background in the HCI-related vision-based researches on hand. Chapter 3 is an overview of the hand-related topics about hand gesture and function from anatomical, biomechanical experiments on hand, and about hand modeling from the point of computer graphics as well as computer vision. Chapter 4 concerns with virtual hand modeling. Approaches are provided of building static and dynamic hand models with Open Inventor. Results will be shown in the chapter. Chapter 5 presents the design and implementation of an interactive GUI interface based on the hand model. The interface performances will be exhibited through various functions of the testbed such as different gestures, hand animation, image recording, and etc. Chapter 6 discusses the application of the testbed in computer vision experiments on the human hand. The testbed's APIs are also included in this chapter. Chapter 7 presents conclusions and future work.

Chapter 2

HCI-related Vision-based Researches on Human Hand

2.1 Human-Computer Interaction

Human expectation for an ease-of-use environment between man and machine has always been an interesting and challenging disciplinary engagement since human could manipulate man-made tools. With the appearance of the computer and the achievements in computer science and technology and its related engineering fields, the intensity of this hope has become so strong that an acronym *HCI* is a common word today.

HCI stands for Human-Computer Interface (or Interaction). A working definition was given by [39]:

> Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them.

Another two definitions were introduced in [69] as a complement for the above one, to emphasize on four issues of HCI: *people, computing system, usability, and interaction.*

Thus HCI "draws from supporting knowledge on both the machine and the human sides" [39]. On the human side, it focuses on the human information-processing characteristics, that is, how the models of human cognition are structured. On the machine side, HCI concentrates on the input/output devices, computer graphics, and dialog architecture and techniques. Humans have five senses: sight, hearing, touch, smell, taste. “Almost any natural communication among humans involves multiple, concurrent modes of communication,” and therefore “people prefer to interact multimodally with computers” [70]. Traditional computers have made full use of “touch faculty”: keyboard and pointing devices such as mouse, touchpad, and trackball. At present, HCI is trying to take as its modalities those other natural means that human employ, especially the sight(vision) sense. Vision-based HCI is a very vigorous research branch of HCI.

2.2 Vision-Based Human-Computer Interaction

By vision we refer to the use of cameras and a set of visual or graphical techniques for presenting and processing information. Compared with the non-vision-based HCI systems, vision-based HCI has the advantage of unobtrusiveness and gives a sense of “naturalness” and being comfortable during the process of human-machine interaction. The input to computer is now a set of images (obtained with a camera) of human activities (gestures, movements, emotional expressions) or object movements. The representative features corresponding with the human or object are extracted from the images, analyzed, and synthesized for higher processes such as tracking and recognition.

The use of this visual sensing is incorporated as a vigorous and promising input mode into other modalities of HCI. For example, video signals from the cameras are synchronized with the input from the physiological sensors, allowing an unambiguous record of physiological response to driving events [36]; a computer pantomime theater [64], an interactive virtual environment interface [80]; a speech and vision integration system [41]; a platform for simultaneously tracking of multiple people and recognition of their behaviors for high-level interpretation [71]; a 3-D operating system [1]; an eye interpretation engine [32, 33]; an affective computing system [62, 63, 66]; the affection/emotion detection mouse [9]; a driver's fatigue monitoring system [13, 14, 44] face detection [20, 31]; on-road car detection [74, 75].

2.3 HCI-related Vision-based Researches on Human Hand

Human hand gestures, from the simplest actions of pointing at objects to the most complex ones that can express our feelings, are a kind of non-verbal interactions among humans. The most commonly used hand gestures in HCI context are symbols: the gestures that have a linguistic role and belong to the communicative gesture category [61]. Hand gesture modeling (temporal and spatial), analysis, and recognition are issues in vision-based hand HCI studies.

Human hand gestures are a process of natural dynamic actions over a certain interval and the motion of hands conveys meaningful indications. The *temporal gesture modeling* is required for this dynamic hand gesture analysis, in which temporal segmentation of gestures from other unintentional hand movements is a big issue. Because of the complexity of gestural interpretation in the allowed temporal variability of hand gestures, the work in hand gestural studies has been largely confined to the hand posture estimation [61].

Two major approaches have been used in hand spatial modeling: 3D hand model-based methods and appearance-based methods. The former considers hand joint angles and palm position as hand parameters, in addition to hand skeleton structure.

The latter focuses on appearance of hand in the image and models hand gestures by relating the appearance of any gesture to the appearance of a set of predefined template gestures.

Hand gesture analysis and recognition are challenging topics in vision-based HCI. There are three steps in the gesture analysis: hand localization and segmentation, hand feature extraction, and hand model parameter computation. The results from the former step are fed into latter step. And the final results (computed model parameters – trajectory in parameter space) are utilized for gesture tracking and recognition [61].

The approaches in hand gesture analysis and recognition have provided a variety of applications in many areas such as: American Sign Language [46, 72], hand pose recognition [10]; hand gestures for replacing mousing in window operation [48]; tracking [40, 68]; graphics animation [79]; selecting targets [45]; controlling TV and lamp by using different types of hand postures [27].

Chapter 3

Overview of Hand-related Topics

3.1 The Dexterity of the Human Hand

Humans have five senses: sight, hearing, smell, taste, and touch. With exception of the sense of touch, animals enjoy far more acute, perfect sensory faculties. "We ought to define the hand as belonging exclusively to man," Sir Charles Bell acknowledged in 1885 [23, p. 26] after having thoroughly studied and painstakingly compared the animal extremities that have thumb and fingers with human hand, "corresponding in sensibility and motion with that ingenuity which converts the being who is the weakest in natural defence, to the ruler over animate and inanimate nature." The hand's capability of executing "whatever man's ingenuity suggests" corresponds with human's superior mental intelligence [23, p. 157]. A "lovely hand is the product of a lovely mind. The involvement of the hand can be seen in the face which is in itself a sort of mirror to the mind" [58, p. 21].

Naturally, humans have made a full use of the hand—"the ready instrument of the mind [23]"—in not only administering the harsh natural environment but also enriching human spiritual life. For example, ancient men did have meditative thought on the hand (Figure 3.1a). For the passing decades, computer system has been using one of its peripheral devices–mouse to get man's message through hand (Figure 3.1b).

And in the future, the very human hand gestures will directly feed information in the human-computer interaction (Figure 3.1c).

(a) (b) (c)

Figure 3.1: The human hand as intermediate tool: (a) A red hand silhouette painted on a stone slab, about 10,000-8,000 years ago (taken from [2]); (b) Mouse/hand as a medium between computer and human (taken from [6]); (c) Hand gestures provide input in HCI (taken from [4]).

Really, this "ready instrument of the mind" deserves a mindful look at its functionality and gesticulation.

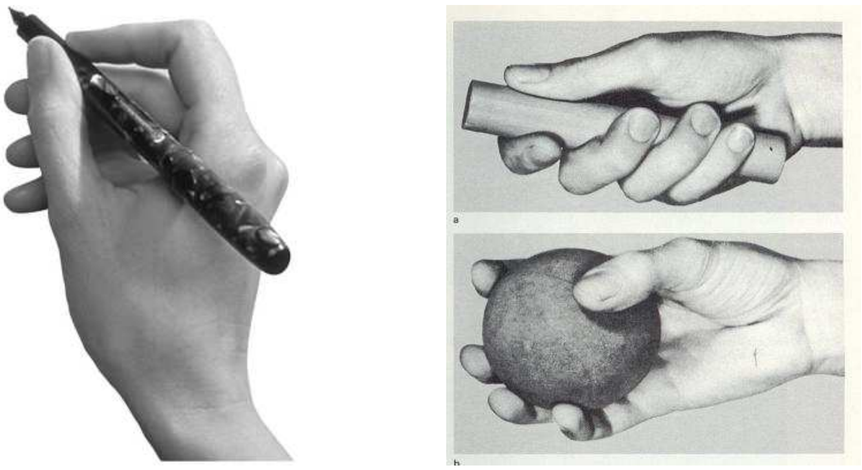
3.1.1 Function of the Hand

"The hand at rest is beautiful in its tranquility, but it is infinitely more appealing in the the flow of action" [58, p. 21]. The graceful movement of the thumb and fingers demonstrates the practical functionality of hand, which can be roughly seen through opposition, prehensile and non-prehensile movements, all of which are not excluded with each other.

Opposition is the most important movement of the human hand. The thumb plays a significant role in opposition. Napier gave a definition for it: "Opposition is a movement by which the pulp surface of the thumb is placed squarely in contact with – or diametrically opposite to – the terminal pads of one or all of the remaining digits" [58, p. 68]. Example of opposition can be seen in Figure 3.2a.

Generally, there are two classes of hand movement of which human is capable: prehensile and non-prehensile. Some of the non-prehensile movements include pushing, lifting, tapping and pushing movements of the whole hand.

Prehensile movements are typified by gripping or pinching between the digits and palm an object, whether fixed or free. Napier conjectured that there are two main prehensile patterns (power grip and precision grip, Figure 3.2b) and two subsidiary prehensile patterns (hook grip and scissor grip). It should be noted that the type of grip used in any activity is a function of the activity itself and does not depend on the shape or size of the object gripped [58, p. 77].



(a) (b)

Figure 3.2: Functions of the hand: (a) Opposition in practice (taken from [7]); (b) Power grip (upper) and precision grip (lower) (taken from [58, p. 75]).

3.1.2 Gestures of the Hand

David McNeil did a careful studies on people's spontaneous gestures that accompany their speaking and pointed out "that *gestures are an integral part of language as much as are words, phrases, and sentences–gesture and language are one system*" [56, p. 2].

Napier studied the evolution, mechanics, and functions of the human hand and came to a conclusion that "if language was given to men to conceal their thoughts, then gesture's purpose was to disclose them" [58]. McNeil went on to explore *how* human thoughts are disclosed in gestures and classified human's gestures into four types of gesture, *iconics, metaphorics, beats, deictics*, in addition to a special kind of gesture called *cohesives* which is a combination of any of the four gestures [56].

A gesture can be analyzed in its time and space patterns. A gestures cycle begins with gesture preparation in which the hand moves to a rest position, culminates in stroke period during which the full meaning of gesture is expressed, and ends with retraction in which the hand returns to a rest position. Different types of gestures tend to congregate in different regions in the gesture space which can be visualized as a shallow disk in front of the speaker [56].

3.1.3 Whole-hand Input Issues for HCI

By whole-hand input, Sturman defined it as "the full and direct use of the hand's capabilities for the control of computer-mediated tasks" [73]. It means the direct measurement of hand motion (finger flexions/abductions/rotations and/or whole hand rotation/displacement) rather than measurement of the motion of a device manipulated by the hand. Sturman also distinguished three important issues on whole-hand input: *appropriate use, control design, and device.*

The effectiveness of whole-hand action can be evaluated with the experiments on low-level physiological capability, task performance and the cognitive basis of hand functions. Fitts' law [54, 73] and "steering law" [8, 15] can be used for this purpose.

3.2 Biomechanics of the Human Hand

This section is an overview of results from medical studies on hand: anatomy, biomechanics, and constraints of human hand.

3.2.1 Anatomy of the Human Hand

The human hand can been seen as an organic structure composed of muscles, tendons, and bones. Tendon is the tough cord or band of dense white fibrous connective tissue that unites the muscle with some other part (a bone) and transmits the force which the muscle exerts, thus movement at a joint occurs. In the case of the hand, most of the muscle mass lies in the forearm with long tendons which transmits force to the fingers and allows the hand to be light and flexible without sacrificing strength. Hand movement is thus described with the interconnections and interactions of muscles and tendons. But our interest is in the hand skeleton structures, so we will dispense with this concept but use the structure of hand bones. Therefore, we can say that hand movement is determined by the position/orientation of the hand bones. A hand skeleton is shown in Figure 3.3.

3.2.2 Hand Joint Motion

The hand as a machine

Although the hand is more than a machine and studying the hand with only mechanical principles is to "debase a thing of exquisite beauty and and sensitivity to compare it with the crude and smelly things that we call machines" [26], according to Paul Brand, there are reasons why we have to think of the hand as a machine sometimes, to understand that the mechanical principles can be applied on hand, and to be able to produce modified mechanics of hand joints and tendon placements [26].

Figure 3.3: Hand (left) bones (taken from [5]).

He considered three major aspects of the hand machine: the motor (muscles), the transmission (tendons, bones, joints), and the application (through skin and pulp tissues). The relationship among them is shown in Figure 3.4.

Hand Joints

Movement at a joint is produced by the contraction of muscles. And hand motion involving different joints are fulfilled through the muscle contraction, tendon transmission and skin tissue application. Figure 3.5 demonstrates some common motions

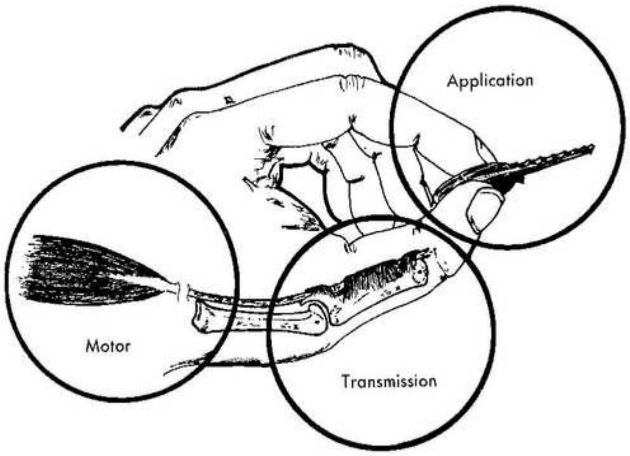


Figure 3.4: The three major elements of hand motion (taken from [26, p. 9]: muscles served as motor, tendons/bones/joints used as the transmission, and skin and pulp tissues for application. They are controlled and moderated by motor nerves and by sensation.

of hand joints.

To describe and measure specific hand motion, it would be better at first to give a brief explanation of the hand joints. Figure 3.6 is structural diagram for fingers and thumb from which we can easily identify the joints:

- Finger joints

  DIP Distal interphalangeal joint

  PIP Proximal interphalangeal joint

  MCP Metacarpophalangeal joint

- Thumb joints

  IP Thumb interphalangeal joint

  MCP Thumb metacarpophalangeal joint

  CMC Thumb carpometacarpal joint

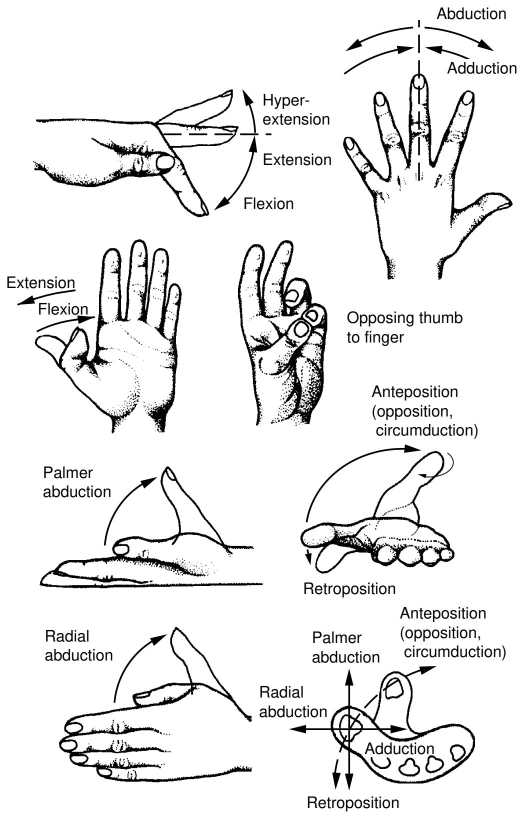


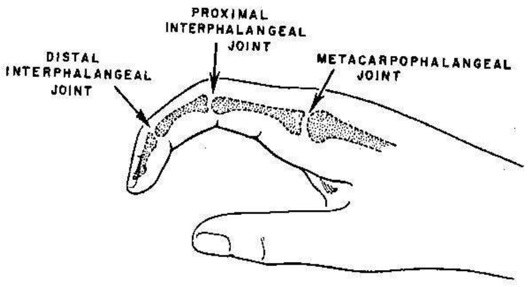
Figure 3.5: Common hand joint motions (taken from [73] and [60]).

Hand joint motions

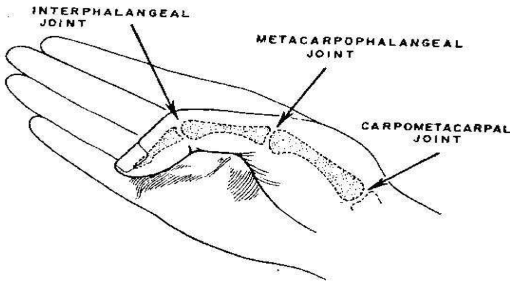
All the motions of a joint are measured from *zero starting position*, which varies with different hand joints. There are three general types of joint motions [60]:

1. *One plane* (or *degree*) *freedom* of motion, or the hinge joint, has natural motion in one direction only from the Zero Starting Position. For example, flexion and extension (away from and return to Zero position) at middle finger PIP.

2. *Two plane* (or *degree*) *freedom* of motion are about the joints with natural motion in two planes originating from the Zero Starting Position. Wrist movement (flexion and abduction) is an example.



(a)



(b)

Figure 3.6: Finger and thumb joint definition (taken from [60]): (a) Finger joints; and (b) Thumb joints.

3. *Ball and socket* joint motion consists of three dimensional, compound or rotatory motion. A should is a typical example of this type of motion. Strictly speaking, there is no such motion in hand, but we can say thumb motion at CMC is a three dimensional motion.

There are detailed descriptions and specifications of the zero starting positions and movement ranges for all the (hand) joints in [60].

