Towards Perceptual Interface for Visualization Navigation of Large Data Sets

Min C. Shin
Dept of Computer Science
UNC at Charlotte
Charlotte, NC 28223
mcshin@uncc.edu

Leonid V. Tsap
Electronics Engineering Dept
Lawrence Livermore National Lab
Livermore, CA 94551
tsap@llnl.gov

Dmitry B. Goldgof
Dept of CS&E
Univ of South Florida
Tampa, FL 33620
goldgof@csee.usf.edu

Abstract

[1] *This paper presents a perceptual interface for visualization navigation using gesture recognition. Scientists are interested in developing interactive settings for exploring large data sets in an intuitive environment. The input consists of registered 3-D data. Bezier curves are used for trajectory analysis and classification of gestures. The method is robust and reliable: correct hand identification rate is 99.9% (from 1641 frames), modes of hand movements are correct 95.6% of the time, recognition rate (given the right mode) is 97.9%. An application to gesture-controlled visualization is also presented. The paper advances the state-of-the-art of human-computer interaction with a robust attachment- and marker-free gestural information processing for visualization.*

1. Introduction

1.1. Motivation

Large and complex data sets are produced at a more rapid pace than tools and algorithms for their processing, analysis, and exploration. For example, nearly a terabyte of satellite data is produced daily. Advanced physics simulation at Lawrence Livermore National Laboratory (LLNL) is generating large data sets, which are expected to increase to one terabyte every five minutes by 2004.

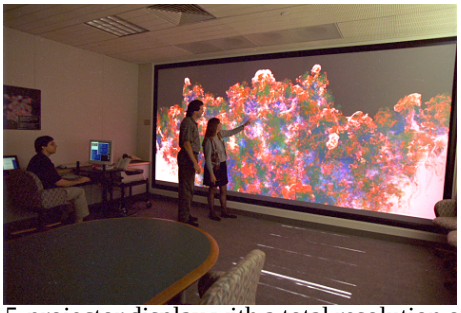
Among the tools used to help explore and understand large data, visualization aids in gaining greater insight into important physical parameters (such as temperature, height, stress, velocity or pressure) and in finding anomalies. State-of-the-art visualization displays keep pace with the data requirements. For instance, a "power wall" (Figure 1(a)) is a 15-projector system that displays approximately 19.7 million pixels on a 16- by 8-ft screen. Systems like this allow for detailed data analysis and team collaboration. However, applying even simple commands to the data (such as zoom, rotation and translation) requires a secondary, or "background," communication process between scientists working with the data (see the two standing by the screen in Figure 1(a)) and the "operator" responsible for executing selected commands (sitting, left). This reduces productivity of the team and affects quality of presentations.

Therefore, scientists are interested in developing new, interactive settings for exploring their data in a more intuitive environment. A gesture-recognition system can interpret commands and supply data manipulation parameters to visualization software (Figure 1(b)) without having an "operator" involved in the process.

