Hand-Eye: A Vision-Based Approach to Data Glove Calibration

Te-Shun Chou Ashley Gadd
Department. of Electrical and Computer Engineering
University of British Columbia
2356 Main Mall
Vancouver, BC
Canada V6T 1Z4
+1 604 822 2872
te-shunc/gadd@ece.ubc.ca

Dave Knott
Imager Computer Graphics Lab
Department of Computer Science
University of British Columbia
2366 Main Mall
Vancouver, BC
Canada V6T 1Z4
+1 604 822 8122
knott@cs.ubc.ca

ABSTRACT

We describe a new method for data glove calibration that uses computer vision techniques to create a filter for individual customization of input obtained from the glove. Our major observation is that it is possible to create a linear correlation between the hand posture reported by the data glove and the observed posture of the hand itself.

We use a feature-based computer vision system independently to extract information about hand posture from video images of a human hand that is using the data glove. We simultaneously collect the glove data for the same posture. Linear regression is used on the combined sets of reported data to establish a filter that customizes the data reported from the glove for an individual user. The filtered glove data is mapped onto a computer-generated image of the hand's skeletal structure. We show by comparison that the computer-generated hand image exhibits a posture quite similar to that of the actual hand.

Keywords
data glove, calibration, computer vision, computer graphics, linear regression, hand model

1 INTRODUCTION

Of the existing human computer interaction (HCI) techniques, keyboards and mice are probably the best-known input devices. However, these devices constrain the dexterity and naturalness of interaction with computer-controlled applications. This limitation becomes more apparent when we employ these devices settings such as virtual reality applications that require a wide variety of input from the user. Thus, in recent years there has been a tremendous push in research toward novel devices and techniques for natural and friendly interaction with computers. In particular, the analysis of hand motion has proved attractive to many computer animation and virtual reality researchers since the human hands perform most everyday tasks, assist in communication with other individuals and can sometimes express our feelings.

Human hands are a very complex and delicate mechanical structure with approximately 30 degrees of freedom, which vary amongst individuals. Consequently, successfully interpreting an individual's hand motion becomes a necessary and significant task.

In this paper, a new approach for hand model calibration is reported. We base our observations on a data glove device called the CyberGlove [2,6]. It is constructed with stretch fabric and eighteen resistive bend sensors that describe the position of the finger joint angles (see Figure 1). The idea is to find a relationship between raw sensor data reported by the CyberGlove and the actual joint angles of the hand obtained using computer vision technique. The relationship is established through the use of a statistical linear regression analysis method. This may seem like a straightforward approach but the implementation is in fact relatively complicated. First, the accuracy of the joint angles obtained from a glove-based input device is usually dubious. This problem stems primarily from the restriction that a data glove is not tailored the specific geometry and features of an individual user's hand. Perforce, the glove manufacturer assumes that a user's hand conforms to a generic model. This problem is compounded by the fact that data from such gloves is usually taken directly from the glove's sensors without filtering or altering it to match a specific user. Secondly, the actual human hand configuration is quite different from model assumed by the data glove. Each joint of the human finger has its own relative degree of freedom that varies from one person to another. On the other hand (so to speak), the CyberGlove has only a single degree of freedom flexion in each joint. Thirdly, the movements of some fingers are not completely independent from those of others. For example, the middle finger will comply with the plane of palm when the index finger is abducted from it. Fourth, one finger's joint movement may affect several CyberGlove sensors instead of one. For example, when we rotate the thumb toward palm, the values of both the thumb rotation sensor and the thumb abduction sensor are changed. Therefore, understanding how to map the CyberGlove sensor data to the hand joint angles is a critical task.