3.2.3 Boimechanics of the hand

From a biomechanical viewpoint, the human hand can be considered as a linkage system of connected bony segments, in which the joints between each phalanx are spanned by ligaments, tendons, and muscles. To study hand motion and its force analysis under various functional activities, a three-dimensional normative model of the human hand was established in [28], based on the anatomical study of normal hand specimens.

Experiments were carried out for four of the finger motions: flexion-extension at distal interphalangeal (DIP) joint, proximal interphalangeal (PIP) joint, and metacarpophanlangeal (MCP) joint, as well as abduction-adduction of the MCP. Six Cartesian coordinate systems were established for a finger to define the locations and orientations of tendons (of the finger) and to describe the joint configurations. At a joint, a pair of coordinate systems are assigned (See Figure 3.7):. the proximal system (X_P, Y_P, Z_P) which is located at the approximate center of rotation of the phalangeal and metacarpal head, and the distal system (X_D, Y_D, Z_D), which is a translation of the proximal system to the center of the concave articular surface.

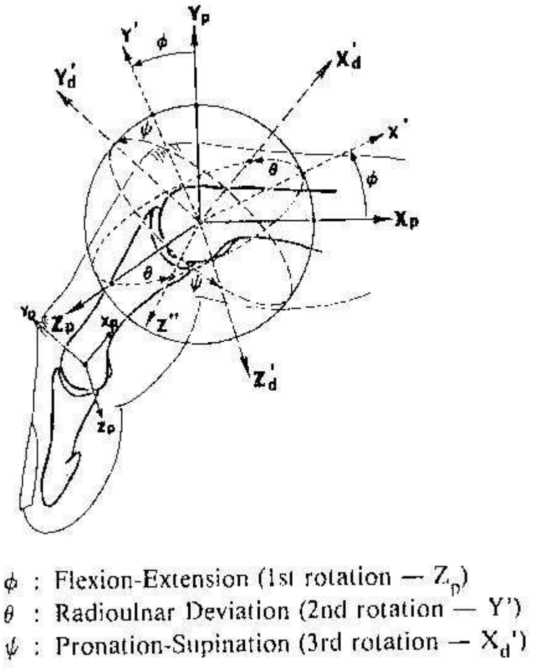


Figure 3.7: Eulerian angles used to define the orientations of finger digits (taken from [28]).

When the finger is in a functional configuration other than the neutral position, there exists a transformation (rotation and translation), at a particular joint, between the proximal coordinate system and the distal system:

$$\begin{bmatrix} X_D \\ Y_D \\ Z_D \end{bmatrix} = \begin{bmatrix} c\theta c\phi & c\theta s\phi & -s\theta \\ -c\psi s\phi + s\psi s\phi c\theta & c\psi c\phi + s\psi s\theta s\phi & s\psi c\theta \\ s\phi s\psi + c\psi c\phi s\theta & -s\psi c\phi + c\psi s\theta s\phi & c\psi c\theta \end{bmatrix} \begin{bmatrix} X_P \\ Y_P \\ Z_P \end{bmatrix} + \begin{bmatrix} X_O \\ Y_O \\ Z_O \end{bmatrix}$$

in which

- $X_D, Y_D, Z_D =$ coordinates of a tendon point or components of a vector measured with respect to the *distal* system,

- $X_P, Y_P, Z_P =$ coordinates of a tendon point or components of a vector measured with respect to the *proximal* system,

- $X_O, Y_O, Z_O =$ coordinates of the origin of the *proximal system expressed in the distal system*,

- $\theta, \phi, \psi =$ Eulerian angles for flexion-extension, radio-ulnar deviation (abduction-adduction), and axial rotation respectively, see Figure 3.7,

- $s, c =$ sine and cosine functions.

Tendon excursion and moment arm at a joint can provide useful information for the consideration of deformation at a joint in hand modeling. The excursion and moment arm are calculated, depending on the pattern of tendon movement. One of the patterns concerns that tendon runs in a tendon sheath which holds it firmly in a constant position against the shaft of the bone but allows the tendon to curve smoothly, conforming to the joint architecture (Figure 3.8).

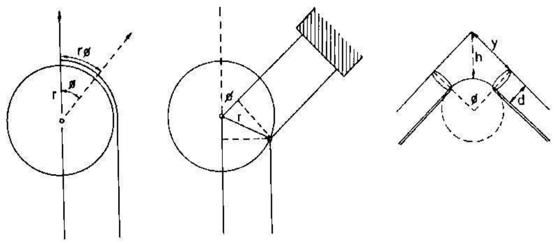


Figure 3.8: Tendon bridging a joint.

Hand joint orientation and range of motion

Forty normal subjects,with no clinical or radiologic evidence of joint disease, were studied in the normal finger joint motion (twenty-seven for the thumb case) [28]. The results of the study of normal finger joint motion are summarized in Table 3.1. Those on the thumb in Table 3.2. All the results are in degrees in mean value ($n = 40$).

Table 3.1: Finger joint motion ranges (taken from [28])

Finger Joint Motion Ranges(degrees)					
Joint	*Motion*	*Index*	*Middle*	*Ring*	*Pinky*
DIP	Flexion	73	80	75	78
	Extension	11.45			
PIP	Flexion	101	103	105	103
	Extension	10-12			6.5
MCP	Flexion	83	90	88	90
	Extension	-22	-22	-23	-34
Total flexion		256	274	268	272

Table 3.2: Thumb joint motion ranges (taken from [28])

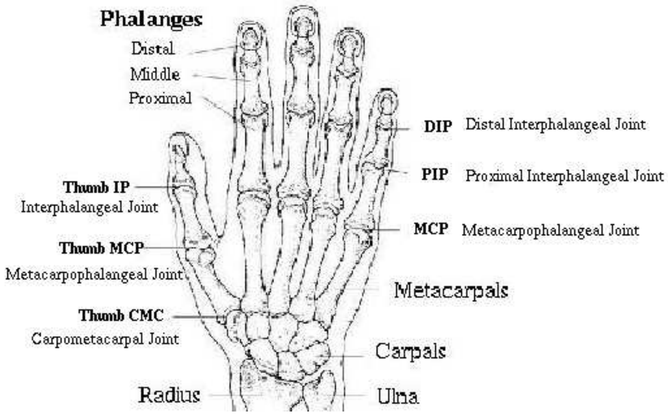
Thumb Joint Motion Ranges(degrees)		
Joint	*Motion*	*Thumb*
IP	Flexion-extension	100 ± 9
	Abduction-adduction	7.5 ± 10
	Rotation	8.4 ± 9
MCP	Flexion-extension	45 ± 16
	Abduction-adduction	8.7 ± 3.2
	Rotation	12.1 ± 4
CMC	Flexion-extension	53
	Abduction-adduction	42
	Rotation	17

The researchers noticed that metacarpophalangeal motion had considerable variability. In flexion, there was a range of $65^o - 107^o$ from neutral position when measured with respect to the metacarpals. The interphalangeal joint are more consistent with a smaller range of variation of $92^o - 125^o$. The variation range for the DIP extension angle is $10^o - 23^o$. Abduction-adduction and rotational motions were generally very small for the DIP and PIP joints but significant for MCP joints (not given quantitatively). Variations for the thumb are given in the Table 3.2.

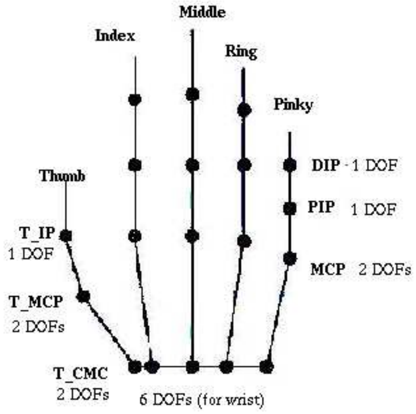
3.3 Hand Constraints

The human hand, consisting of many connected parts as the above sections indicated, is highly articulated. On the other hand, it is also highly constrained such that hand can not make any arbitrary gestures. Hand movements can be divided into two categories: natural movements in the hand without external interaction and unnatural movements under external force. The first one is also called active movement and the other passive movement. Hand constraints deal with the first case, for example, while you bend middle finger your ring will automatically bends.

We employ Figure 3.9 to introduce the hand constraints. From the diagrams, each finger has three joints: DIP, PIP, MCP and the thumb has also three: IP, MCP, CMC. We use T, I, M, R, P to represent thumb, index, middle, ring, and pinky (little). Also, "A-A" stands for "abduction-adduction", "F-E" for "flexion-extension", "F" for "flexion", θ for degree in flexion, abduction, etc. We can calculate the degrees of freedom (DOFs) in the hand following robotics conception. There are 27 DOFs: 1 DOF for DIP/PIP/IP (flexion-extension), 2 DOFs for MCP/CMC (flexion-extension and abduction-adduction), 6 DOFs for the wrist (rotation or supination-pronation, bend or flexion-extension, side-side or radial-ulnar deviation, and 3 DOFs for displacement). That is, 4 DOFs for each finger, 5 DOFs for the thumb, 6 for wrist.



(a)



(b)

Figure 3.9: Hand joint representation: (a) The hand joints—They can be seen as a structure of 17 active joints with 23 degrees of freedom (adapted from [58, p. 29]; (b) The stick representation of the hand structure (joints) (adapted from [81]).

Furthermore, since finger flexion might cause flexion of the adjacent fingers and a finger's extension is hindered by by the flexion of others, two phrases, *static joint angle limit* and *dynamic joint angle limit*, were introduced in [51]. These distinguish respectively between the joint angle limit calculated for all possible hand configurations and that calculated with the other joints in some specific configurations.

Three types of hand constraints are summarized in [53]: (1) the constraints limited by the finger motion as a result of hand anatomy, or static constraints, i.e., joint motion ranges, (2) the constraints imposed on joints during motion, or dynamic constraints, for example, the intra-finger and inter-finger constraint, and (3) that applied in performing natural motion, for example, curling all the fingers at the same time to make a fist.

Following is an enumeration of the hand constraints from studies on finger and thumb motion:

1. The four fingers are planar manipulators with the exception of the MCP joints [51].

2. The DIP and PIP joint flexions have a dependency [65] represented by:

$$\theta^F_{DIP} = \tfrac{2}{3}\theta^F_{PIP}$$

3. The MCP joint in middle finger displays little abduction and adduction [51, 65]:

$$\theta^{A-A}_{MCP(M)} = 0$$

4. The joint angle limits of the MCP joints depend on those of the neighboring fingers according to the following [51]:

   - $dmax(\theta^F_{MCP(I)}) = min(\theta^F_{MCP(M)} + 25, smax(\theta^F_{MCP(I)}))$
   - $dmin(\theta^F_{MCP(I)}) = max(\theta^F_{MCP(M)} - 54, smin(\theta^F_{MCP(I)}))$
   - $dmax(\theta^F_{MCP(M)}) = min(\theta^F_{MCP(I)} + 54, \theta^F_{MCP(R)} + 20, smax(\theta^F_{MCP(M)}))$
   - $dmin(\theta^F_{MCP(M)}) = min(\theta^F_{MCP(I)} - 25, \theta^F_{MCP(R)} - 45, smin(\theta^F_{MCP(M)}))$
   - $dmax(\theta^F_{MCP(R)}) = min(\theta^F_{MCP(M)} + 45, \theta^F_{MCP(L)} + 48, smax(\theta^F_{MCP(R)}))$
   - $dmin(\theta^F_{MCP(R)}) = min(\theta^F_{MCP(M)} - 20, \theta^F_{MCP(L)} - 44, smin(\theta^F_{MCP(R)}))$
   - $dmax(\theta^F_{MCP(L)}) = min(\theta^F_{MCP(R)} + 44, smax(\theta^F_{MCP(L)}))$
   - $dmin(\theta^F_{MCP(L)}) = max(\theta^F_{MCP(R)} - 48, smin(\theta^F_{MCP(L)}))$

   where *dmax, dmin, smax*, and *smin* are the variables for the dynamic and static joint angle limits with *min* and *max* being the minimum and maximum values.

5. There exists [51]:

$$dmax(\theta^{A-A}_{MCP}) = k \times smax(\theta^F_{MCP})$$

   where $k = \left(1 - \frac{1}{smax(\theta^F_{MCP})}\right)\theta^F_{MCP}$

6. The thumb has following equations [65]:

$$\theta^F_{CMC} = 2(\theta^F_{MCP(T)} - \tfrac{1}{6}\pi)$$

$$\theta^{A-A}_{CMC} = \frac{7}{5}\theta^{A-A}_{MCP(T)}$$

7. The MCP and PIP joints of a finger have a dependency [50] represented by:

$$\theta^{F}_{MCP} = k\theta^{F}_{PIP} \; 0 \leq k \leq \frac{1}{2}$$

where $k = \frac{1}{2}$ is suitable for dynamic cases.

8. There exist "finger planes" [29] for each finger: the five points—wrist joint, MCP joint, PIP joint, PIP joint and the finger tip—are coplanar (there may be a different wrist point for each finger).

9. There exists a "weak constraint" on thumb [29]:

$$\theta^{F}_{IP(T)} = a\theta^{A-A}_{MCP(T)} \; a \geq 0$$

10. There exists a "thumb plane" [29]: four points—CMC joint, MCP joint, IP joint, and thumb tip—are coplanar.

11. There exists a "palm plane" [29]: all the four MCP joints of the four fingers are located in a plane. It is assumed that this plane is perpendicular to the "finger plane" of the middle finger.

12. The axis of the thumb and the axes of the two phalanges (distal and middle) of each of the fingers converge to a point when fingers are clenched into a fist [51] (Figure 3.10). The process of the convergence means that the abduction and adduction angles continuously decrease as the flexion angles of the MCP joints increase.

One big advantage of applying hand constraint is its computational efficiency: reducing the size or dimensions of the search space and thus making the estimation

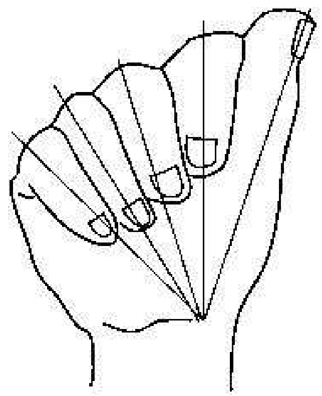


Figure 3.10: Convergence of fingers when the fist is clenched (taken from [51]).

of hand postures more effective; and synthesizing natural hand motion and thus producing realistic hand animation in hand animation and hand sign language analysis. Instead of considering the total 27 DOFs, Chua et al. [29] simplified the hand model by reducing it to 12 DOFs, Lin et al. [53] reduced it to 15 DOFs.

3.4 Hand Modeling

The human hand, consisting of bones, ligaments, muscles, tendons, soft tissues and skin, is an organic component of human body. It can be seen as a highly articulate, exceedingly complex mechanical structure: bones are linked at joints while muscles and tendons and ligaments provide the architectural basis for the flexibility of the hand. Hand modeling is a challenging topic belonging to the area of modeling, deformation, and animation of the human body. And human modeling and animation bring together state-of-the-art technologies [16, 17, 18, 19, 24, 57, 76].

Roughly speaking, hand shape models can be categorized into three groups [81]: geometrical models, physical models, and statistical models. The main issue in the latter two models is the consideration of deformation of hand: physical hand shape model focuses on an explicit representation of hand deformation under the internal and external forces applied to the hand, while statistical hand model implicitly learns the deformation of hand through a set of training examples (i.e., 2-D images of hand gestures). As for the geometrical hand shape models, a wide variety of choices are obtained, depending on the complexity, from simpler parameterized geometric shapes (such as cylinder of super-quadric), 3-D polygon meshes, to the complicated spline-based geometrical surfaces.

3.4.1 Hand Modeling and Animation with Deformation

General approaches to modeling

In virtual environments or HCI systems there is a need for behavioral animation on virtual humans with individualities—the severe behavioral requirement on the underlying animation system that must render its physical bodies. In body animation, there are two basic approaches: (1) to record the motion by using motion capture systems and then to modify or re-target such a motion to create individuality, and (2) to create computational models that are controlled by *a few parameters*. Researches [34, 76] have shown that the first method is tedious and difficult to implement and that the computational models, which give the body posture at specific time and thus are suited for physically-correct dynamic simulation, can offer promising solution on modeling and animation of a virtual human. Badler et al. [17, 18] provided various aspects and issues on consistent parameterizations for gesture and facial actions and described a parameterized action representation that allows a virtual agent to act, plan, and reason about its actions or actions of others.

Modeling in the area of graphics

Modeling the human body from the view of computer graphics can be classified into four categories [59]:

- *Stick figure models* consisting of a hierarchical set of rigid segments (limbs) connected at joints. The complexity of these models depend on the number of limbs and joints involved.

- *Surface models* containing a skeleton and an external shape (the skin). The problem with this kind of model is the difficulty in controlling the evolution of the surface across joints.

- *Volume models* using a collection of elementary volume primitives (such as ellipsoids, spheres, cylinders) to approximate the structure and the shape of the body. Although better results can be obtained, these models lack adequate control mechanisms for a large number of primitives during animation.

- *Multi-layered models* containing the skeleton layer, intermediate layers (muscles, fat, bones), and the skin layer to simulate the body animation following the human body physical aspects. This modeling is hard but currently it is considered necessary for a realistic human body animation.

Multi-layered modeling with deformation

There are three basic steps in multilevel deformation modeling [59]: rigid body for skeleton, physics for muscle design and deformation, and skin generation.

The first step considers the basic structure modeling: the definition of the joints, their positions and orientations, and the geometric model that describes the body hierarchy.