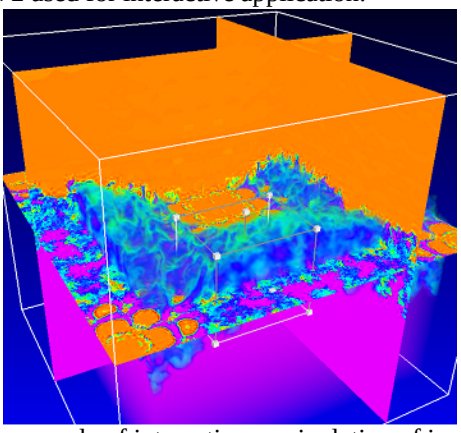
1.2. Previous Work

A significant amount of work is being performed in the area of recognition, where Hidden Markov models (HMMs) are often employed successfully [6, 12] by allowing researchers to address the highly stochastic nature of human gestures. Yacoob and Black proposed parameterized representation of human movement in the form of principal components [13]. Bobick and Wilson [2] treated gesture as a sequence of states and computed configuration states along prototype gestures. Yang and Ahuja [14] used motion trajectories for recognition. Hong *et al.* [3] treated each gesture as a finite state machine (FSM) in the spatial-temporal space; FSMs were trained using k-means clustering. Hongo *et al.* [4] performed recognition by a linear discriminant analysis in each discriminant space by using four directional features. The only published method based on parametric curves was reported by Lee and Xu [5]. A review by Aggarwal and Cai [1] classified approaches to human motion analysis, the tasks involved, and major areas related to human motion interpretation. A review by Pavlovic *et al.* [7] addressed main components and directions in gesture recognition research for human-computer interaction (HCI). This research extends previous work with advances in accuracy and robustness of the system, and in the speed of stereo processing.

[1]This work was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract number W-7405-Eng-48. UCRL-JC-152416



(a) A 15-projector display with a total resolution of 6,400 by 3,072 used for interactive application.



(b) An example of interactive manipulation of isosurface and volume rendering parameters.

Figure 1: The "power wall" setup at LLNL. (Adapted with permission from ACI Visualization brochure).

2. Framework

In this Section, we describe the method for recognizing three gesture types: rotation, zoom, and translation. Given the 3-D trajectory of the manipulating hand, we fit a Bezier curve to the trajectory. The curvature of the curve is used to determine the gesture.

Gesture recognition involves five steps:

1. Detecting the manipulating hand
2. Identifying the beginning of the gesture,
3. Detecting the end of the gesture
4. Computing the 3-D trajectory of the manipulating hand
5. Recognizing the gesture by fitting the curve to the 3-D trajectory.

These steps are described in separate Sections of this paper.

2.1. Manipulating Hand Detection

This presents a brief overview of the process. For more details, please see [11].

Detecting skin regions. To identify a hand involved in virtual object manipulation, the color image is converted to the SCT color space to reduce lighting artifacts [10]. Skin pixels are selected with a minimum distance classifier using Mahalanobis distance (the system contains skin statistics collected off-line during training). After the noise removal (by a sequence of erosions and dilations), the skin image is segmented using connected component analysis. Small regions or unlikely shapes are removed from further consideration.

Identifying the hand involved in manipulation. We assume that the hand gesture occurs in front of the body. Therefore, the skin region closest to the camera is identified as the manipulating hand. We determine the 3-D location of the region by using histogram-based noise filtering, where all smaller bins are considered noise [11]. Hand gestures can be classified in general as object manipulations and other hand movements. Selection of meaningful gestures is described next.

2.2. Gesture On, Gesture Off

We recognize the gestures for manipulation of virtual objects by "grabbing" them. Both grabbing and releasing are detected by checking the area change of the hand. If the area has decreased, then the hand is closed and therefore the object is grabbed. If the area has increased, then the hand is opened and therefore the object is released. Since the speed of the hand closing and opening could vary, we build a history of hand area change. We accumulate the area change on the past three frames. Grabbing is detected when the accumulated change of area is less than -75%, and releasing is detected when the accumulated change is greater than 75%. Note that the negative area change means the area decreased.

Changes due to small movement or noise around the hand area could occur, and these should not be counted toward the determination of grabbing and releasing. Therefore, we ignore change of area smaller than 15%. We assume that the palm of the hand is facing the camera. No side views are currently allowed. However, the overlap between the hand and the face is acceptable.

2.3. Bezier Curves

Representation and Properties A Bezier curve uses control points to represent a curve.

$$C(u) = \sum_{i=0}^{n} B_{i,n}(u) P_i \tag{1}$$

Points along a curve are determined by using u, $0 \leq u \leq 1$. n indicates the degree of curve which is one less than the number of control points. P_i is the ith control point where $P(0) = C(0)$ and $P(n) = C(1)$. $B_{i,n}$ is a Bernstein

polynomial.

$$B_{i,n} = \frac{n!}{i!(n-i)!} n^i (1-u)^{n-i} \quad (2)$$

We use Bezier curves with three control points describing quadratic curves. The Bernstein polynomials with $n = 2$ are shown below.

$$B_{0,2} = (1-u)^2 \quad (3)$$
$$B_{1,2} = 2u(1-u) \quad (4)$$
$$B_{2,2} = u^2 \quad (5)$$

Therefore, the equation for the Bezier curve with three control points is simplified.

$$C(u) = (1-u)^2 + 2u(1-u) + u^2 \quad (6)$$

Curve Fitting - General Method We use the curve approximation method described by Piegl and Tiller [8]. We describe the method for fitting curves with n+1 control points with m+1 data points.