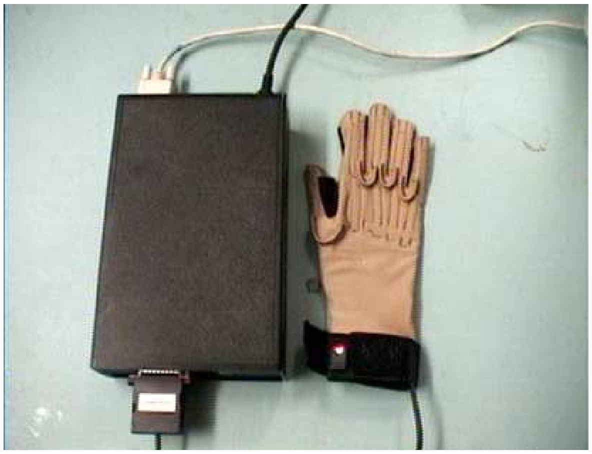


Figure 1 - The Virtual Technologies CyberGlove

To solve all these problems described above, we divide our calibration system into two steps. In the first step, a monocular camera system is used to take a series of images of the CyberGlove while the user wears it. To assist in the identification of features in the image, a coloured marker is placed on each visible joint of the CyberGlove (see Figure 5). These images are then processed using edge detection to extract joint angles related to the hand model. Once the features are extracted from vision system, the second step consists determining the mapping from CyberGlove sensors to hand joint angles through the use of linear regression analysis. Finally, once an appropriate sensor mapping has been determined, a virtual hand is created which employs the results of the calibration in the display of a graphical model of the user's hand based on real-time data obtained from the glove.

Hand model building in this paper is to combine both techniques of computer vision system and computer graphics. The objectives are to recognize the CyberGlove reading from the computer vision system extracted features and to create a visual displaying hand providing an immediate feedback. Also, the model is easy to calibrate to any person's hand since we do not hope it constricts anyone from using this hand model system. These issues will be described in the following sections.

2 DESCRIPTION OF THE HUMAN HAND MODEL SKELETON

Finding a relationship between the data reported by the glove's sensors and the actual hand posture of the user is dependent on using a reasonable description of the skeletal structure of the hand. The human hand is actually a very complicated piece of machinery for which it is difficult to gain a physically exact model. Hence, many researchers have used simpler approximations to the hand's physical structure [4,5,6,9]. We have relatively little to add to this subject, except to note that we have augmented features described by these models to match our particular problem domain.

The skeleton of human hand is modeled with a hierarchical structure of links connected by joints with one or two rotational degrees of freedom. Figure 2 illustrates our model of the human hand skeleton.

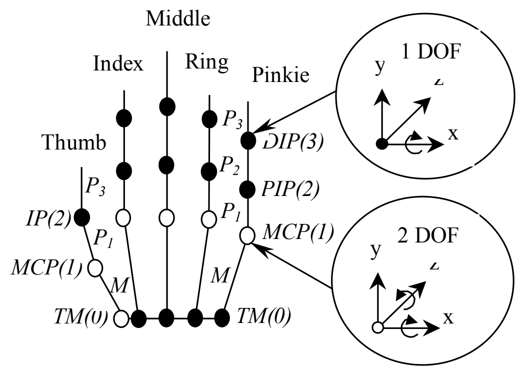


Figure 2 - Hand skeleton notational diagram

The abbreviations used to describe the fingers, links, and joints in this diagram are summarized in Table 1.

Fingers		Joints		Links	
T	Thumb	TM	Trapezio Metacarpal	P_1	Proximal Phalanx
I	Index	MCP	Metacarpo Phalangeal	P_2	Middle Phalanx
M	Middle	IP	Interphalangeal	P_3	Distal Phalanx
R	Ring	PIP	Proximal Interphalangeal	M	Metacarpal
P	Pinky	DIP	Distal Interphalangeal		

Table 1 – Finger, joint and link abbreviations

3 NOTATIONS

We have developed a notation for the representation of the joint angles in the hand model and the sensor data from the CyberGlove.

We begin with the notation used in the hand model. In order to describe a rotational position around a specific joint, we label five fingers thumb as *T*, index as *I*, middle as *M*, ring as *R*, and pinkie as *P*. Each joint of the fingers is also labeled as shown in Figure 2.

The rotational position of each joint of a finger is represented by θC_j^k, where *k* represents the number of the joint on the finger *C* and *i* represents the axis of rotation, either *X* or *Z*. For example, θT_X^1 refers to a rotation around the X-axis of the thumb's *MCP* joint. In total, there are twenty-three joint angles taken into consideration.