Next, the muscles are attached to bones across joints and work like springs along *action lines* which have a fixed end point and a movable end point. All the muscle (surface) points, producing the mesh for the deformable model, are controlled by three different forces: elasticity force, curvature force, and constraint force. The application of those forces give new positions for the mesh model.

The third layer is equivalent to the human skin. To smoothen the skin surface, B-spline patches are used which are sampled on semi-regular cylindrical grids with ray-casting method.

Applications of the multi-layered modeling with deformation can be found in [16, 24, 47, 57]. Figure 3.11 and Figure 3.12 are two examples.

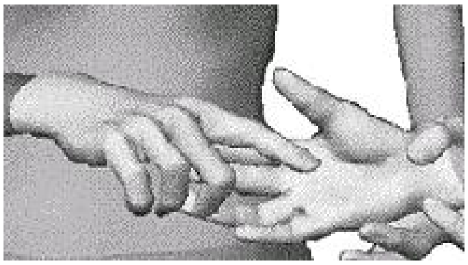


Figure 3.11: Deformable hand model (taken from [57]).

Figure 3.12: Deformable hand model with some morphological variations of hands: different middle finger length, thickness of the hand (taken from [47]): .

3D deformable hand model with statistical methods

In this section we introduce a 3D deformable hand model using statistical methods [37, 38]. In fact, it belongs to the computer vision area, not to the computer graphics. The key problem of producing 3D statistical deformable models is how to do the collection of landmark coordinate data from a set of training examples (images). Heap and Hogg provided a physically-based modeling technique [37]: first a physical model of the hand is constructed from the MRI hand data; the model is then deformed, under the actions of various forces, to fit each training example to obtain new locations of the model vertexes; and finally those new points are used to construct a Point Distribution Model (PDM).

Under exertion of forces, physically-based models have the ability to deform. Consider a physical model which is composed of N 3D point masses whose motion over time is governed by standard Newtonian dynamics with two types of forces applied:

- *Internal forces*: the elastic forces under which the point masses interact with one another to drive the model toward a stable rest configuration.
- *External forces*: the point masses are "attracted" toward particular image features so as to fit the model to the image data. External forces comes from two sources:
  - *Guiding forces*: these are set up manually for the coordinates of prominent object features in the training images in order to help the model find its approximate destination.
  - *Image forces*: these are exerted on vertexes to drive each vertex toward a "good" position.

By using Newtonian laws of motion and discrediting the action time, the new position of the model can be calculated from its previous two positions. This iteration can be repeated until a good result is obtained. Figure 3.13 gives an example of this process: a mesh with 498 vertexes of hand. This mesh was deformed from an initially random position to fit MRI hand data.

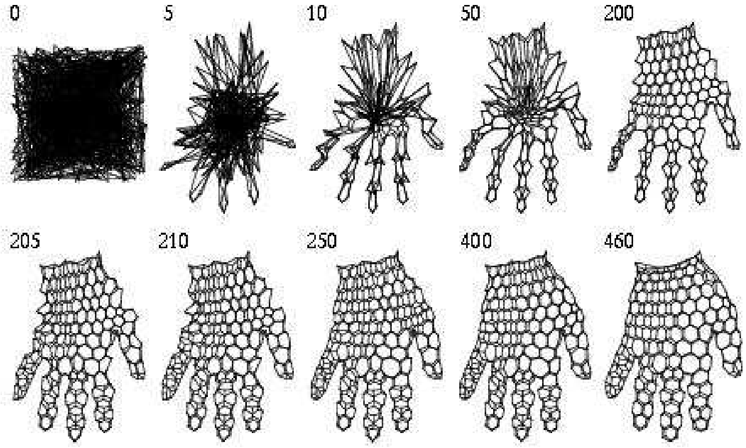


Figure 3.13: Deforming a hand mesh from an initially random position to fit MRI hand data (numbers show iteration) (taken from [37]).

For each training example, a corresponding hand mesh can be thus produced. The resulting meshes are then used to construct a statistical deformable hand model (a PDM). A PDM is built purely from the statistical analysis of a number of examples of the objects to be modeled (in this case, the hand images). Cartesian coordinates of N strategically-chosen landmark points are recorded for each image of the training examples. Thus each training example can be represented by a vector whose size is $3N$ for a 3D model. After aligning the training examples (translation, rotation, and scale), the mean shape is calculated by finding the mean position of each landmark points. The modes of variation are found with Principal Component Analysis (PCA) on the deviations of examples from the mean and are represented by orthonormal variation vectors. Therefore, the result is given by a mean shape and a set of orthogonal deformations which can be combined to produce a range of valid hand shapes. Figure 3.14 displays two such significant modes of deformation.

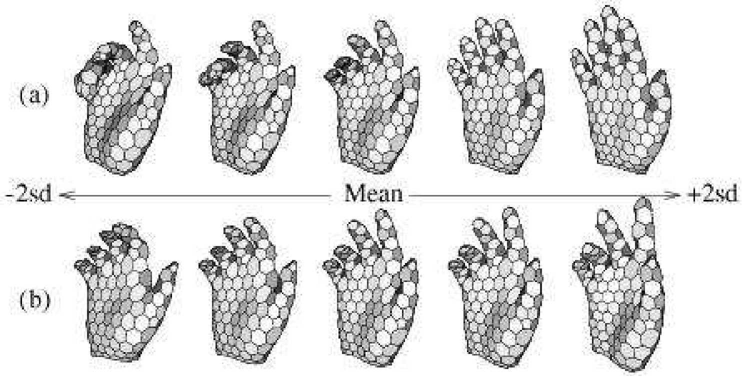


Figure 3.14: Two modes of variation of a PDM hand model (taken from [37]).

PDM is a very promising technique in modeling and tracking objects in computer vision [38]. Non-linear PDM methods and some of its application have been more fully discussed in [25].

3.4.2 Hand Modeling in the Computer Vision Application

Compared with the hand modeling and animation in the area of graphics, which emphasize the realistic rendering display during the process of animation, hand modeling in computer vision focuses on the realization of hand functions under different hand configurations and does not care much about the hand visualization. In the following we will introduce some examples of hand modeling in the application of computer vision projects.

Kuch and Huang presented a lifelike hand model [49, 50] (Figure 3.15a) which articulates in a realistic manner. They used cubic B-splines with about 300 control points to represent the individual surfaces of the palm, fingers and thumb. A total of 23 DOFs are included in the model. Both hand constraints and model calibration, based on the anatomical analysis, are also considered. This model has been used for hand tracking, hand articulation analysis [53, 82].

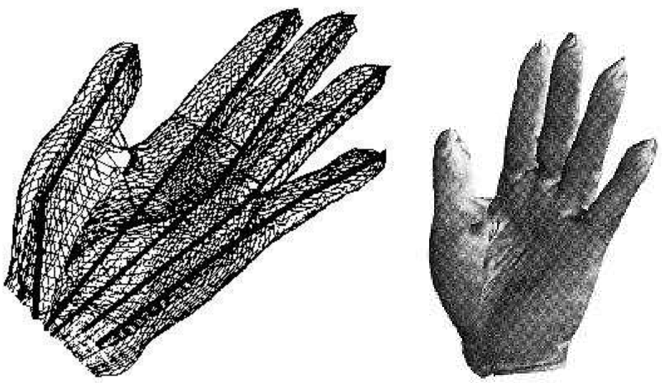
A simple graphic hand model [30] (Figure 3.15b), based on anthropometric measurements on hands of 32 subjects, was created in a CAD package CATIA. It was made up of 24 solid segments with 23 DOFs at 17 movable joints. The model was used to help a designer display hand-held products and controls in use, for example, grasping a ball.

(a) (b)

Figure 3.15: Hand models used in computer vision: (a) Two views of a fully calibrated hand model (taken from [50]); (b) A simple graphic hand model in default position (taken from [49]).

A new approach for modeling the human hand is proposed in [83] with the consideration of the dynamics and the natural constraints of the motion and the shape of hands. This dynamic hand model (Figure 3.16) is composed of a skeletal model and a skin model. The former is modeled following a hierarchical structure that consists of rigid links and joints. The skin surface model, initially based on the digitized 3D shape information of a real human hand, is generated following the skeleton link movements with a certain deformation. The model contains about $8,000$ triangle meshes.

An improved anatomically-based hand model was constructed For use in American Sign Language [55] (Figure 3.17a). The joint rotations in the model are based on the bone and muscle configurations of the hand, and a forward kinematic solution is



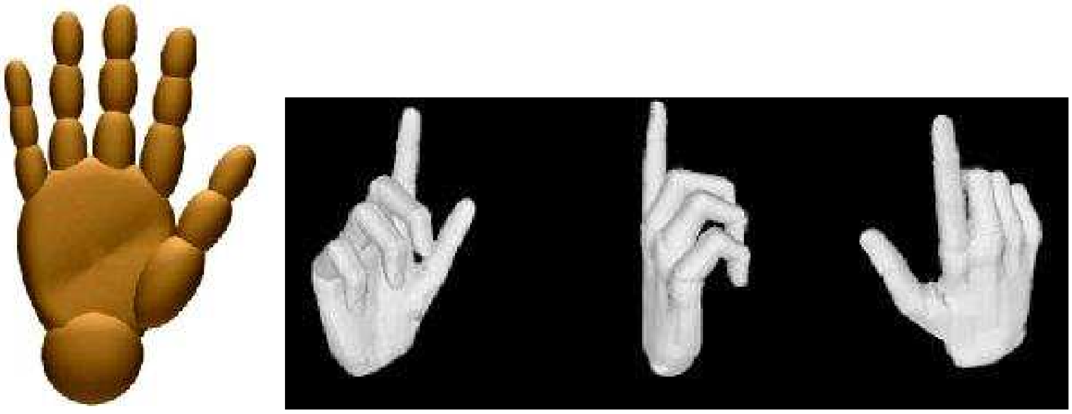
(a) (b)

Figure 3.16: A hand model with constraints[83]: (a) The wire framed hand model; (b) When in motion (one frame).

used to position the hand. It consider the base joint of thumb as a saddle joint with nontrivial rotational axes and centers.

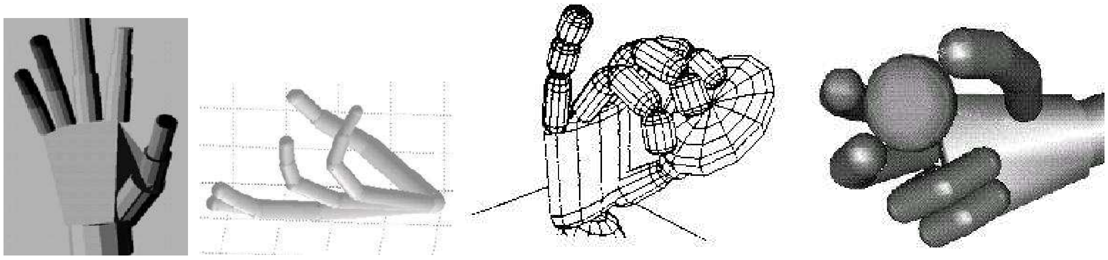
Another anatomy-based approach was proposed [42, 43] to model and animate hand motion gestures by dynamic simulation. A gesture coding system, named HACS (hand action coding system), was combined to codify hand gestures and to describe high-level hand behaviors in terms of hand muscle activation commands. It can generate and animate anatomically realistic intermediate motion or in-betweens from only the starting and the ending static postures of hand gestures. The interpolated postures produced by the synthetic hand model are anatomically realistic. Some views of this hand model can be seen in Figure 3.17b.

Other examples of using hand models include: estimating hand postures with genetic algorithms in a 3D hand-model-fitting method [52] (Figure 3.18a); 3D hand posture estimation with eight 2D projected feature points from a single 2D image [29] (Figure 3.18b); and in the analysis and application of hand grasping [65, 67] (Figure 3.18c and Figure 3.18d).



(a) (b)

Figure 3.17: Hand models used in computer vision: (a) An improved anatomically based hand model for use in the animation of American Sign Language (taken from [55]); (b) Different rendered views of an anatomy-based hand model with point posture (taken from [43]).



(a) (b) (c) (d)

Figure 3.18: Hand models used in computer vision: (a) A synthesized 3D hand model in use for hand posture estimation (taken from [52]); (b) A hand model for the use of grasping analysis (taken from [65]); (c) A hand model for the use of grasping analysis (taken from [65]); (d) Hand grasping a ball (taken from [67]).

Chapter 4

Hand Modeling

In this chapter, a 3D human hand model, *virtual hand*, is proposed for the use in the virtual environment of computer vision applications. This model is based on the results of the anatomical studies on human hand mentioned in the previous chapter. With the *virtual hand*, we can produce infinite number of hand gestures required in the hand posture calculation and hand analysis and recognition. *Open Inventor* [35, 77, 78] is used to create the hand model and its animation (dynamic modeling).

Basically, *virtual hand* is a rigid body, a combination of thousands of polygons in predefined order. A certain kind of deformation is considered for thumb and finger movements in what we call *dynamic hand modeling*. This can provide better rendering for a more realistic hand than the models in the applications of computer vision research projects mentioned in last chapter.

4.1 About Open Inventor

Open Inventor is an object-oriented toolkit for developing interactive 3D graphical applications [35, 77, 78]. Based on *OpenGL*, it provides a library of objects (a set of building blocks) that programmers can use, modify, and extend to meet new needs through the use of subclassing and callback functions.

As Figure 4.1 shows, the Inventor toolkit consists of

- 3D scene database primitives, such as shape, property, group, and engine objects, used to create *hierarchical* 3D scene;
- interactive manipulators, including handle box and trackball for user interaction;
- components, such as material editor, light and examiner viewer, providing some high-level interactive tasks.

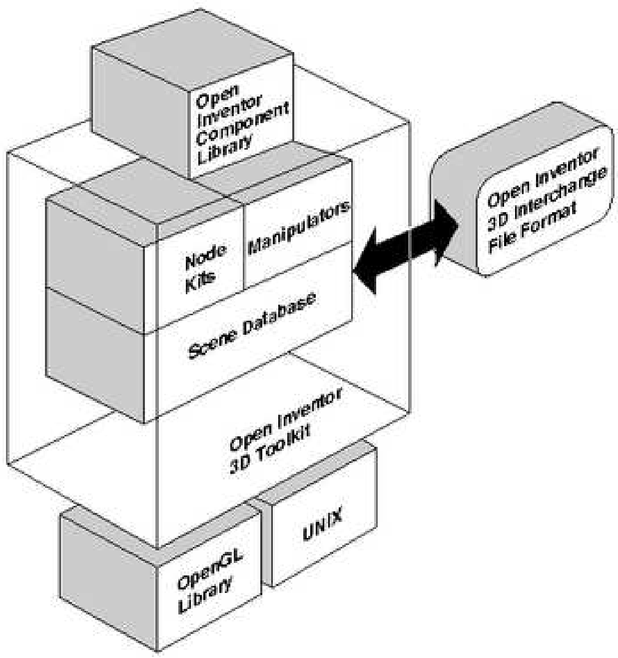


Figure 4.1: Inventor architecture (from [78])

Generally speaking, Inventor provides an object-oriented interface to OpenGL by using OpenGL for rendering. While OpenGL provides immediate-mode access to the frame buffer, thus the rendering is explicit, in Inventor, rendering is encapsulated in the objects just as other operations such as picking, reading, writing and calculating a bounding box. Special attention should be on this difference if animation design is involved in which Inventor *engines* are extended for specific applications (for example, in the design of dynamic hand model).

4.2 Static Hand Model

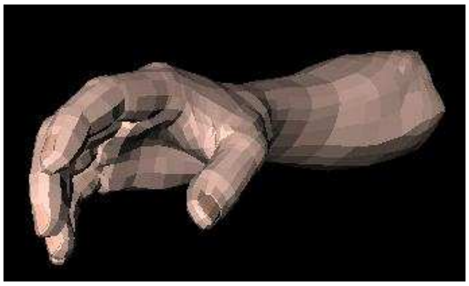
The starting point to create a hand model is to get the raw hand data of reasonable size. Usually the data includes the 3D positions of the points that consist of hand and the connections among the points. While some companies provide hand model and data with various resolution (i.e. size), other sets can be found on the Internet for free.

When the data is ready, normals for vertexes or polygons should be calculated depending on the rendering approaches, although in some cases there is no need to provide normals and the rendering mechanism can automatically bind the vertexes with corresponding normals. But for preparation of dynamic modeling where the hand is not considered as a whole object but as a combination of different rigid objects (fingers, palm), it is convenient to calculate normals at the beginning. In Open Inventor, we use *property nodes* to represent (define) appearance and qualitative characteristics of a scene (here, hand visualization). And the hand 3D point locations, connections, and normals provide the *metrics* part of the Open Inventor *property nodes* for the hand model.

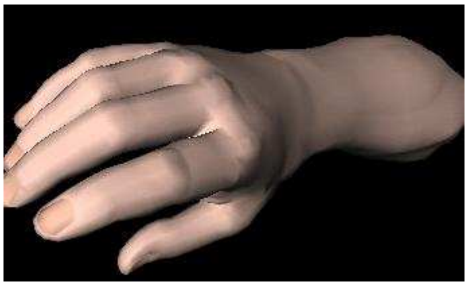
The *appearance* part of the property nodes is based on the *color value* given in each vertex. As for hand, only two different colors are needed: one for fingernails, the other for the rest of the hand.

The *transform* part of the property nodes is required if we want to manipulate (translate, rotate, scale ...) the hand in its display window.

With the 3D hand data we can reconstruct, with help of Open Inventor, the static hand model. Figure 4.2 shows the reconstruction of a hand model with about $3,000$ polygons. The forearm is included because movements at the wrist are inevitable part of the whole hand movement. Furthermore, without the forearm, the overall displaying effect will be awkward and and uncomfortable to look at.