A B-spline curve is described as

$$C(u) = \sum_{i=0}^{n} N_{i,p}(u) P_i \quad (7)$$

where $0 \leq u \leq 1$. The details of finding $N_{i,p}$ is left out for now, since the values get very simple for Bezier curves with 3 control points.

Given a set of 3-D points to fit $Q_0, \ldots, Q_m$, we estimate control points $P_0, \ldots, P_n$ where $m > n$. $\bar{u}_k$ is computed using the length of the curve up to Q_k.

$$|Q| = \sum_{i=1}^{m} |Q_i - Q_{i-1}| \quad (8)$$
$$\bar{u}_k = \frac{\sum_{i=1}^{k} |Q_i - Q_{i-1}|}{|Q|} \quad (9)$$

We first set $P_0 = C(0) = Q_0$ and $P_n = C(1) = Q_m$. We then estimate the unknowns which are the intermediate control points $P_1, \cdots, P_{n-1}$ such that E is minimized.

$$E = \sum_{k=1}^{m-1} |Q_k - C(\bar{u}_k)|^2 \quad (10)$$

For the system with (n+1) control points, the estimation of P becomes solving

$$(N^T N)P = R \quad (11)$$

$$N = \begin{bmatrix} N_{1,p}(\bar{u}_1) & \cdots & N_{n-1,p}(\bar{u}_1) \\ \vdots & \ddots & \vdots \\ N_{1,p}(\bar{u}_{m-1}) & \cdots & N_{n-1,p}(\bar{u}_{m-1}) \end{bmatrix} \quad (12)$$

where R is the vector of $n-1$ points

$$R = \begin{bmatrix} N_{1,p}(\bar{u}_1)R_1 & + \cdots + & N_{1,p}(\bar{u}_{m-1})R_{m-1} \\ & \vdots & \\ N_{n-1,p}(\bar{u}_1)R_1 & + \cdots + & N_{n-1,p}(\bar{u}_{m-1})R_{m-1} \end{bmatrix} \quad (13)$$

Each R_k is defined as

$$R_k = Q_k - N_{0,p}(\bar{u}_k)Q_0 - N_{n,p}(\bar{u}_k)Q_m \quad (14)$$

$$P = \begin{bmatrix} P_1 \\ \vdots \\ P_{n-1} \end{bmatrix} \quad (15)$$

P is a matrix with unknown control points in each row where x, y, and z are stored as a separate element.

2.4. Recognition

Bezier Curve. A Bezier curve uses control points to represent a curve.

$$C(u) = \sum_{i=0}^{n} B_{i,n}(u) P_i \quad (16)$$

Points along a curve are determined by using u, $0 \leq u \leq 1$. n indicates the degree of curve which is one less than the number of control points. P_i is the ith control point where $P(0) = C(0)$ and $P(n) = C(1)$.

Good Input? The gesture is recognized using the trajectory between grabbing and releasing. However, the gesture is ambiguous if the movement is too small. Also, if the trajectory contains abruptly sampled or too few points, computing the shape of trajectory becomes difficult.

We consider the gesture to be invalid if the trajectory is shorter than 20cm, if the trajectory is irregularly sampled, or if the trajectory contains less than six points. We selected the minimum number of points to be six because (1) we observed that at least six points were needed to robustly fit to three control points, and (2) we assumed that the gesture roughly requires one or two seconds and the frame rate was close to six frames/sec.