Next, we consider the sensor data of CyberGlove. There are 18 bend sensors on the CyberGlove that monitor the motions of the human hand [1,7]. The sensors are located over or near the joints of hand and wrist. We use X_i, where $0 \le i \le 17$, to denote these eighteen sensors, each of which has 8-bit resolution. For the thumb, there are three sensors X_0, X_1, *and* $X_{2,}$ which measure the *TM*, *MCP*, and *IP* joints respectively. For the remaining fingers, index, middle, ring, and pinkie, there are two sensors for each finger to measure the *MCP* and *PIP* joints, which are X_4, X_5, X_6, X_7, X_9, X_{10}, X_{12}, and X_{13}. The CyberGlove also has four sensors to measure the abduction angles, which are thumb-index abduction X_3, middle-index abduction X_8, ring-middle abduction X_{11}, and pinkie-ring abduction $X_{14.}$ Finally, X_{15} measures the palm arch angle, and X_{16} and X_{17} measure the wrist pitch and yaw angles respectively.

4 CONSTRAINTS ON THE HAND MODEL

Normally, it is impossible to move the *DIP* without moving the adjacent *PIP* joint and vice versa in human finger joint movements. According to the previous study result [4,5,6], it revealed that a linear relationship exists between these two joint angles. θ_{DIP} can be expressed as:

$$\theta_{DIP} = \frac{2}{3}\theta_{PIP} \qquad (1)$$

As described earlier, the eighteen-sensor CyberGlove has no sensors on the *DIP* joints. We use the above constraint in our hand model calibration to infer the *DIP* joint angle from the *PIP* joint angle. In addition, we do not consider the wrist pitch and yaw angles in order to simplify our hand model calibration and display, since they do not contribute to the overall posture of the hand itself.

After defining the joint notation, CyberGlove sensor data, and hand model constraints, the following section demonstrates how we derive the relationship between the movement of each joint and CyberGlove sensor data.

5 DATA MAPPING

Theoretically, the digitized sensor values of the CyberGlove vary linearly with values ranging from 0 to 255. We can mathematically describe the relationship between the actual joint angle and the raw CyberGlove sensor readings as a function *f*:

$$\theta = f(x) \qquad (2)$$

where θ is the actual joint angle and *x* is a set of the values reported by individual CyberGlove sensors. As mentioned earlier, it is important to recognize that one joint angle may depend on the values of multiple glove sensors.

In order to find the relationship between CyberGlove sensor readings and hand joint movement, we observed the CyberGlove output in a variety of hand postures while wearing the glove. By noting which sensors exhibited a change when a given joint angle changed, we could determine the sensor dependencies for each joint. These relationships are summarized in Table 2.

Joint	Relation	Joint	Relation	Joint	Relation
θI_x^1	X_4	θT_x^1	X_1	θI_x^0	X_{15}
θI_x^2	X_5	θT_x^2	X_2	θR_x^0	X_{15}
θM_x^1	X_6	θT_x^0	X_0, X_3	θP_x^0	X_{15}
θM_x^2	X_7	θT_z^0	X_0, X_3		
θR_x^1	X_9	θI_z^1	X_8	θI_x^3	2/3 θI_x^2
θR_x^2	X_{10}	θM_z^1	X_8, X_{11}	θM_x^3	2/3 θM_x^2
θP_x^1	X_{12}	θR_z^1	X_{11}, X_{14}	θR_x^3	2/3 θR_x^2
θP_x^2	X_{13}	θP_z^1	X_{14}	θP_x^3	2/3 θP_x^2

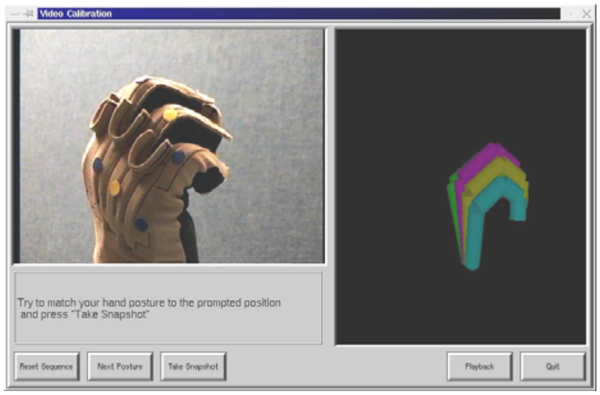
Table 2 - Joint/Sensor relationships

Note that some joint angles are related to only one CyberGlove sensor while some are related to two. See Section 7 for a discussion of the hand model calibration, one-on-one mapping, and one-on-two mapping.