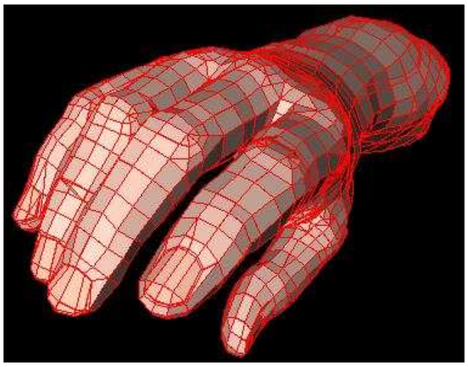
(a)



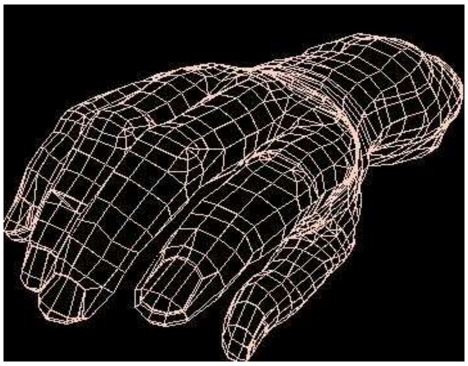
(b)

Figure 4.2: A static hand model: (a) the original hand, and (b) with normals applied.

Open Inventor provides different drawing styles which can be used to display the hand model(see Figure 4.3). These displays are very useful for dynamic modeling when we need to clearly identify the polygons for specific part of the hand.



(a)



(b)

Figure 4.3: Different displays of the hand model: (a) hand wireframe overlay display, and (b) hand represented as hidden lines.

4.3 Dynamic Hand Modeling

A good hand model should display as realistically as possible the hand gestures, which involve the different movements at joints: abduction and adduction, flexion and extension, and even rotational (twist) movements. The static hand model, as shown above, is a rigid object made up of thousands of polygons. Naturally, this rigid hand model can be disintegrated into hand parts. These parts have different motions and thus fill different locations in space and take various time sequences, allowing for a new configuration by incorporating those space and time features. The following concerns with the issues of this dynamic hand modeling: partitioning of static hand model, movement design of hand parts, and integration of hand parts.

4.3.1 Division of Static Hand Model

Hand gesticulations can be seen as the movements of various hand parts at different hand joints. It is reasonable to take these joints as dividing points for the disintegration of hand. As Figure 3.9 suggested, it would be a good starting point to divide the hand into 17 hand parts: 3 for thumb and each of the fingers, one for palm and one for forearm.

On the other side, from the programming view of Open Inventor, it is beneficial to use the Inventor's *SoSeparator* nodes to represent hand parts of the highly articulate hand and hierarchical hand structure: transformations at parent node level will have effect on its children nodes while siblings have no affections with each other. Figure 4.4 gives the programming design architecture with Open Inventor symbols. In this chapter and next one, Open Inventor scene graph symbols and Inventor-specific terms for its classes, functions, data structures) will be used without explanation. Refer to [35, 78] for their interpretation.

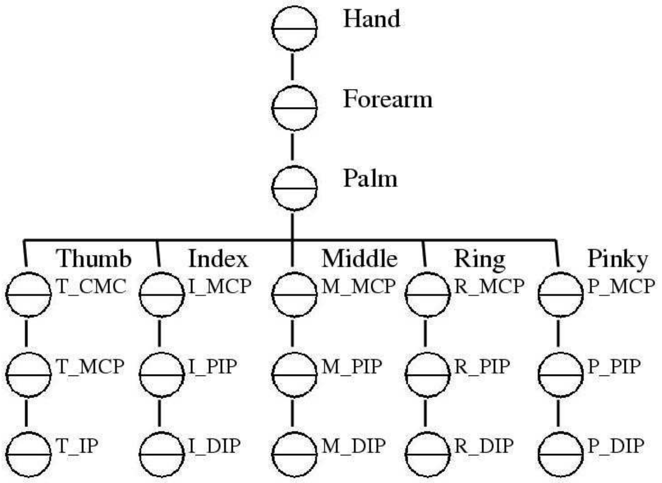


Figure 4.4: Dynamic hand modeling architecture (using Inventor scene graph symbols).

The architecture is a tree structure in which the parent node and children are defined just as in the tree of data structure. Furthermore, each node can represent *both a joint and a part* (segment) of the hand. This is advantageous to the programming design for the movement of the hand. When only movement is considered, the node is a point through which there can be found different rotational axes for the motions. When a hand part is to be rendered for display, the node is taken as a rigid object. For example, node *T_CMC* can be seen as a hand part (the thumb base) and the center (point) of the border between the palm and the thumb base. It is rendered as a common thumb base in display and rotates at a point for its abduction and flexion with 2 DOFs (Degrees Of Freedom). *Palm*, also as a hand part (displayed as a common palm) and a point at wrist with 6 DOFs: 3 for abduction, flexion, and rotation, and 3 for its translation in 3D space (this translation can been considered as the result of the rotational movement of the *forearm*). If we regard *forearm* with 3 DOFs we will not consider *palm*'s translational movement.

The naming for the nodes is same as in the Figure 3.9. That is: *T, I, M, R, P* for thumb, index, middle, ring and pinky (little finger); finger's *DIP, PIP, MCP* are for the tip, middle, and base of a finger; thumb's *IP, MCP, CMC* are for thumb tip, middle, and thumb base.

In order to partition the static hand model we have to number the vertexes and render them with the model. We then collect the possible edge vertexes that make the border of two adjacent hand parts. Now we can render any hand part or any combination of some hand parts.

Figure 4.5 is the index finger and Figure 4.6 is thumb base part, all their vertexes are numbered.

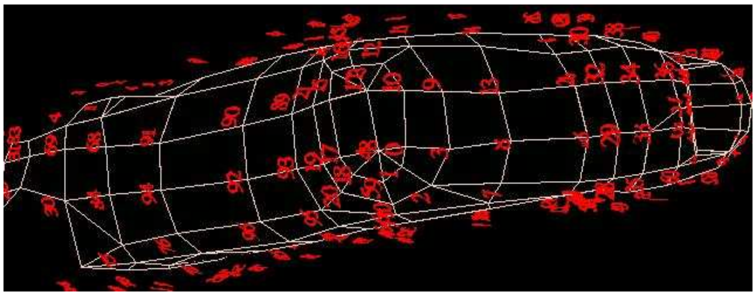
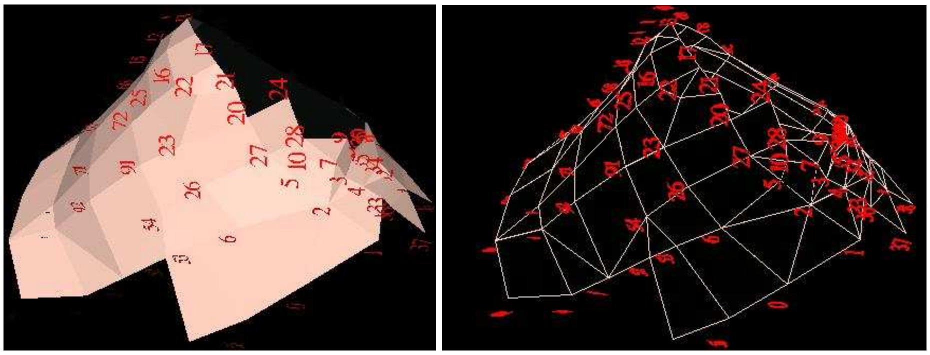


Figure 4.5: Index finger vertexes marked with numbers.



(a) (b)

Figure 4.6: Thumb base displayed in: (a) polygons, and (b) hidden lines.

With the hand disintegrated into 17 parts, we can deal with each part and consider the movement of each part.

4.3.2 Movement Design of Hand Parts

When involved in motion, a hand part will rotate around the different axes (of abduction, flexion, and rotation) for a certain angle. The rotation can be defined with the Inventor's data structure *SoRotation*, *SbMatrix*, and the combination of a *SbVec3f* for axis and a *float* for an angle. These data structures are implemented in Inventor as classes. Also a vertex in space is an object of the class *SbVec3f*. These classes provide many methods that can be employed for the vector and rotation calculations.

The bordering points of hand parts are used to compute the direction and location of the rotational axes of a hand part. And the center of the hand part, the definition and measurement of the *direction* of the hand part can also be computed from (the distribution of) all the vertexes of the hand part. Sometimes the direction of a hand part can be obtained from the centers of two adjacent hand parts. The center and direction of the hand part can be utilized to adjust the rotational axis.

Because each hand part is treated as a rigid body and the position given by static model is considered as the *starting zero position* of the dynamic model, different *SbRotation*s are generally required for flexion, extension, abduction, and adduction. Figure 4.7 shows an example of the index middle part involved in fexion and extension (around two different axes). The same case occurs in the abduction and adduction.

After those rotational axes have been determined, rotational angles can be applied to the hand parts. Figure 4.8 shows the effects of the extended and flexed virtual hand.

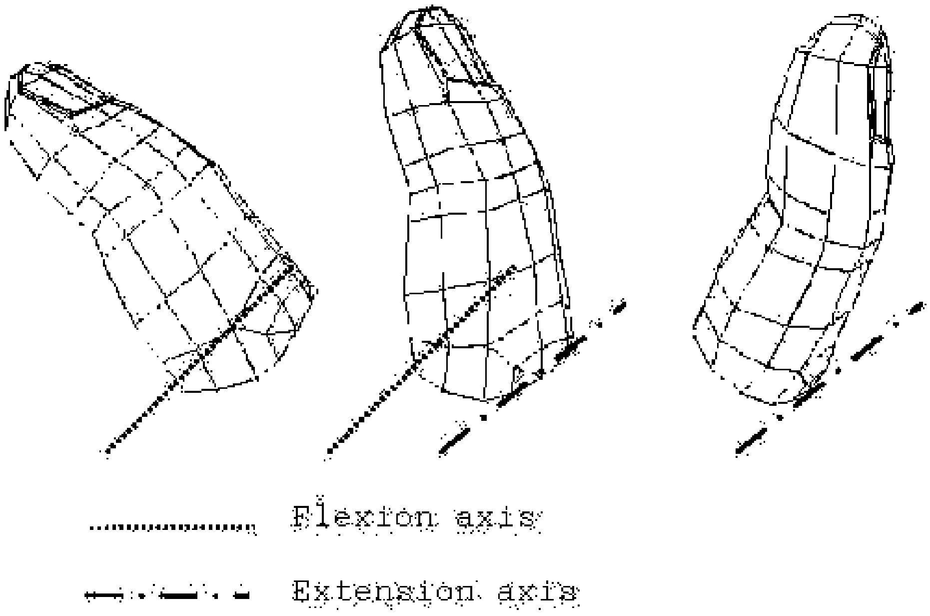
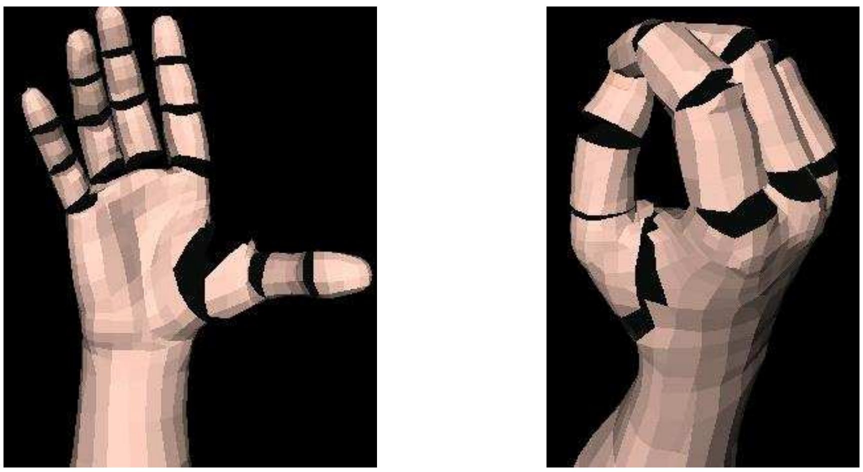


Figure 4.7: Hand part rotates around axes: here, index middle part (*I_PIP*) rotates around flexion and extension axes.



(a) (b)

Figure 4.8: Hand model in motion with "holes" or "cracks": (a) a stretched (extended) virtual hand; (b) virtual hand in flexion.

“Holes” or “cracks” are produced in the hand as the results of the rotations of the *rigid* hand parts. To handle this problem, we relate a hand part with the hole left by that hand part and deal with the hand part rotation and hole patching at same time. Figure 4.9 gives the programming design concept with the use of Inventor. All the nodes except the *Hand* node in Figure 4.4 are treated as *similar one* which is here represented as node *XFinger*. And *XFinger* in turn is made up of two nodes: node *XFinger_Sep* for the hand part and node *XFinger_xSep* for the hole patch resulted from the rotations of the hand part.

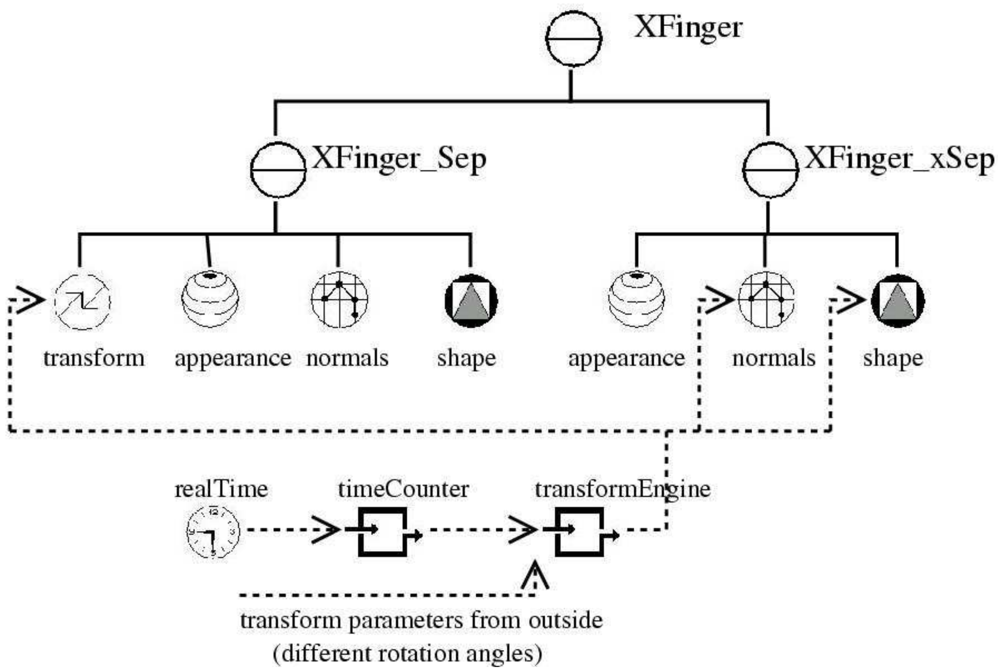


Figure 4.9: Hand part modeling using Inventor nodes.

The main nodes of *XFinger_Sep* and *XFinger_xSep* include:

- *transform*: rotational transformations for hand part. When this parameter is renewed, the hand part automatically rendered. This property node has several *fields* which connect to the output of *engines*, thus making automatic rendering possible. This mechanism of rendering is different from OpenGL's. Inventor's *field, engine* implementation can be found in [35, 78]. Note that there is no *transform* node in *XFinger_xSep* (hand part patch) node, because its *shape* node consists of the patch's 3D vertexes and those 3D coordinates are obtained from the engine node *transformEngine* (see **transformEngine** below).

- *appearance*: a property node that provides the material characteristics of the hand part and hand patch, in the form of colors for the vertexes.

- *normals*: an Inventor metric node that provide normals and normal binding methods for the *shape* node. In *XFinger_Sep*, normals are embedded in the *shape*'s automatic rendering process without updating. As in the case of *XFinger_xSep*, *normals* are connected to the *transformEngine*'s output, updated for every possible transformation. (see **transformEngine** below).

- *shape*: a node that determines hand part and patch's shape. In the case of *virtual hand*, it contains a set of 3D vertexes. The difference between *XFinger_Sep*'s *shape* and *XFinger_Sep*'s *shape* is same as that in the above *normals.*

- *realTime* and *timeCounter*: nodes that offer time control parameters for the virtual hand's animation. For a hand part and its corresponding patch, we can use them to control their performance over time period (cycle): the abduction and flexion angle ranges over these rotational steps (how many steps, how many degrees per step respectively).

A special engine, *transformEngine*, is a key point in patching up the "holes" or "cracks" and a *drive* engine in virtual hand animation, and thus deserves more attention.

transformEngine

In Open Inventor, *engines* are used to animate parts of a scene and to constrain certain elements of a scene to some other ones. In *virtual hand*, an extended Inventor engine, *transformEngine*, is constructed to control virtual hand animation and movements of the hand parts (and their hand patches). In the programming design, *transformEngine* is implemented with two main C++ classes derived from *SoEngine*: one for thumb base, another for other hand joint nodes. They differ only in the method of producing the patch 3D vertexes for filling up the "holes".

The function of *transformEngine* is to provide movement *drive* to Inventor's rendering of virtual hand and to produce the 3D point coordinates for hand part patches. Its inputs include:

- *rotational axes and angles of hand parts*. By rotational axes, we mean the axis of flexion, that of extension, abduction, and adduction. The engine can determine which pair (flexion–extension and abduction–adduction) of the axes based on the degree of angle.