In order to determine irregularity, we have computed U_k for all the points along the trajectory, then counted the points in three bins ranging from [0.0, 0.33], [0.34, 0.66], and [0.67, 1.0] with U_k values. If there are no items in any of the three bins, we consider the trajectory to be invalid.

Determining Gesture Types. A geometric (rather than statistical) model is motivated by the nature of the

application (representation of visualization operations) and its requirements

(robustness and simplicity). Using its angle and direction, the trajectory is classified into rotation, translation or zoom. Examples of three gestures are shown in Figure 2. The inside angle at the middle point is used to recognize the

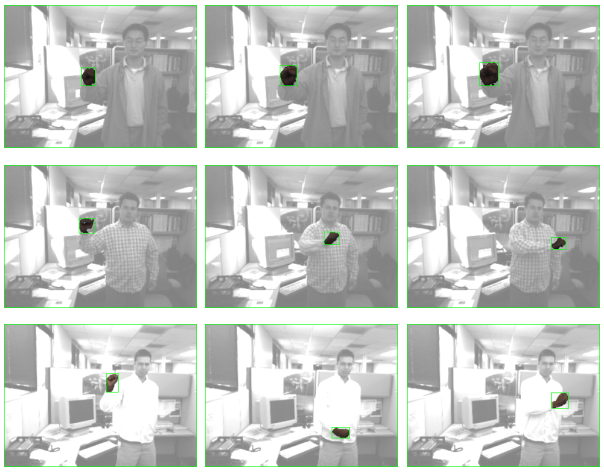


Figure 2: Selected frames for three sample gestures: zoom (top row), translate (mid row), and rotate (bottom row). Hands used for manipulation are (automatically) detected and shown in natural skin colors, whereas the rest of each image is displayed as greyscale for visualization.

gesture from 3-D trajectory Using two end control points (P_0, P_2) and the middle point ($C(0.5)$), we compute the inside angle of triangle inside the curve (θ) by

$$\vec{v_{10}} = C(0.5) - P_0 \quad (17)$$

$$\vec{v_{12}} = C(0.5) - P_2 \quad (18)$$

$$\vec{v_{10}} \cdot \vec{v_{12}} = |\vec{v_{10}}||\vec{v_{12}}|cos(\theta) \quad (19)$$

We classify "flat" trajectory as translation or zooming while "curved" as rotation. The inside angle would be smaller if the curve has high curvature. So if $\theta \leq 100°$, then we classify the gesture as rotation. Otherwise, the gesture is translation or zooming. If the movement was mostly in z direction, then we recognize it as "zooming" or otherwise "translation." To do so, we compute the ratio of the change of depth between first and last control points vs. distance between them.

$$\frac{|P_{2,z} - P_{0,z}|}{|P_2 - P_0|} \quad (20)$$

If the distance is greater than 0.5 then it is classified as zoom, otherwise it would be a translation.

3. Results

Experimental setup consists of a Digiclops system (Point Grey Research, [9]) on a Pentium 4 PC 1.5GHz with 512 MB RAM. The results are organized in four Sections: manipulating hand detection; manipulation mode detection; gesture recognition; and overall performance. Testing data set includes 100 gestures, 226 manipulation modes, and 1641 frames of manipulating hands of four people. We have presented initial manipulating hand tracking results on a smaller data set in [11]. The data was captured in an indoor setting with varying natural light and a complex background. Four different people with various skin tones participated in the experiments.

3.1. Manipulation Detection

Manipulating Hand Detection We detected the manipulating hand correctly in 1639 of 1641 images (99.9%). Two incidences of misdetection occured because the depth readings of the manipulating hand and face were identical.

Evaluating Manipulation Detection. Each manipulation mode is defined by the starting frame, end frame, and its mode (on or off). We evaluated the manipulation mode detection in four categories: correct, incorrect, over, and under detection. Detected manipulation mode on and off is shown as a bar as the frame number increase from left to right. Upper bar is for Algorithm Detected (AD) and lower bar is for Ground Truth (GT).