6 DESCRIPTION OF VISION SYSTEM

In order to measure the actual angles of the joints and the lengths between the joints, we marked each joint with a small coloured sticker. We then used a monocular camera system to record a series of images of the hand as the user made different movements with the CyberGlove. During the measurement, the user was required to form a series of prompted hand postures, as shown in Figure 3. The graphical prompt depicts the hand position that the user should be forming, while the camera view shows what the camera will record.

Once the user is satisfied that the camera view matches the on-screen prompt as closely as possible, he presses a button on the keyboard. Both the camera image and the corresponding CyberGlove sensor values are simultaneously recorded. This procedure is repeated for several predetermined hand postures specifically designed for the calibration procedure. Colour filtering and edge detection techniques were then used to extract a set of features from each image, and the features were analyzed for desired geometric relationships such as joint angles.



Try to match your hand posture to the prompted position
and press "Take Snapshot"

Figure 3 - Interface for vision data acquisition

6.1 Markers

Coloured markers are used to help identify the features in the image [2,3]. The image is processed using a colour mask, with all colours but those of the marker filtered out. The position of the marker on the image then determines the location of the relevant feature. By putting different colours on different features, the locations of specific features can be discerned. For example, different colours on each of the fingers will partially mitigate occlusion by other fingers - even a partially occluded finger can be identified as long as the marker can be seen.

We investigated several marking techniques before deciding on the use of coloured dots. These included dots, rubber bands, and lines. Dots placed over the joints would allow identification of the main segments of the fingers, but would require repositioning for different users. Rubber bands would be placed around both the proximal and distal ends of each bone in the fingers, allowing the joints to be identified by their relationship to the bones. This method requires a certain amount of extrapolation because the bands cannot be placed around the joints themselves as they would be likely to slide off to one side or the other. Bands would also require repositioning for different users. Lines on the glove, up the back of each finger, would present a continuous feature that could also be used for data extraction, but would be prone to occlusion if the hand were turned away, unlike the bands.

Both the rubber bands and the lines would work well for the fingers. However, to extract information for the rest of the hand is a more difficult matter. Rubber bands don't fit well around the palm. The best way to use markers to assist in feature extraction from the palm is to use lines crossing the back of the hand; such lines could be easily detected and examined for curvature, which could then be related to the parameters of the palm.

For the purposes of our prototype, however, dots proved to be the most versatile. Because they are simply stickers, they can be moved around to experiment with different marking positions and combinations, and to avoid occlusion. We leave open the option of experimenting with bands or lines in future work.

6.2 Colour Filtering

The main part of the feature extraction process involved filtering the colours of the markers to determine their positions. Since the illumination is fairly consistent, a simple thresholding algorithm was used to determine where the markers are. This information was then translated into (x,y) coordinates for use in the feature extraction routine.

Four colours of marker were employed: red, green, blue, and yellow. Each colour was unique in the image, in that no objects other than the markers had these colours. Each colour is defined by its red, green, and blue (RGB) components; for example, the yellow marker is mainly red and green, with little or no blue, while green has almost no red or blue. To determine where the markers are, the image was divided into its three colour planes and scanned for peaks. If, for example, a peak in the green colour plane coincided with lows in the red and blue colour planes, the colour at that point was interpreted as green. By finding the points with the strongest characteristics for each colour of marker, the positions of the markers were determined.

The filtering process was done in four steps to find each of the four colours of markers, with the results stored in four separate bitmaps – on each bitmap, a pixel was set if the corresponding pixel of the original image was of the respective colour, and cleared otherwise. The result was four two-tone images, each of which had pixels set where there were markers of the related colour. The centroid of each area of pixels was then determined; this was defined as the position of the marker.