- *the edge of hand part* which consists of the 3D point coordinates and their normals of the bordering vertexes shared by two adjacent hand parts. But in the case of the thumb base, two groups of vertexes have to be considered: one on the border with palm and the other with the middle part of the thumb (*T_MCP*).

- *animation parameters* that come from the output of the *timeCounter* engine.

- *rotational angles* that are provided directly by the user.

The outputs include:

- *transform* in Inventor-specific form to "drive" the hand part to rotate: how and how much.
- *3D shape coordinates* for hand part patch to fill up the holes.
- *normals* for the above 3d shape coordinates of the hand patch.

How to produce the hand patch 3D points? Generally, from the edge vertexes of hand part and the transformation axes and angles. For every hand part except for the thumb base, all its edge points, following the transformation of the hand part (the edge a part of the hand part), will leave trajectories in space (one edge point corresponds one trajectory). Those trajectories can be sampled to produce a set of 3D points which are then combined in a certain order to form a 3D surface—the patch that fill up the holes. Normals can also be calculated from those points.

As for the thumb base, this sampling of the trajectories did not give good results. An interpolation with limited deformation was used and an acceptable results were obtained. As above mentioned, there are two groups of edge points. One group will follow the transformation of the thumb base while other remains fixed on the palm. Between a pair of (two) edge points (from the two group), several points are interpolated linearly first. And then some of the interpolated points which are close to the edge are extended toward outside of a boundary which is made up of the interpolated points with same interpolation factor. Figure 4.10 gives an explanation of this method. Those deformed interpolated points can be arranged in some order to form the thumb base patch, filling up the holes between rotated thumb and palm. Normals for those points can be computed easily.

Now that all the hand parts are moved to their new place according to their

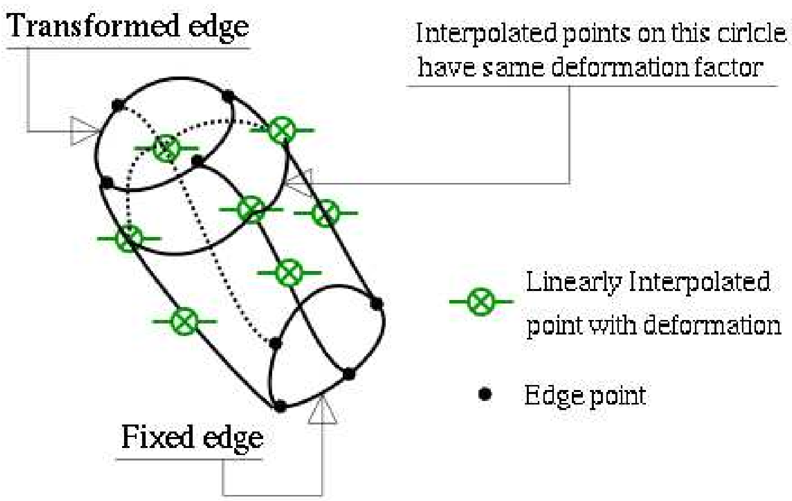


Figure 4.10: The interpolation and deformation of the thumb base patch points.

rotational parameters and all their corresponding hand part "holes" or "cracks" an be filled up with those patch points calculated from above, the static hand model has been transformed, giving a dynamic hand model.

4.3.3 Dynamic Hand Model

We are now ready to render a dynamic hand model after the above procedures. Only some programming issues with Open Inventor are to be considered. Figure 4.11 gives an general design concept (refer to Figure 4.9 and Figure 4.4 for overall understanding).

We have to insert some Inventor nodes to finish the virtual hand rendering:

- *camera*: it necessary to add a camera for us to be able to see the rendered model.
- *light*: at least one light source is needed.
- *transform* and *appearance*: same meaning as explained before.

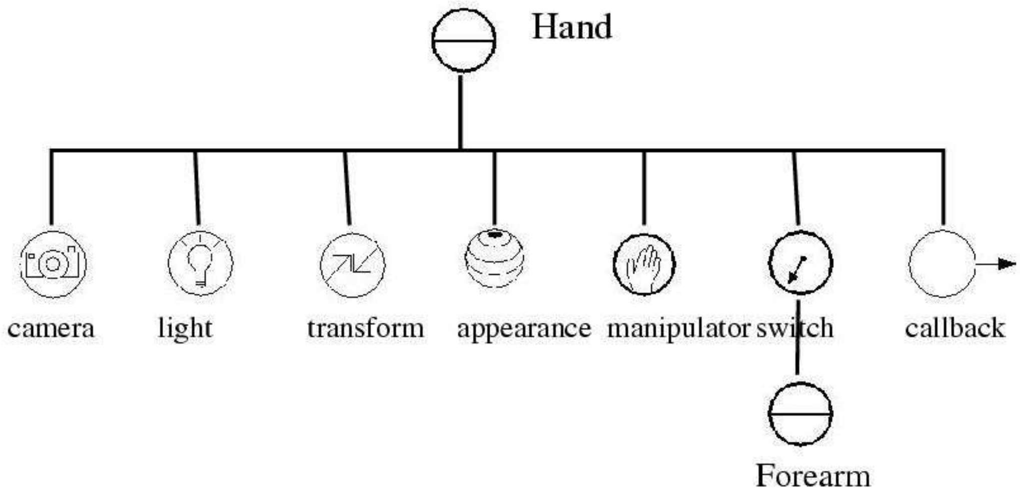


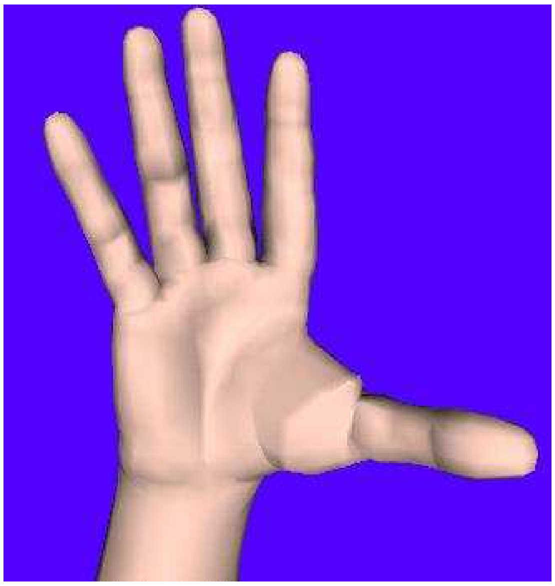
Figure 4.11: Dynamic hand modeling using Inventor nodes.

- *manipulator*: a very powerful node in Open Inventor that can be used to dynamically interact with the rendered model.

- *switch*: by encapsulating the whole hand node with a Inventor's *SoSwitch* node, we can switch on and off a hand easily. It will be used in the hand virtual environment where hands operate on virtual objects.

- *callback*: by setting up callback functions, we can apply Inventor *actions* to the rendering process (though this is not the only way to apply actions). We implemented a callback function to read virtual hand's GL *ViwingMatrix*, *ProjectionMatrix*, *ViwingMatrix*, and other inner rendering parameters.

Two examples of the virtual hand in two gestures are given in Figure 4.12.



(a)

(b)

Figure 4.12: Dynamic hand model—*virtual hand*: (a) a stretched (extended) virtual hand; (b) virtual hand in flexion.

Chapter 5

Interface Design

After the dynamic hand model is created, we can use it to design hand gestures, imitate hand movement in virtual environment according to the input of hand postures from cameras, and even take this artificial virtual hand as a substitute for real human hand in computer vision experiments on hand gesture analysis. Examples of applications will be given in next chapter. To accomplish all these tasks effectively and comfortably, the first step is to design an interface for the operations on the virtual hand. The goal of the interface is to provide a *testbed* for research on hand in computer vision. In the following, general considerations of the interactive interface will be discussed first, and then the implementation and performance of the testbed will be given with results.

5.1 Design Issues

The basic concept in a virtual HCI environment design is the interaction between the user and the environment. In our testbed virtual hand accepts control signals from the user, adjusts to the corresponding gesture, displays the effect in the virtual environment, and sends back the results in some form. Figure 5.1 gives the design architecture of the virtual hand testbed. Some explanations for the design concept in the figure are given next.

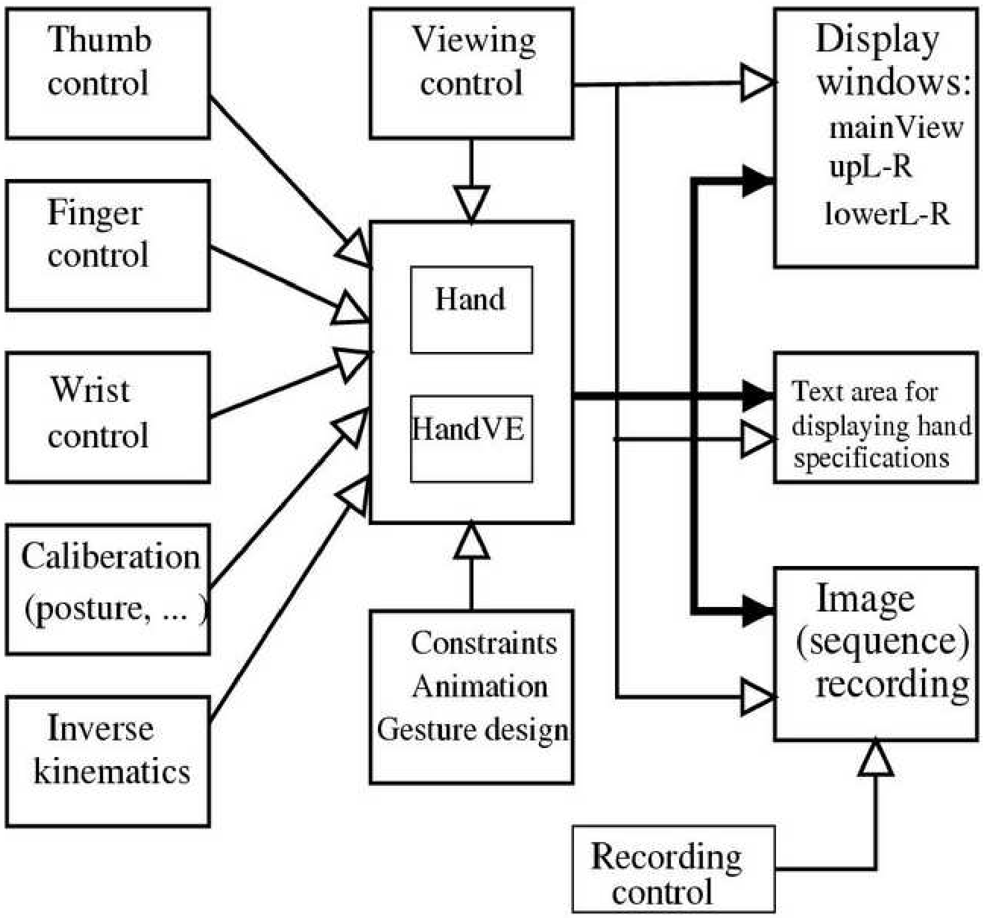


Figure 5.1: Virtual hand interface design architecture.

Hand and *HandVE* are the kernel of the virtual hand testbed. This kernel includes the dynamic hand model and an virtual environment that is required for experiments. The kernel accepts signals from:

- *Thumb, Finger, Wrist Control*: flexion-extension, abduction-adduction, and rotation (only for wrist) of the hand part at the appropriate hand joint.

- *Viewing control*: used for setting background, virtual hand's visualization (drawing) styles.

- *Calibration*: necessary as the first step if virtual hand is to represent the real hand movement, to estimate hand posture, and to test hand tracking.

- *Constraints, Animation, Gesture design*: some hand constraints should be applied to the virtual hand to make it look as realistic as possible. Also hand animation process design should embedded into the hand model during dynamic visualization. A gesture database can be constructed through the virtual hand for analysis and recognition of hand gestures.

- *Inverse kinematics*: used to study hand manipulation on some objects, for example, hand grasping a ball.

In the testbed, rendered scene graph is sent to 5 *Display windows*. One large display window and four small ones display the virtual hand and environment from different viewpoints. Virtual hand specifications, such as coordinates of hand joints, transformation matrices, will be output into *Text area for displaying hand specifications*. The rendering process of the virtual hand and environment can be recorded as a sequence of images and text files of hand specifications. The *Recording control* factors include frame control, image size, camera selection, etc.

5.2 Performance

The virtual hand testbed was implemented with *Qt*[12] and *Coin3d*'s *SoQt* interface library [3] under the development of *kdevelop2.14* [11] with *RedHat8.0*. No kdevelop-specific or Linux-specific function(libraries) are used, so the testbed can run under other platforms such as MacOS, Window.

Performance of and results from this testbed will be presented in this section following the design in Figure 5.1. A general outlook of the interface is shown in

Figure 5.2. The background for the interface and that of the virtual hand environment may be changed through the *Display* menu.

The interface shown in Figure 5.2 is roughly divided into two parts: the upper half for display windows lower half for *Qt Tag* widgets for controlling virtual hand and viewing, for *text area for displaying hand specifications*, and for testbed logos.

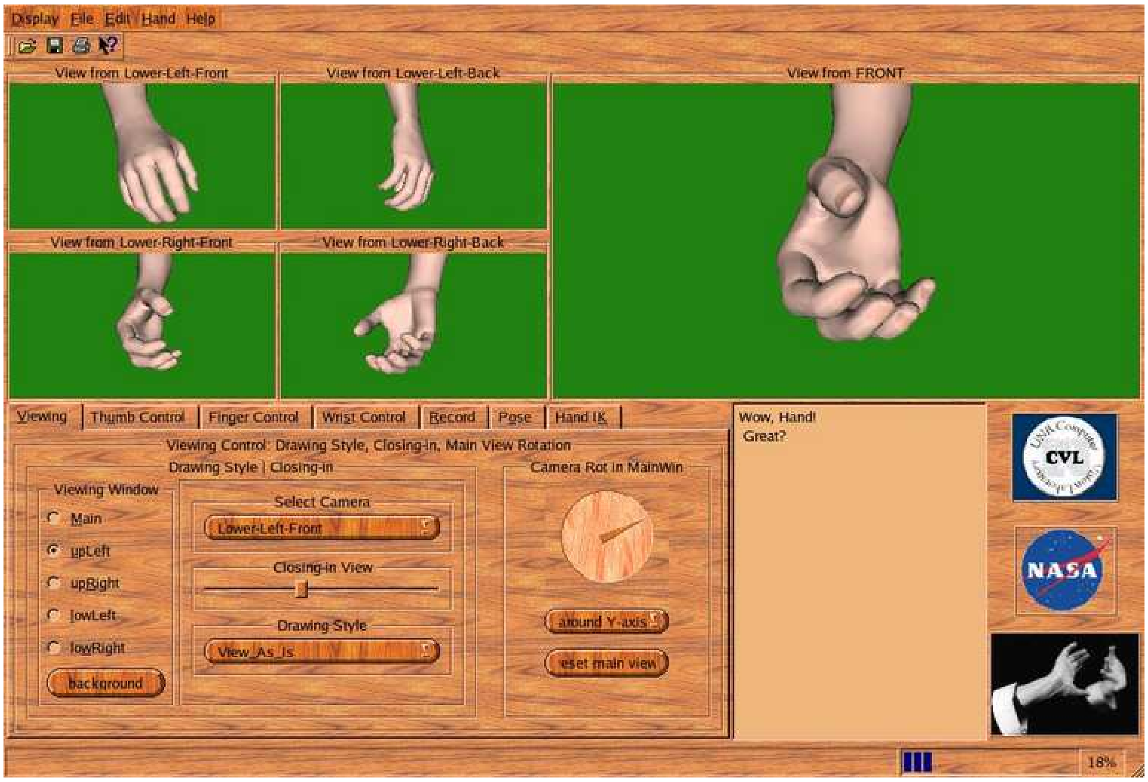


Figure 5.2: an outlook of the testbed interface.

The virtual hand can be seen from different viewing points through the display windows via different cameras in the virtual environment, with one window as main display area. The display effect can be adjusted through *Viewing control.*

Viewing control

The virtual hand environment is like a glove box where human hands manipulate objects. In the virtual environment, 8 cameras are set up in the 8 corners and another camera, not fixed to a certain location, can move around the virtual hand.

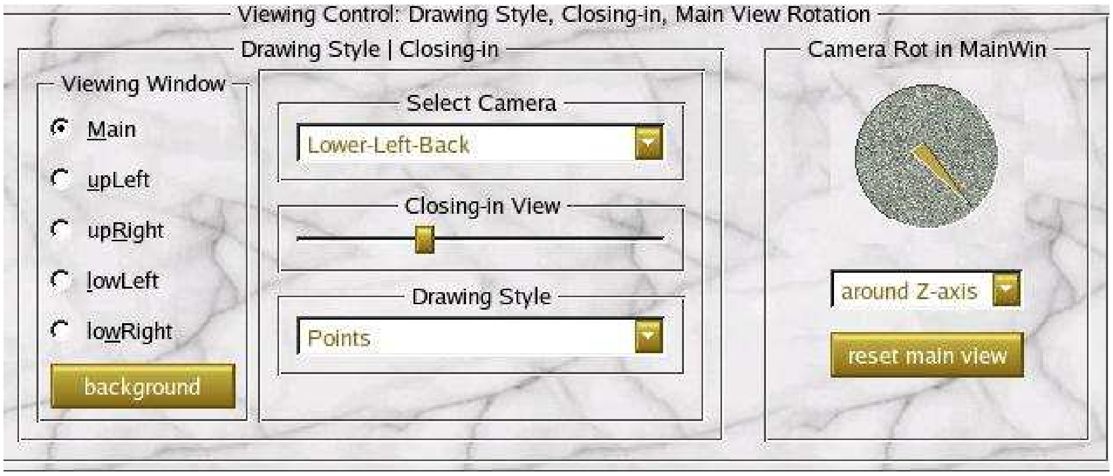


Figure 5.3: Viewing control panel.