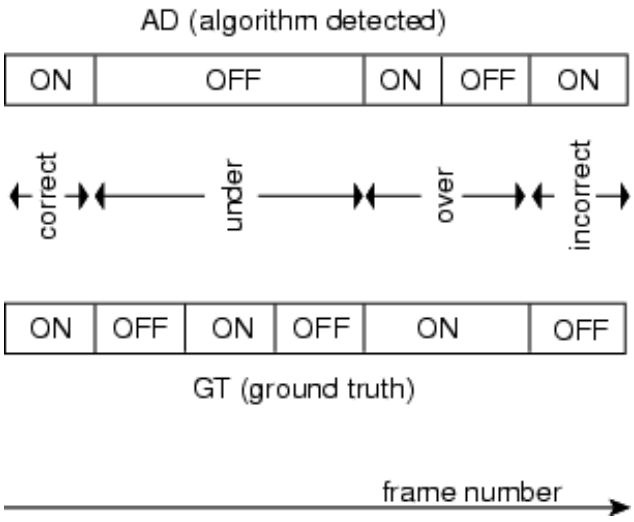


Figure 3: Evaluating Manipulation Mode Detection.

Four possible outcomes are classified below:

- **Correct detection.** AD and GT detections with at least 90% overlap in duration and same mode.
- **Incorrect detection.** AD and GT detections with at least 90% overlap in duration but different mode.
- **Over detection.** Multiple AD detections with at least 90% overlap in duration with one GT detection with same mode.
- **Under detection.** Multiple GT detections with at least 90% overlap in duration with one AD detection with same mode.

Results. The results of manipulation mode detections are shown in Table 1. The method correctly identified 216

Table 1: Manipulating Mode Detection

data set	correct	incorrect	over	under	total
1	19	0	0	0	19
2	35	0	0	0	35
3	11	0	0	1	13
4	24	0	0	1	25
5	8	0	0	0	8
6	30	0	0	0	30
7	31	0	0	0	31
8	16	0	0	1	19
9	18	0	0	1	21
10	24	0	0	1	25
total	216	0	0	5	226
total (%)	95.6	0.0	0.0	2.2	100.0

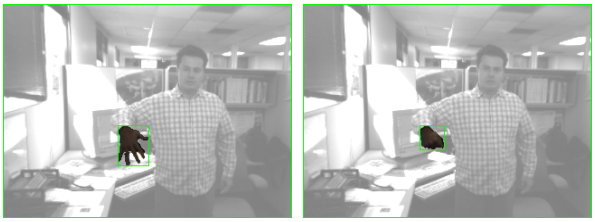


Figure 4: Two misidentifications of Manipulation Modes. Area change of 63% does not meet the threshold of +/-75%

of the 226 modes (95.6%). All five underdetections were due to missing the manipulation mode change (hand opening or closing). For these five missed detections, the area change was 63%, 74.7%, 71.4%, 68%, and -54%, which did not meet the threshold of +/-75%. The hand detections of 63% is shown in Figure 4.

3.2. Gesture Recognition

In order to compute the results of gesture recognition, we have taken the correctly identified manipulation modes and checked the recognition rate. Note that recognition rate is evaluating the goodness of recognition rate by using Bezier curve representation. The overall performance of evaluating the correctness of recognition versus intended recognition is shown in the next Section. The algorithm identified gestures correctly 93 out of 95 times (97.9%). For the two incorrect recognitions, the zoom gestures were recognized as rotation by estimating the inner angle (θ) as 104.9 and 125.1. The reason for the failure was due to using three-control points bezier curve, and the curves in question were too concave to be adequately described by three control points.