Once the position of a marker was known, it was recorded as an (x,y) coordinate pair, with the corresponding colour also noted. These data were then be used to extract desirable features from the image. There was another pre-processing step, though, that added information to the marker positions to aid the extraction of features. The next section discusses that step, edge-detection.

6.3 Edge Detection

The edge detection algorithm uses a centre-surround function to filter the image. The result is analyzed for zero crossings, and then passed through a threshold filter to remove noise. The resulting image is a map of the edges found in the original image, with the different parameters for the centre-surround function providing different edges in the result. Most of the functions needed to process the image are provided by the Vista image processing suite by Art Pope [8].

The centre-surround function used is the Laplacian of Gaussian function:

$$\nabla^2 \cdot G = (1 - \frac{x^2 + y^2}{2 \cdot \sigma^2}) \cdot e^{-\frac{x\ +y^2}{2\cdot\sigma^2}} \qquad (3)$$

It is radially symmetric and looks like the function shown in Figure 4 when sliced through its centre-point. It is weighted so that the positive part of the graph integrates to +1, while the negative portion integrates to –1, and it is used as a convolution mask. It can be of varying size, usually ranging from 3 to 15 pixels square, with a σ value of about 1/7 of that value. The effects of these values will be discussed below. The ∇^2G function is convolved with the image, producing a gradient map of the image.

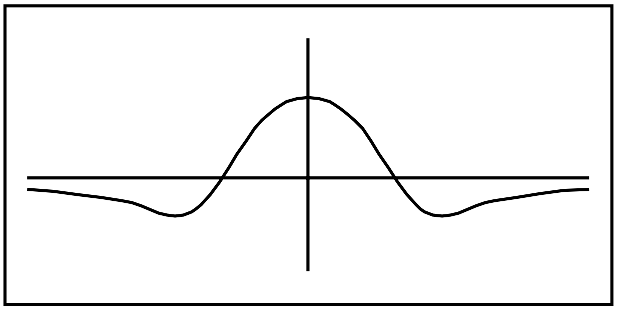


Figure 4 - ∇^2G function cross-section

The image map is passed to a zero crossing detection procedure, which joins up zero crossings into likely edges. However, because image noise produces many zero crossings in a flat area, the image must also be filtered. A thresholding algorithm uses a double threshold to determine which zero crossings to retain – the gradient must rise above a threshold to be retained, but once the gradient is accepted all connected zero crossings must fall below a lower threshold before being rejected. The operation of the thresholder is conceptually similar to a heterodyne, in which different thresholds are used depending on whether the signal is increasing or decreasing.

The result of the image processing steps above is an edge map of the image. The edges detected depend on the size of the convolution mask, with larger sizes averaging the gradient over wider features in the picture. The sigma value also affects what scale edges are detected, and the two threshold values trade off between noise insensitivity and feature insensitivity.

The resulting image, consisting of all the selected edges of the original image, was then analyzed for feature extraction. As a first step, the edges were approximated as line segments. Then, interesting line segments were examined. Two line segments converging on a point indicate a joint; two parallel line segments probably indicate a single segment, such as a bone in a finger. These interesting features, in tandem with the marker data, were used to identify features such as the finger joints and palm structure of the hand. These features are then easily measured to extract the relevant data for the regression analysis.

6.4 Feature Extraction

There are two kinds of features that we wanted to extract from our image data. These were joint angles and relative link lengths. The first required that, for each joint, the joint and its two neighbouring links were determined. The second required that the two joints surrounding a single link were found.

By marking each joint with a dot, and making each finger's markers a different colour, the salient features were extracted. For example, in Figure 5, the back view of the hand, we marked the index finger with green, the middle with blue, the ring with yellow, and the pinkie with blue. Note that there is also a single blue marker on the back of the palm; this marker provides an easy reference for all the others. The analysis consisted of filtering the colours to retrieve a single point where each marker is located. Then the desired features were determined from the locations of those points, along with the known original colours of the points.