Viewing control panel (Figure 5.3) is used to select which camera for displaying in a window, which hand rendering style, and to control the movable camera:

- *Viewing window*: choose which displaying window to modify viewing effect (close look or zoom, camera, drawing style).
- *Select camera*: choose which camera for the chosen viewing window (the movable camera named *Front*, the 8 fixed cameras named *Lower-Right-Front, Lower-Right-Back, ..., Upper-Left-Back*).
- *Closing-in view*: adjust the zoom for the chosen camera.
- *Drawing style*: the hand can be rendered as "View-as-is", "Wire-frame", and "Points".
- *Background*: to set up a particular background color for the virtual environment. All the displaying windows have same background.

- *Camera Rot in MainWin*: to rotate the movable camera around 3 axes.

- *Reset main view*: to set the movable camera to its initial position and direction.

Figure 5.4 shows the virtual hand rendered from different cameras with different foci and displayed in different viewing windows. Figure 5.5 displays the virtual hand in different backgrounds and rendered in different styles.

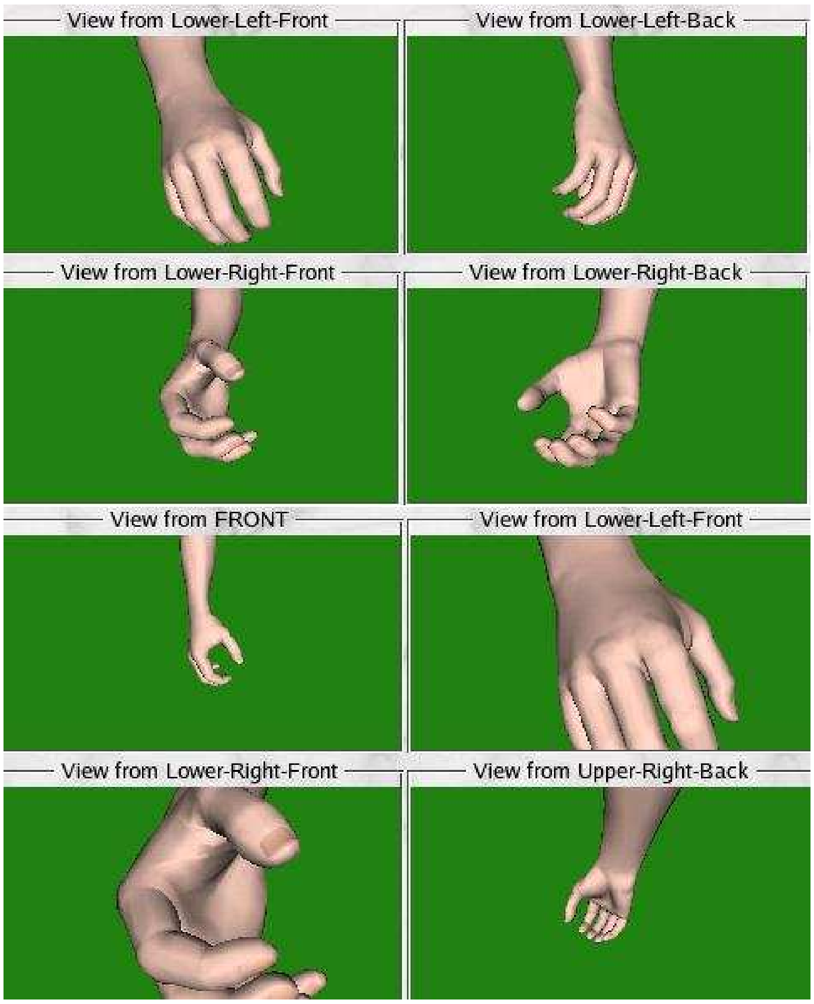


Figure 5.4: Virtual hand viewed from different viewing points.

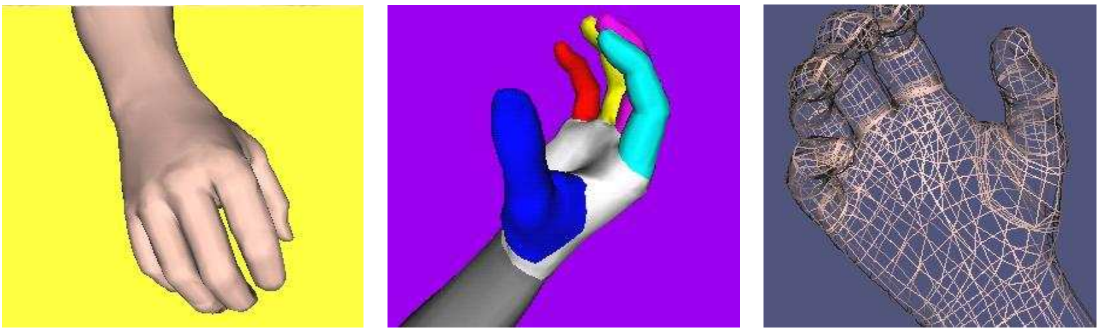


Figure 5.5: Virtual hand in various backgrounds and styles.

Thumb control

The thumb of our virtual hand has 5 DOFs. There are 2 DOFs for thumb base (CMC) thumb middle part (MCP) each: flexion-extension and abduction-adduction movements. The thumb tip (IP) part has only one movement: flexion-extension. There is one Qt *dial* or *slide* for controlling one DOF. Figure 5.6 gives the *Thumb control* panel.

Figure 5.6: Thumb control panel.

The *Thumb control* panel includes following control items:

- *IF/F*: used to set thumb tip flexion angles.
- *MCP/F*: for adjusting thumb middle part's flexion.

- *MCP/A*: controlling thumb middle part's abduction angles.
- *CMC/F*: for adjusting thumb base's flexion.
- *CMC/A*: controlling thumb base's abduction angles.

Figure 5.7 shows the virtual thumb's motion process when the thumb's joints have been given different rotational angles.

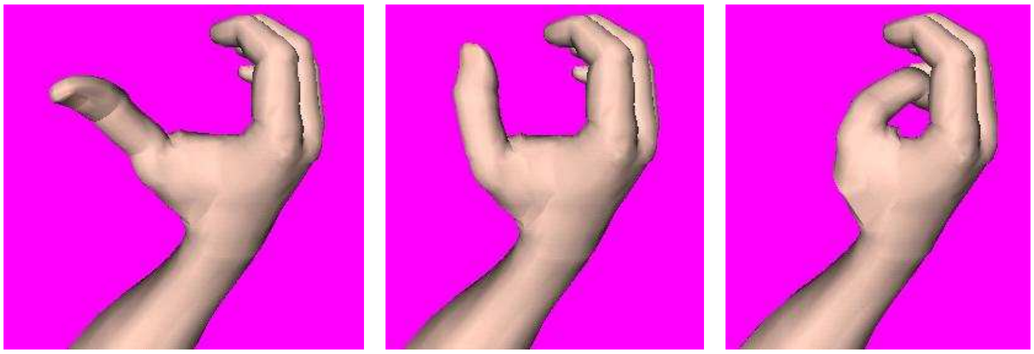


Figure 5.7: Virtual (hand) thumb in motion.

Finger control

Virtual hand has four fingers: index, middle, ring, and pinky (little finger). Each finger has 4 DOFs. There are 2 DOFs for finger base (MCP) flexion-extension and abduction-adduction movements. The finger middle part (PIP) and the finger tip (DIP) has only one movement each: flexion-extension. Figure 5.8 is the *Finger control* panel.

The *Finger control* panel includes following control items:

- *Fingers*: to select a finger whose joints need to be rotated.
- *DIF/F*: used to set the chosen finger tip's flexion angles.

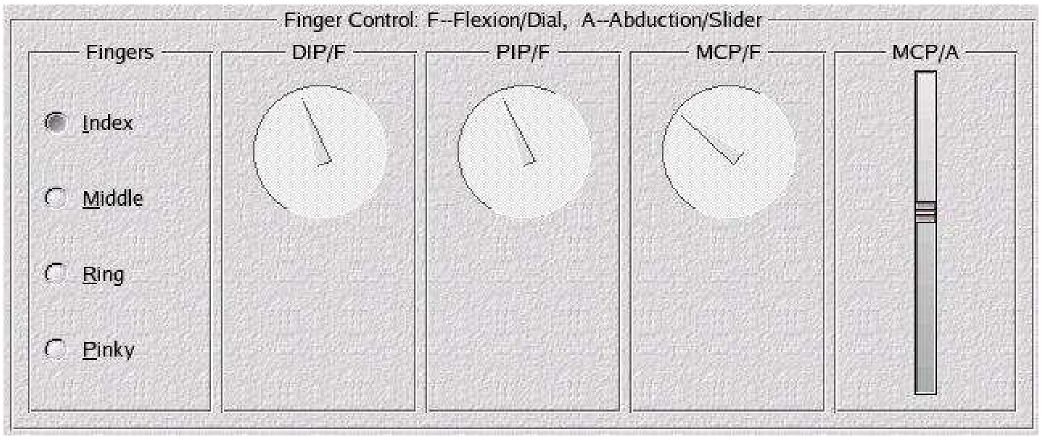


Figure 5.8: Finger control panel.

- *PIP/F*: for adjusting the chosen finger middle part's flexion.
- *MCP/F*: controlling the chosen finger base's flexion angles.
- *MCP/A*: controlling the chosen finger base's abduction angles.

Figure 5.9 exhibits the virtual index's motion process when index's joints have given different rotational angles while Figure 5.10 shows all the four fingers' movements. All these finger joints' settings correspond to different hand gestures (configurations).

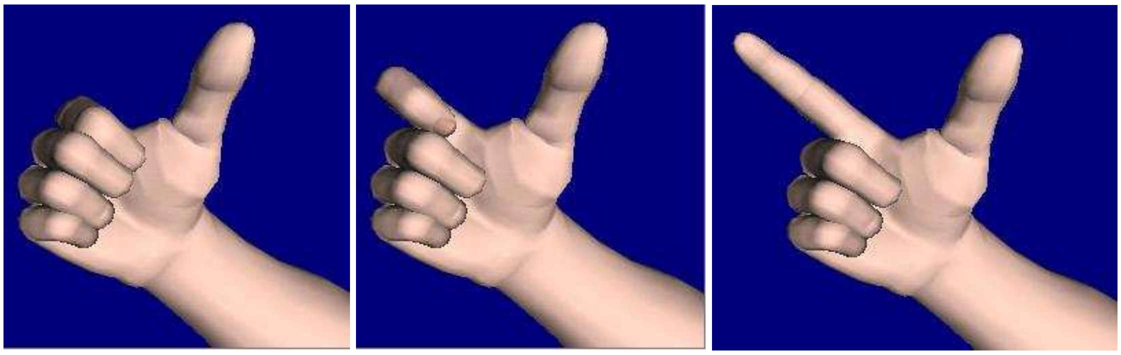


Figure 5.9: Virtual (hand) index in motion.

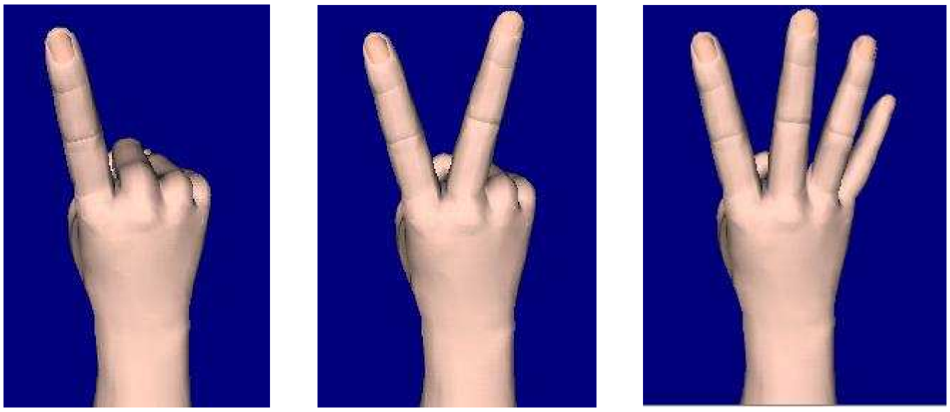


Figure 5.10: Virtual (hand) fingers in motion.

Wrist control

In the general case, a human wrist has 3 DOFs if the hand and forearm are considered at same time because the elbow joint rotation will provide translation movement for the wrist. But in the virtual hand testbed, the emphasis is on the *hand* not the arm, and the addition of the forearm is only to make the virtual hand and environment look more realistic. Therefore, the indirect translation as result of the forearm's rotation at elbow is considered as the *direct* translation of the wrist. So the virtual hand wrist has 6 DOFs: twist (rotation) around the center of the wrist, side-side movement, bend-extension, and translation along X, Y, Z. Figure 5.11 gives the *Wrist control* panel.

The *Wrist control* panel includes following control items:

- *Twist*: rotation around the center of the edge points of the between palm and forearm. This rotation in fact is forearm's twist/rotation.

- *Side-side*: hand's ulna-radius movement which can be compared to the abduction-adduction movement of fingers.

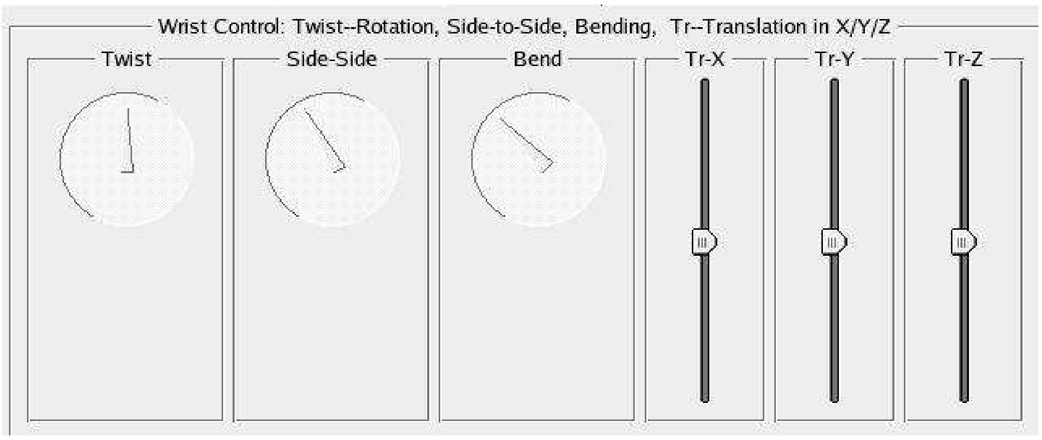


Figure 5.11: Wrist control panel.

- *Tr-X, Tr-Y, Tr-Z*: translations as result of forearm's movement. Adjusting these parameters cause the whole hand (including forearm) to move to a new location.

Figure 5.12 displays all these movements at the wrist.

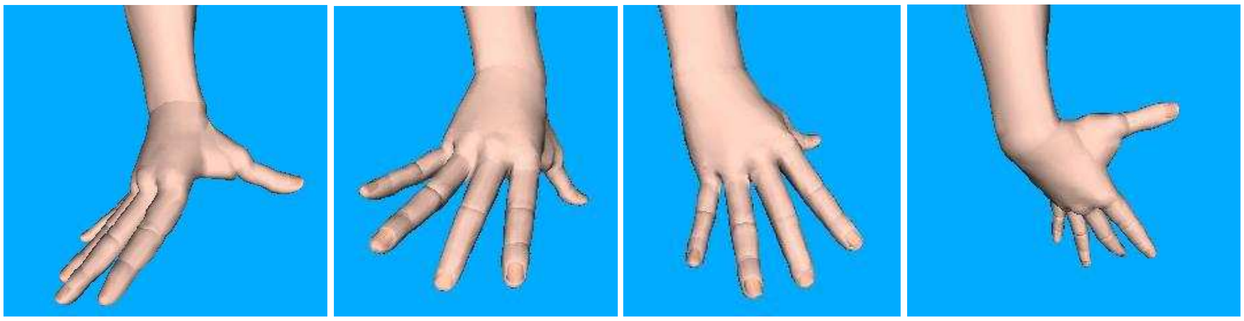


Figure 5.12: Virtual (hand) wrist in motion: twist, side-side, bending-extension, and translation in space.

Calibration

The virtual hand in the testbed must be calibrated before being used to represent the movement of real hand or hand model. By calibration, we refer to two processes:

1. *Coordinate system alignment.* Human hand (or artificial hand model) has reference coordinate system–the world coordinate system–in which it handles phys-

ical objects. And the hand and objects themselves have their own local coordinate system. On the other side, the virtual hand is placed in a virtual environment which has also an imaginative world coordinate system. To coordinate the events in the virtual environment with the those in the real physical world, the two world coordinate system must be at first aligned.

2. *Hand pose calibration.* Hand pose is defined by the hand joint angles and hand orientation. When the hand pose is estimated from the real hand or hand model, the virtual hand in the testbed should exhibit this change by the updated estimation, adjusting to the new posture. Other attentions should be paid on: (1) difference in the hand measurement between the two world coordinate systems, and (2) difference between the local coordinate systems of virtual hand and real hand relative to their world coordinate systems.

Figure 5.13 displays the preliminary work to hand calibration.

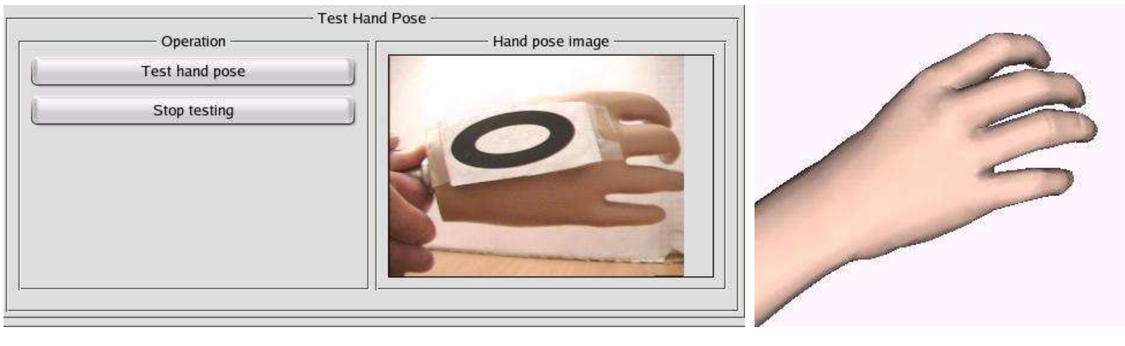


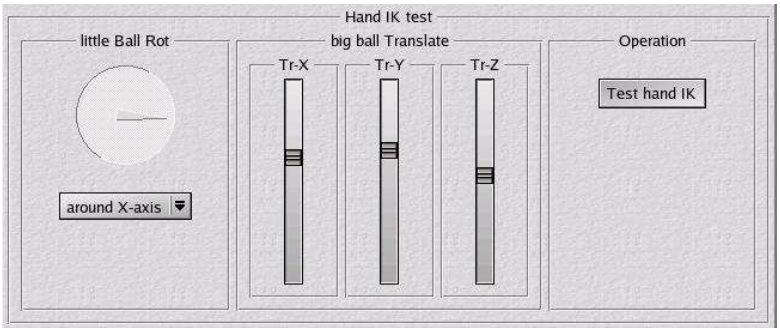
Figure 5.13: Virtual hand (posture) calibration.

Inverse kinematics

Inverse kinematics for a hand is a big a topic. Given an object in the workspace of the hand, there is no definite solution on the best way how the hand is to grasp it.

The highly articulated hand (with more than 20 DOFs) can not give a closed-form solution. But with simplification the hand configuration and decrease of the degrees of freedom, an approximation to the inverse kinematics on hand is possible. In some cases, forward kinematic methods can be combined into the inverse solutions.

Figure 5.14 gives a simple preparatory testing environment for inverse kinematics studies on hand: the virtual hand and a large ball and a small one.



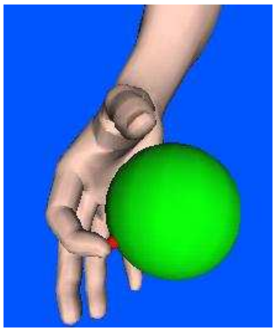


Figure 5.14: Virtual hand inverse kinematics testing.

Constraints, animation, and gesture design

Results from the studies on natural movements (gestures) of human hand provided a set of movement ranges and constraints on virtual hand. Limitations of hand joint motion were given in Table 3.1 and Table 3.2 and hand constraints were enumerated in the Chapter 2. All the hand joint movement ranges have been applied in the *virtual hand testbed* and some of the constraints such as those on finger DIP and PIP joints and middle finger constraint are considered in the model.

Figure 5.15 exhibits the constraints applied in finger movements. In the research on hand tracking and hand posture estimation, a color hand model is preferred for some algorithms. In the following results colored virtual hand will be displayed from time to time.

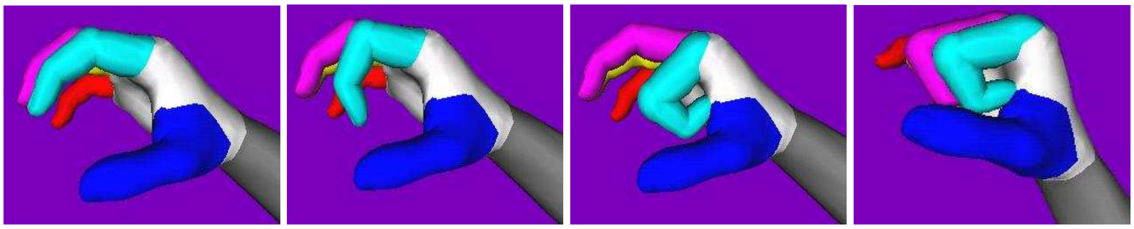


Figure 5.15: Virtual hand constraint application: the first two images show constraints on index DIP and PIP and last two on the middle and ring's DIPs and PIPs.

Virtual hand animation is implemented through the engines *realTime* and *timeCounter* (see Figure 4.9). For every hand part (joint), its flexion-extension and abduction-adduction ranges, the time duration and and time distribution over the duration, are set up. We call those parameters *animation patterns*. Once these are set you then turn on the engines' switch, and the hand will automatically be rendered following the defined animation pattern.

For animation of a series of different gestures, two data structure should be defined. One responds for animation pattern for a specific gestures and another one for a series of gestures that are combined to make up the animation. We are working on making this more straightforward and easy to use.

Figure 5.16 shows part of a very simple animation process which is from the extended hand to a fist.

Gesture design is concerned with a simple data structure which stores the rotational parameters of all the hand parts of the virtual hand. Gesture storage and retrieve is management of the gesture databases. In fact, all the above results can be seen as some kind of hand gestures. Implementation of the gesture management is also under design.

Figure 5.16: Virtual hand animation: rendered in line styles.

Image recording control

The rendering process of virtual hand in animation can be recorded. This feature is very useful if we want to make a movie of the animation. It is also helpful to record one image of a virtual hand gesture because an image recorded this way has better quality than those caught from a screen or from a real camera. The recording process is controlled by the inner time counter provided by *SoQt* timing mechanism. The rendering will "stop" from frame to frame in order to allow all the viewing images in one frame from all the chosen cameras have stored in image file. This is because the rendering is much faster than the recording images to hard disk.

Rendering parameters of virtual hand for each frame can also be recorded in text file. The parameters include hand joint angles, rendering matrices.

Figure 5.17 is the rendering process recording control panel. Sometimes virtual hand images rendered with only basic light model (diffuse object color, not the default

Phong light model) are useful in computer vision research on hand, so this case is considered in the recording control. Figure 5.18 displays some virtual hand images rendered with such basic light model in the environment.

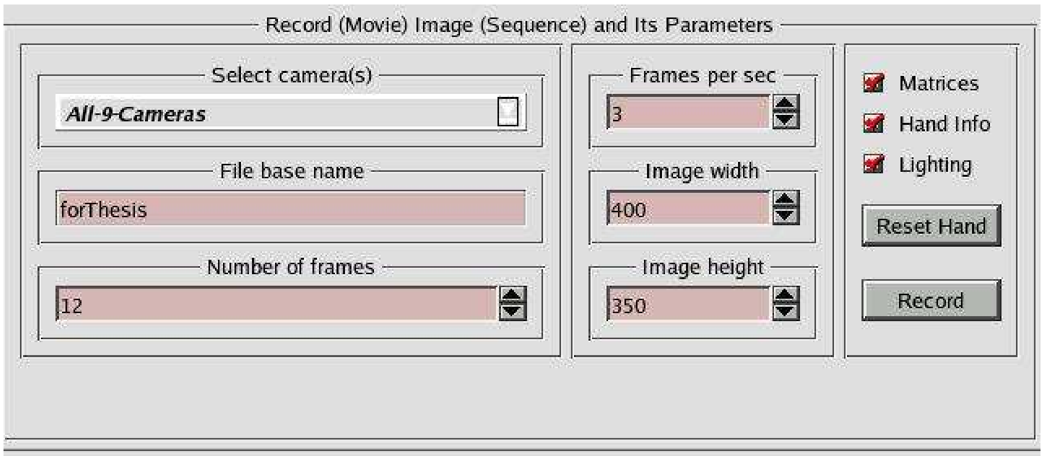


Figure 5.17: Image recording control panel.

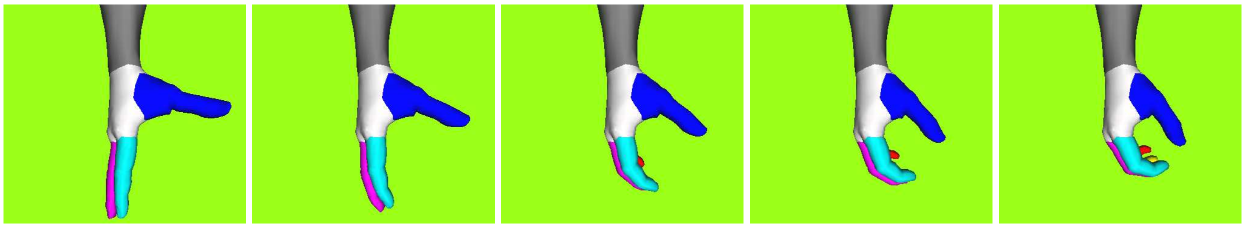
Figure 5.18: Virtual hand images rendered with basic light model (only diffuse object color) in environment.

Image recording of virtual hand has the following features:

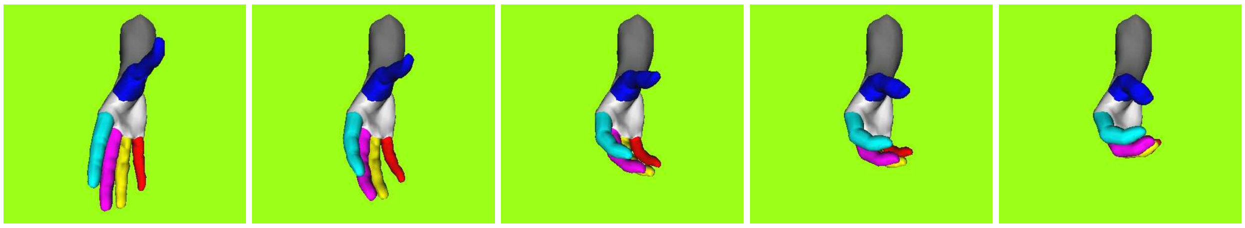
- Select cameras. Any number of the 9 cameras can be chosen. Thus a frame of virtual hand can have this number of images viewed from the chosen cameras.
- Set up image size, image file name.
- Define recording frames and frequency (how many frames per second).

- Lighting switch.
- Reset hand initial position button.
- A switch to record in the text file the hand joint angles (current values and the movement ranges).
- A switch to record in the text file the Open Gl rendering parameters of the virtual hand: its position, box size in which the virtual hand resides, camera parameters, rendered area size, model matrix, and projection matrix.

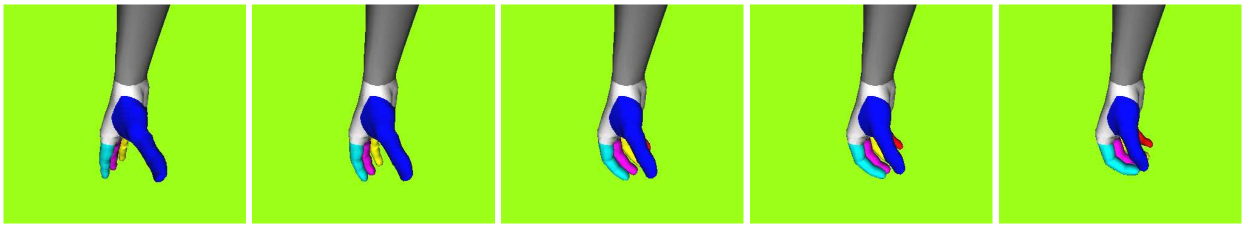
Figure 5.19 is a part of virtual hand images recorded during an animation. Only five frames were sampled from the original image sequence and the sampled images came from five different cameras. The original images were recorded at speed of 3 frames per second, with image size of 400×350 under lighting condition.



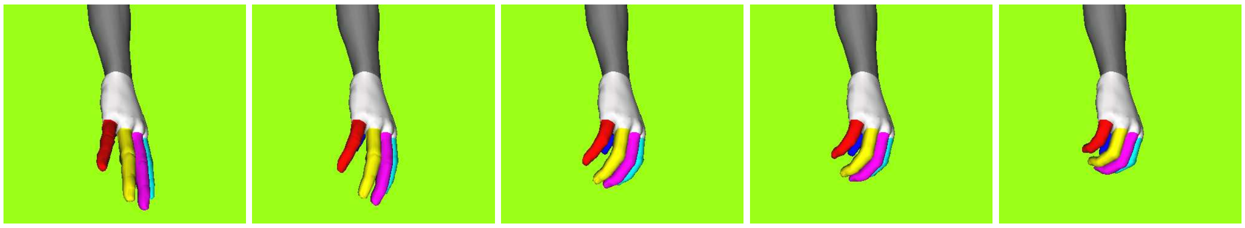
(1) From *front* (movable) camera.



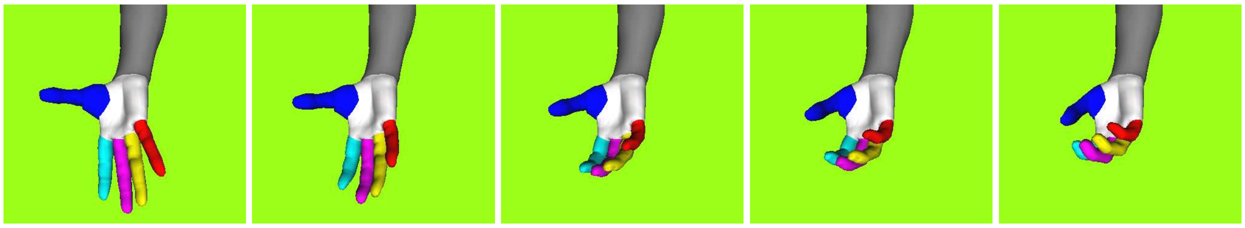
(2) From *lower-right-front* camera.



(3) From *upper-right-front* camera.



(4) From *lower-left-back* camera.



(5) From *lower-right-back* camera.

Figure 5.19: Recording virtual hand's rendered image sequences: at a speed of 3 frames per second in an animation process from 5 cameras under lighting condition. Original image size is 400×350. The virtual hand rendered as a colored object.

Chapter 6

Virtual Hand—a Testbed

With the construction of dynamic hand model and a GUI based on the model, we are now ready to provide a testbed, *Virtual Hand*, which can be used in computer vision experiments on the human hand. The applications include providing hand data and displaying results and API functions.

6.1 Applications

In hand pose estimation and tracking, the first step is to obtain hand images. *Virtual Hand* can provide synthetic hand data for this very first step for two reasons: (1) If the image acquisition system is not successfully installed and tested; (2) *Virtual Hand* produces very accurate hand images through definition of camera parameters and hand configurations. Inside of *Virtual Hand*t he hand configuration can be controlled quantitatively at each hand joint for every different movements. The virtual hand's size, location, and orientation can be determined and adjusted accurately according to the requirements during experiments.

Another benefit of the use of virtual hand for hand input images (data) is that we can create a database of hand gestures, in which every gesture is defined as a set of precise numbers for the angles of hand joint movements (flexion, abduction, and rotation). Based on this database, it would be very easy to produce any gesture images from infinite number of viewing points, because in *Virtual Hand* there is a movable camera whose intrinsic and extrinsic parameters can be given arbitrarily. The OpenGL rendering matrices can also be calculated for every image. Therefore a complete hand data is provided in the format of the image and rendering GL matrix.

Virtual Hand thus provides,to some extent, an "ideal" hand input data which in some case is necessary before hand pose estimation and tracking system can be able to real hand. The testbed has been used in such a case in one project [21, 22].

On the other side, *Virtual Hand* as a testbed can be used to display (output) environment to test experiment's results. After having aligned the real world's coordinate system and the virtual environment's coordinate system (See Figure 5.13), we can test the hand pose estimation algorithm by observing the virtual hand's movement. Also the testbed provides visual environment to test inverse kinematics on hand (See Figure 5.14).

In fact, all the performances introduced in the last chapter can be used to test algorithms on hand analysis and tracking in that the input controlling sequence comes from the hand tracking system, not *Virtual Hand* per se (such as the control dial and slide). *Virtual Hand* provides input and output API functions for this purpose. A brief overview of this API is given in the next section.

6.2 Application Programming Interface

Virtual Hand is designed and implemented following C++ style. Classes *Hand, Finger*, and *XFinger* have been written and the relationship of their substantiated objects is shown in Figure 4.4 in which there is *no inheritance but composition* among the classes (objects): a *Hand* consists of *forearm* (a hand part) and *palm* (a hand part) and *Finger*s and *Thumb*. *Finger* and *Thumb* are composed of hand parts *DIP, PIP, MCP, IP, CMC*. All the hand part objects are substantiated from a hand part class named *XFinger*.

There are a lot member functions in the classes and these member functions provide profound input and output API functionality. For example, through the input APIs, you can control

- whole hand's location and orientation,
- any hand joint's flexion, extension, abduction, adduction, and rotation,
- camera's intrinsic and extrinsic parameters,
- rendering effects (background, styles, light model), and
- recording image.

In the same way, the rendered image, the hand configuration, and OpenGL rendering matrices can be not only displayed in real time as exhibit in last chapter but also but output to other programs through *Virtual Hand*'s output API functions. Due to the length of the code we are referring interested readers to the User Documentation for *Virtual Hand* for more information on how to use the API. currently there are $3,000$ lines of code in ".h" files and $12,000$ lines of code in ".cpp" files.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

This thesis introduced a testbed, *virtual hand*, which can be used in experiments on hand gesture analysis and recognition and hand pose estimation in computer vision domain. It includes a dynamic hand model and a GUI interface for the model.