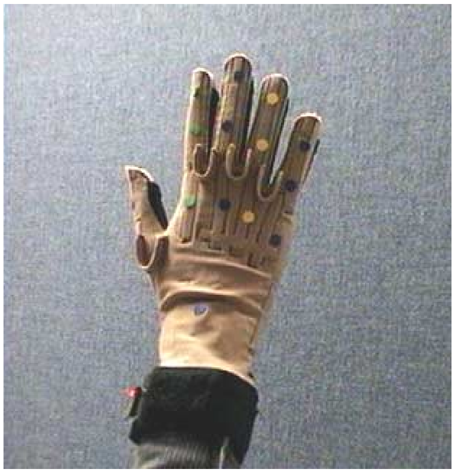


Figure 5 - Data glove with markers

In the example, the abduction angle between the index and middle fingers can be determined by extrapolating straight lines from the three green Index markers and the three blue Middle markers. The extra green marker is used to determine in which direction the extrapolation must occur in case there is any doubt. The angle between the two lines is the abduction angle between the two fingers. Similarly, by placing the markers on the edges of the fingers and viewing the hand from the side, joint angles such as θI_x^{1}, the metacarpophalangeal joint of the index finger, can be found.

One problem with this approach is that the fingertip is hard to mark – since the marker should be centred about the middle of the joint, it must be hanging off the end of the finger. One way of getting around this difficulty is to use edge detection to find the end of the finger. Once the edges have been found, as described in Section 6.3, the end can be found by several methods. One method is to examine only the edge set, looking for two parallel segments attached to two or more converging segments. Another method uses the known joint positions as starting points and extrapolates distally until the end edge of the finger is intersected. The latter method is easier to implement but works easily only for straight-fingered postures.

The second type of feature we wish to extract from the image is the relationship between the various link lengths. Since we are using only a monocular camera system, absolute distances cannot be determined without a known reference point. However, the relative lengths between links can be determined, and one link measured externally to the system. The relative lengths are easily found by measuring the distances between the markers on each finger, with the fingertip known from edge detection.

Two types of features were extracted from the images of the hand: joint angles and relative link lengths. These features were extracted from a combination of coloured marker identification and edge detection. Once the data is extracted, it can be correlated to the CyberGlove data as demonstrated in the following section.

7 HAND MODEL CALIBRATION

When the sequence of images captured from computer vision system are analyzed, a set of data for each finger's joints is obtained. Each of these corresponds to a set of CyberGlove data, recorded at the same times. These two sets are:

$$X = (x_1, x_2, ..., x_n) \text{ and } \tag{4}$$

$$\phi = (\theta_1, \theta_2, ..., \theta_n), \tag{5}$$

where X is the CyberGlove sensor data and ϕ is the image joint angles data.

To determine the function f in equation 1, calibration of the human hand model is divided into two correlation parts, one-to-one mapping and one-to-two mapping, as described in section 5.

7.1 One-to-One Mapping

In this section, we consider those joint angles that have a one-to-one relationship with a sensor. A simple linear regression method employing least squares analysis is used. The general simple linear regression equation can be written as:

$$Joint\_Angle = Gain \cdot (\alpha + \beta \cdot Sensor\_Value) \tag{6}$$

where *Joint_Angle* is the angle in radians, *Sensor_Value* is a value from zero to 255, α and β are the regression coefficients, and the *Gain* is a manually adjustable parameter to ensure that the displayed graphic joint angle is consistent with the physical joint angle.

In the calculation of the two regression coefficients, α and β, S*ensor_Value* and *Joint_Angle* are denoted as X and Y respectively. Then the equations for α and β are

$$\beta = \frac{\sum_{i=1}^{n} xy}{\sum_{i=1}^{n} x^2} \text{ and } \tag{7}$$

$$\alpha = \overline{Y} - \beta\overline{X}, \tag{8}$$

where $x = X - \overline{X}$ and $y = Y - \overline{Y}$, with $\overline{X}$ as the average value of all X and $\overline{Y}$ the average value of all Y.

After calculating α and β, each of one-to-one mapping joint angles is obtained. Then the four *DIP* angles can be derived from equation 1. For example, a possible posture of the index finger is shown in Figure 6. Three joint angles, *MCP*, *PIP*, and *PIP*, are represented in equations 9 to 11 respectively.