The hand model is much simpler than those involved in the deformation-related modeling of human body in the graphics research area. It looks realistic and can be used as a reasonable human hand model for computer vision experiments on the human hand. It is better than those hand models used in most of the computer vision projects. The design of hand movements (gestures) is based on the anatomical and biomechanical studies on human hand.

The interface implementation for this model provides an easy-to-use platform in which the hand can be controlled quantitatively in every hand joint in any movement: flexion, extension, abduction, adduction, rotation, and translation of whole hand. Also, the movement process (animation) can recorded not only for the rendered images, but also for the configuration and transformation parameters: joint angles, maximum-minimum movement ranges, box size and location and the hand, transform matrices, and camera parameters in rendering. The interface provides stronger controlling functionality than do those interfaces used in hand-related computer vision projects found in the literature.

The testbed runs in real-time producing gestures and animation sequences. It has already been put into use in one project [21] in the Computer Science Department at the University of Nevada, Reno.

7.2 Future work

As a pioneering testbed for experiments on the human hand, the testbed needs to have more:

- Construction of a database of common hand gestures will greatly enhance the usage of the hand model.
- Animation design based on the gesture database will enrich the testbed applications.
- More work on the thumb base of the hand model will make the model look more realistic.
- Hand constraints' implementation during animation, though a hard work, should guarantee a beautiful hand motion.
- Calibration of the hand model in the testbed with the real hand in physical environment is the first step of applications.
- More work on the interface design: interpretative messages should be automatically output as of the hand configurations and about the user's manipulation on the hand model; instructions on how to use this testbed.

Bibliography

[1] 3D Visual OS [online, cited July, 2003]. Available from World Wide Web: http://atwww.hhi.de:80/~blick/3-D_Visual_OS/3-d_visual_os.html.

[2] Athena Review Image Archive [online, cited July 3, 2003]. Available from World Wide Web: http://www.athenapub.com/sraphand.htm.

[3] The coin source [online, cited November 26, 2002]. Available from World Wide Web: http://www.coin3d.org.

[4] Control Gestures [online, cited July 3, 2003]. Available from World Wide Web: http://www.gvu.gatech.edu/ccg/publications/iswc2000-pendant/img6.png.

[5] Hand – Posteroanterior (PA) View, Labelled [online, cited July 3, 2003]. Available from World Wide Web: http://www.rad.washington.edu/RadAnat/HandPALabelled.html.

[6] Hand Model [online, cited July 3, 2003]. Available from World Wide Web: http://www.handmodel.ca/.

[7] Hand Opposition Example [online, cited July 3, 2003]. Available from World Wide Web: http://www.dpt.state.va.us/comcur/commcurr/2000/mar/hand.gif.

[8] Human Performance Evaluation of Interaction Tecnologies [online, cited July 3, 2003]. Available from World Wide Web: http://hci.stanford.edu/~craig/hci/lectures/zhai/1/sld012.htm.

[9] IBM's Blue Eyes [online, cited July 3, 2003]. Available from World Wide Web: http://www.almaden.ibm.com/cs/blueeyes/.

[10] Inductive learning in hand pose recognition [online, cited July 3, 2003]. Available from World Wide Web: http://www.computer.org/proceedings/fg/7713/77130078abs.htm.

[11] Kdevelop [online]. Available from World Wide Web: http://www.kdevelop.org/.

[12] Qt Programming. Available from World Wide Web: http://www.trolltech.com/.

[13] RSL: Robotic Systems Lab [online, cited July 3, 2003]. Available from World Wide Web: http://www1.volvo.com/group/research_and_technology/maineditnews/1,1901,7_101,00.html.

[14] RSL: Robotic Systems Lab [online, cited July 3, 2003]. Available from World Wide Web: http://www.syseng.anu.edu.au/rsl/.

[15] Johanny Accot and Shumin Zhai. Beyond Fitts' Law: models for trajectory-based HCI tasks. In *ACM Proc. of CHI*, pages 395–302, 1997.

[16] Amaury Aubel and Daniel Thalmann. Realistic deformation of human body shapes [online]. 2000 [cited June 6, 2003]. Available from World Wide Web: http://vrlab.epfl.ch/Publications/publications_index.html.

[17] Norman Badler, Jan Allbeck, Liwei Zhao, and Meeran Byun. Representing and parameterizing agent behaviors. In *in Proc. Computer Animation, IEEE Computer Society*, pages 133–143, Geneva, Switzerland, 2002. Available from World Wide Web: http://www.cis.upenn.edu/~badler/paperlist.html [cited June 26, 2003].

[18] Norman I. Badler and Jan M. Allbeck. Towards behavioral consistency in animated agents, 2001. Available from World Wide Web: http://www.cis.upenn.edu/~badler/paperlist.html [cited June 26, 2003].

[19] Norman I. Badler, Cary B. Phillips, and Bonnie L. Webber. *Simulating Humans: Computer Graphics, Animation, and Control.* Oxform University Press, 1993. Available from World Wide Web: http://www.cis.upenn.edu/~badler/book/book.html [cited June 26, 2003].

[20] G. Bebis, S. Uthiram, and M. Georgiopoulos. Face detection and verification using genetic search. *International Journal on Artificial Intelligence Tools*, 9(2):225–246, 2000.

[21] Geoge Bebis et al. Development of a nationally competitive program in computer vision technologies for effective human-computer interaction in virtual environments [online]. November 2002 [cited July 3, 2003]. Available from World Wide Web: http://www.cs.unr.edu/~aerol/projecthome/publications.htm.

[22] Geoge Bebis et al. Development of a nationally competitive program in computer vision technologies for effective human-computer interaction in virtual environments [online]. June 2003 [cited July 3, 2003]. Available from World Wide Web: http://www.cs.unr.edu/~aerol/report/Report2003.doc.

[23] Charles Bell et al. *The Hand: Mechanism and Vital endowments, as Evincing Design.* Harper & Brothers, New York, 1885.

[24] R. Boulic et al. The HUMANOID environment of multiple deformable human characters, 1995. Available from World Wide Web: http://vrlab.epfl.ch/Publications/publications_index.html [cited June 6, 2003].

[25] Richard Bowden. *Learning Non-linear Models of Shape and Motion*. PhD thesis, deparment of System Engineering, Brunel University, UK, 1999. Available from World Wide Web: `http://www.ee.surrey.ac.uk/Personal/R.Bowden/` [cited June 26, 2003].

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

[27] Lars Bretzner and et al. A prototype system for computer vision based human computer interaction. Available from World Wide Web: `http://www.nada.kth.se/cvap/abstracts/cvap251.html` [cited July 3, 2003].

[28] 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.

[29] Chin-Seng Chua, Haiyin Guan, and Yeong-Khing Ho. Model-based 3D hand posture estimation from a single-2D image. *Image and Vision computing*, 20(3):191–202, March 2002.

[30] Neil A. Davidoff and Andris Freivalds. A graphic model of the human hand using CATIA. *International Journal of Industrial Ergonomics*, 12(4):255–264, 1993.

[31] J. Dowdall, I. Pavlidis, and G. Bebis. Face detection in the near-IR spectrum. *Image and Vision Computing*, 2003.

[32] Gregory Edwards. New software makes eyetracking viable: You can control computers with your eyes. Available from World Wide Web: `http://eyetracking.stanford.edu` [cited July, 2003].

[33] Gregory Edwards. A tool for creating eye-aware applications that adapt to changes in user behavior. Available from World Wide Web: `http://eyetracking.stanford.edu` [cited July, 2003].

[34] Jonathan Gratch et al. Creating interactive virtual humans: some assembly required. Available from World Wide Web: `http://www.cis.upenn.edu/~badler/paperlist.html`.

[35] Open Inventor Architecture Group. *Open Inventor^{TM} C++ Reference Manual: The Official Reference Document for Open Intentor, Release 2*. Addison-Wesley, 1994.

[36] Jennifer Healey, Justin Seger, and Rosalind W. Picard. Quantifying driver stress: Developing a system for collecting and processing bio-metric signals in natural situations, April 1999. Available from World Wide Web: `http://vismod.www.media.mit.edu/cgi-bin/tr_pagemaker` [cited July, 2003]. Proceedings of the Rocky-Mountian Bio-Engineering Symposium.

[37] Tony Heap and David Hogg. 3D deformable hand models [online, cited August,2000]. Available from World Wide Web: `http://www.scs.leeds.ac.uk/vision/proj/ajh/3dmodels.ps`.

[38] Tony Heap and Ferdinando Samaria. Real-time hand tracking and gesture recognition using smart snakes [online, cited August,2000]. Available from World Wide Web: `http://www.scs.leeds.ac.uk/vision/proj/ajh`.

[39] Hewett et al. ACM SIGCHI: Curricula for Human-Computer Interaction [online]. 1996 [cited July, 2003]. Available from World Wide Web: `http://www.acm.org/sigchi/cdg/`.

[40] Hitoshi Hongo and et al. Focus of attention for face and hand gesture recognition using multiple cameras. Available from World Wide Web: `http://www.computer.org/proceedings/fg/0580/0580toc.htm` [cited July 3, 2003].

[41] Thomas Huang et al. Speech and vision integration for display control. Available from World Wide Web: `http://www.ifp.uiuc.edu/IDFL/papers/papers8.html` [cited July, 2003].

[42] Horace H. S. Ip, Sam C. S. Chan, and Maria S. W. Lam. Hand gesture animation from static postures using an anatomy-based model. In *IEEE Proc. of Computer Graphics International*, pages 29–36, Geneva, Switzerland, June 2000.

[43] Horace H S Ip, Maria S W Lam, Ken C K Law, and Sam C S Chan. Animation of hand motion from target posture images using an anatomy-based hierarchical model. *Computer and Graphics*, 25:121–133, 2001.

[44] Qiang Ji and Xiaojie Yang. Real time visual cues extraction for monitoring driver vigilance. International Workshop on Computer Vision Systems, July 7-8, 2001, Vancouver, Canada.

[45] Nebojsa Jojic and et al. Detection and estimation of pointing gestures in dense disparity maps. Available from World Wide Web: `http://www.computer.org/proceedings/fg/0580/0580toc.htm` [cited July 3, 2003].

[46] Mohammed Waleed Kadous. Available from World Wide Web: `http://www.cse.unsw.edu.au/~waleed/gsl-rec/` [cited July,2003].

[47] Prem Kalra et al. Real-time animation of realistic virtual humans [online]. 1998 [cited June 6, 2003]. Available from World Wide Web: `http://vrlab.epfl.ch/Publications/publications_index.html`.

[48] R. Kjeldsen and J. Kender. Toward the use of gesture in traditional user interfaces. Available from World Wide Web: `http://www.computer.org/proceedings/fg/7713/7713toc.htm` [cited July 3, 2003].

[49] James J. Kuch and Thomas S. Huang. Human computer interaction via the human hand: a hand model. In *1994 Conference Record of the Twenty-Eighth Asilomar Conference on Signals, Sysmtems and Computers*, volume 2, pages 1252–1256, October 1994.

[50] James J. Kuch and Thomas S. Huang. Vision-based hand modeling and tracking for virtual teleconferencing and telecollaboration. In *IEEE Proceedings of Fifth International Conference on Computer Vision*, pages 666–671, June 1995.

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

[52] Cheng-Chang Lien and Chung-Lin Huang. The model-based dynamic hand posture identification using genetic algorithms. *Machine Vision and Applications*, 11:107–121, 1999.

[53] 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.

[54] I. Scott MacKenzie. Movement time prediction in human-computer interfaces. http://www.yorku.ca/mack/GI92.html, 1996.

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

[56] David McNeil. *Hand and Mind: What Gestures Reveal About Though*. The University of Chicago Press, Chicago, 1992.

[57] Laurent Moccozet and Nadia Magnenat Thalmann. Multilevel deformation model applied to hand simulation for virtual actors, 1998. Available from World Wide Web: `http://miralabwww.unige.ch/newMIRA/ARTICLES/VSMM97B.pdf` [cited June, 2002].

[58] John Napier. *Hands*. Pantheon Books, New York, 1980.

[59] Luciana Porcher Nedel and Daniel Thalmann. Modeling and deformation of the human body using an anatomically-based approach, 1998. Available from World Wide Web: `http://vrlab.epfl.ch/Publications/publications_index.html` [cited June 6, 2003].

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

[61] Vladimir I. Pavlovic, Rajeev Sharma, and Thomas S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. Available from World Wide Web: `http://citeseer.nj.nec.com/pavlovic97visual.html` [cited July 3, 2003].

[62] Rosalind W. Picard. Affective medicine: Technology with emotional intelligence. Available from World Wide Web: `http://vismod.www.media.mit.edu/cgi-bin/tr_pagemaker` [cited July, 2003].

[63] Rosalind W. Picard, Elias Vyzas, and Jennifer Healey. Toward machine emotional intelligence: Analysis of affective physiological state. Available from World Wide Web: `http://vismod.www.media.mit.edu/cgi-bin/tr_pagemaker` [cited July, 2003].

[64] Claudio S. Pinhanez and Aaron F. Bobick. It/i: A theater play featuring an autonomous computer graphics character, April 1998. Available from World Wide Web: `http://vismod.www.media.mit.edu/cgi-bin/tr_pagemaker` [cited July, 2003]. Proceedings of CHI '98, Los Angeles, CA.

[65] Hans Rijkema and Michael Girard. Computer animation of knowledge-based human grasping. *Computer Graphics*, 25(4):339–348, July 1991.

[66] Jocelyn Riseberg and et al. Frustrating the user on purpose: Using biosignals in a pilot study to detect the user's emotional state. Available from World Wide Web: `http://vismod.www.media.mit.edu/cgi-bin/tr_pagemaker` [cited July, 2003]. CHI98.

[67] Ramon Mas Sanso and Daniel Thalmann. A hand control and automatic grasping system for synthetic actors, 1994. Available from World Wide Web: `http://vrlab.epfl.ch/Publications/publications_index.html` [cited June 6, 2003].

[68] Yoichi Sato and et al. Fast tracking of hands and fingertips in infrared images for augmented desk interface. Available from World Wide Web: `http://www.computer.org/proceedings/fg/0580/0580toc.htm` [cited July 3, 2003].

[69] Andrew Sears. HCI Education: Where is it Headed? [online]. 1997 [cited July, 2003]. Available from World Wide Web: `http://www.acm.org/sigchi/bulletin/1997.1/education.html`.

[70] Rajeev Sharma, Vladimir I. Pavlovic, and Thomas S. Huang. Toward multimodal human-computer interface. Available from World Wide Web: `http://www.ifp.uiuc.edu/IDFL/papers/paperssharma.html#sharmaab18` [cited July, 2003].

[71] Jamie Sherrah and Shaogang Gong. Vigour: A system for tracking and recognition of multiple people and their activities. Available from World Wide Web: `http://www.computer.org/proceedings/icpr/0750/Volume%201/07501179abs.htm` [cited July, 2003]. ICPR'00 -Volume 1.

[72] Thad Starner, Joshua Weave, and Alex Pentland. Real-time american sign language recognition using desk and wearable computer based video. Available from World Wide Web: `http://vismod.www.media.mit.edu/cgi-bin/tr_pagemaker` [cited July 3, 2003]. PAMI; Submitted 4/26/96.

[73] David Joel Sturman. *Whole-hand Input*. PhD thesis, Massachusetts Institute of Technology, Febrary 1992. Available from World Wide Web: `http://xenia.media.mit.edu/~djs/thesis.ftp.html` [cited July, 2003].

[74] Z. Sun, G. Bebis, and R. Miller. On-road vehicle detection using Gabor fiters and support vector machines. In *IEEE International Conference on Digital Signal Processing*, Santorini, Greece, July 2002.

[75] Z. Sun, R. Miller, G. Bebis, and D. DiMeo. A real-time precrash vehicle detection system. In *IEEE Workshop on Applications of Computer Vision*, pages 171–176, Orlando, FL, USA, 2002.

[76] Daniel Thalmann and Jean-Sébastien Monzani. behavioural animation of virtual humans: what kind of laws and rules?, 2002. Available from World Wide Web: `http://vrlab.epfl.ch/Publications/publications_index.html` [cited June 6, 2003].

[77] Josie Wernecke and Open Inventor Architecture Group. *The Inventor Toolmaker: Extending Open Intentor^{TM}, Release 2.* Addison-Wesley, 1995.

[78] Josie Wernecke and Open Inventor Architecture Group. *The Inventor Mentor: Programming Object-Oriented Graphics with Open Intentor^{TM}, Release 2.* Addison-Wesley, 2000.

[79] Andrew Wilson. Luxomatic: Computer vision for puppeteering. Available from World Wide Web: `http://vismod.www.media.mit.edu/cgi-bin/tr_pagemaker` [cited July 3, 2003].

[80] Christopher R. Wren et al. Combining audio and video in perceptive spaces, December 1999. Available from World Wide Web: `http://vismod.www.media.mit.edu/cgi-bin/tr_pagemaker` [cited July, 2003]. 1st International Workshop on Managing Interactions in Smart Environments.

[81] Ying Wu and Thomas S. Huang. Hand modeling, analysis, and recognition for vision-based human computer interaction. *IEEE Signal Processing Magazine*, 18(3):51–60, May 2001.

[82] Ying Wu, John Y. Lin, and Thomas S. Huang. Capturing natural hand articulation. In *in Proc. of IEEE Int'l Conf. on Computer Vision (ICCV'01)*, volume 2, pages 426–432, Vancouver, Canada, July 2001.

[83] Yoshihiro Yasumuro, Qian Chen, and Kunihiro Chihara. Three-dimensional modeling of the human hand with motion constraints. *Image and Vision Computing*, 17:149–156, 1999.