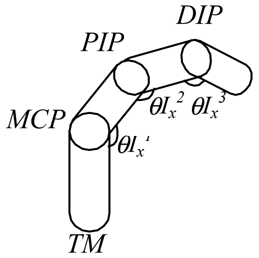


Figure 4 - Gesture of index finger

$$\theta I_x^{1} = Gain(\alpha_1 + \beta_1 X_4) \tag{9}$$

$$\theta I_x^{2} = Gain(\alpha_2 + \beta_2 X_5) \tag{10}$$

$$\theta I_x^{3} = \frac{2}{3}\theta I_x^{2} \tag{11}$$

7.2 One-to-Two Mapping

In this section, we consider those joint angles that have a one-to-two relationship with the sensors. A multiple linear regression analysis method employing least squares analysis is used. The general multiple linear regression equation is defined as:

$$Joint\_Angle = Gain \cdot (\alpha + \beta \cdot Sensor\_Value1 + \gamma \cdot Sensor\_Value2), \tag{12}$$

where *Joint_Angle* is in radians, *Sensor_Value1* and *Sensor_Value2* are values from 0 to 255, and α, β, and γ are the regression coefficients.

In the calculation of α, β, and γ, S*ensor_Value1* and S*ensor_Value2* of the CyberGlove are denoted as X and Z, and *Joint_ Angle* of the vision-system is denoted as Y.

α, β, and γ are then calculated as follows:

$$\beta = \frac{\sum_{i=1}^{n} z^2 \sum_{i=1}^{n} xy - \sum_{i=1}^{n} xz \sum_{i=1}^{n} zy}{\sum_{i=1}^{n} x^2 \sum_{i=1}^{n} z^2 - \left(\sum_{i=1}^{n} xz \right)^2} \qquad (13)$$

$$\gamma = \frac{\sum_{i=1}^{n} x^2 \sum_{i=1}^{n} zy - \sum_{i=1}^{n} zx \sum_{i=1}^{n} xy}{\sum_{i=1}^{n} x^2 \sum_{i=1}^{n} z^2 - \left(\sum_{i=1}^{n} xz \right)^2} \qquad (14)$$

$$\alpha = \overline{Y} - \beta\overline{X} - \gamma\overline{Z} \qquad (15)$$

where $\overline{X}$, $\overline{Y}$, and $\overline{Z}$ are averages and $x = X - \overline{X}$, $y = Y - \overline{Y}$, and $z = Z - \overline{Z}$.

The one-to-two mapping joint angles $\theta T_x^{\,0}$, $\theta T_z^{\,0}$, $\theta M_z^{\,1}$, and $\theta R_z^{\,1}$ are obtained from equations 12 through 15.

8 GRAPHIC DISPLAY OF THE HUMAN HAND

After performing the linear regression analysis from the previous section, we have developed a filter that allows us to tailor the data glove input for an individual user. At this point we are ready to construct a graphical depiction of the hand. When the hand is displayed, a three-dimensional geometric model of the hand's skeleton is presented to the user. This skeleton is represented as a set of linked cylinders. The justification for our purposefully simplistic display of the hand is that we are interested primarily in the analysis of the hand's posture and are therefore not as interested in an exact model of the hand's skin and muscular structure. In fact, these features could potentially obscure vital information regarding the posture of the hand's skeleton.

The hand model is animated in real time based on sensor data retrieved from the CyberGlove. The sensors are polled for data at regular intervals and this data is passed through the regression map filter to update the appropriate joints of the hand model.

As mentioned previously, even after establishing the regression map it may be necessary to fine-tune the calibration. This is done by manually adjusting the gain on the filtered data being passed to the joints. A convenient interface is provided to perform manual calibration (see Figure 7). The manual calibration can be done while the hand model is being animated. This permits easy inspection of the results in an interactive setting. We also accompany the animation of the hand model with a simultaneous video display of the hand to aid in comparison of the posture of the animated model to that of the hand itself.

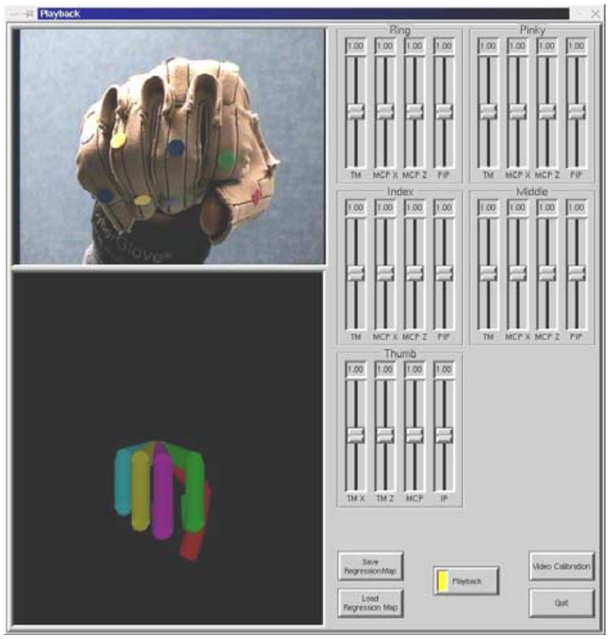


Figure 5 - Playback and manual calibration

9 CONCLUSIONS AND FUTURE WORK

We have presented a novel approach to data glove calibration. By using computer vision techniques and linear regression, along with a representative hand model, we attempted to provide an automated calibration procedure for the Virtual Technologies CyberGlove.

At the time of this writing, we had not yet conducted structured trials to statistically determine the validity of our approach to CyberGlove calibration. However, we drew several conclusions from our experiences with the project.

First, the method by which we obtained data and interacted with the user proved to be very intuitive and straightforward. In particular, gathering the images and glove data for the regression analysis felt very natural. The graphical display of prompted poses with simultaneous video capture was especially effective. In addition, the manual gain calibration of the active data glove seemed likely to be a helpful tool. Also, the regression analysis leant itself to a simple, yet highly usable, implementation.

The edge detection proved to be effective within the limited domain that we were able to use it. However, the vision system suffered from a general lack of accuracy brought on by several factors. These included the volatile nature of hand marker location and difficulty in obtaining three-dimensional data from two-dimensional images.

We have identified several directions through which this work could be expanded in future research. In reality, the joint-sensor mappings are much more complex than the one-to-one and one-to-two mappings to which we restricted ourselves. A more realistic approach would be to implement many-to-many mappings. In addition, it is not completely clear that the mappings are linear, and an exploration of higher-order mapping spaces would be appropriate. As mentioned previously, the largest drawback was the quality of data obtained from the vision system. Better results would likely be obtained by using more sophisticated methods such as three-dimensional model matching. Finally, it would have been aesthetically pleasing to have a more complex and physically accurate display of the surface of the hand available during playback.

REFERENCES

1. "CyberGlove Reference Manual", *Virtual Technologies Inc.*, 1998.
2. M. Fischer, P. Smagt, and G. Hirzinger, "Learning Techniques in a Dataglove Based Telemanipulation System for the DLR Hand", *IEEE Int. Conf. on Robotics & Automation*, pp.1603~1608, 1998.
3. T. S. Huang and V. I. Pavlovic, "Hand Gesture Modeling, Analysis, and Synthesis", *International Workshop on Automatic Face- and Gesture-Recognition, Zurich*, pp.73~79, 1995.
4. J. J. Kuch and T. S. Huang, "Human Computer Interaction via the Human Hand: A Hand Model," *1994 Conf. Record of the Twenty-Eighty Asilomar Conf. on Signal, Systems, and Computers*, pp.1252~1256, 1994.
5. J. Lee and T. L. Kunii, "Model-Based Analysis of Hand Posture", *IEEE Computer Graphics and Applications*, pp.77~86, 1995.
6. V. I. Pavlovic, R. Sharma, and T. S. Huang, "Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review", *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.19, No.7*, pp.677~695, July 1997.
7. "VirtualHand Software Library Reference Manual", *Virtual Technologies Inc.*, 1998.
8. "Vista Computer Vision Programming Library Manual". On-line at: <http://www.cs.ubc.ca/nest/lci/vista/vista.html>
9. Y. Yasumuro, Q. Chen, and K. Chihara, "3D Modeling of Human Hand with Motion Constraints," *IEEE Int. Conf. on Recent Advances in 3-D Digital Imaging and Modeling*, pp.275~282, 1997.