3.3. Overall Performance

Table 2: Overall Recognition Performance.

data	correct	incorrect	missed		total
		recog	mode	recog	
1	7	0	0	0	7
2	14	0	0	0	14
3	6	0	1	0	7
4	10	0	1	0	11
5	4	0	0	0	4
6	13	0	0	0	13
7	12	0	0	0	12
8	7	0	1	0	8
9	8	2	1	0	11
10	12	0	1	0	12
total	93	2	5	0	100

Overall performance is evaluating the number of correctly recognized gestures vs. the number of intended gestures (shown in Table 2.) The algorithm correctly recognized 93 out of 100 intended gestures. Note that in the previous Section, only 95 gestures were considered because 5 were not detected due to misdetection of manipulation mode as shown in Section 3.1. Table 2 also shows the reasons for five non-correct classifications of gestures. We have divided those five incidences into "incorrect" if the gesture was incorrectly recognized as wrong gesture and "missed" if the gesture was not detected. "Incorrect" and "missed" are then subdivided into failures of three steps: "hand" (manipulating hand detection), "mode" (manipulating mode detection), and "recognition" (gesture recognition). Two gestures were incorrectly recognized due to a small angle as mentioned in Section 3.2. Three gestures were missed because the mode detection underdetected the gesture as mentioned in Table 1. The system executed at the rate of 5.5 frames/sec.

We obtained average of 5.5 frames/sec with Dynamic Disparity Range Adjustment [11], and 3.4 frs/sec without it yielding the speed up of 60.1%.

3.4. Application to Virtual Manipulation

Figure 5 shows the application of recognized gestural commands to a small virtual object. The complex virtual objects (3D view of two chromosomes) have been selected to provide a visual illustration of discussed operations. The results of three sequentially applied virtual manipulations through gestures of rotation, zoom, and translation are shown in Figure 5. Both recognition and visualization are in real-time. The latter part is written in OpenGL. Real-time

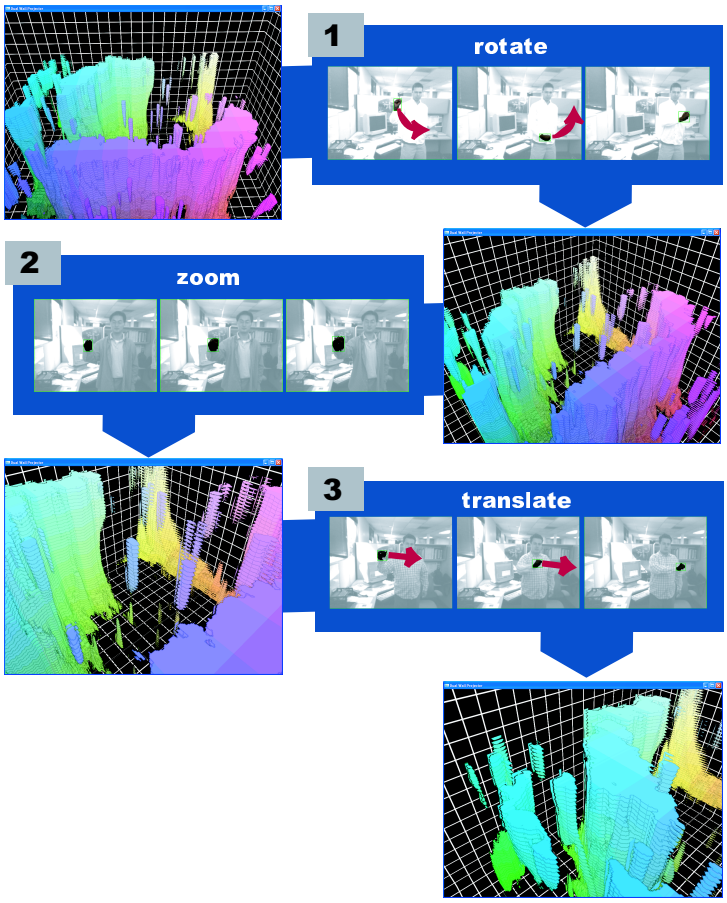


Figure 5: Example of Virtual Manipulation Application. Three sequential manipulations are shown. The red arrows are super imposed on images to enhance the appearance of gesture trajectories.

visualization of large data sets represents a significant bottleneck, which is currently under investigation by various research groups.

4. Conclusions

Visual data exploration has tremendous capabilities for revealing properties and abnormalities in large data sets. This paper described a gesture recognition system for visualization navigation. Scientists are interested in developing interactive settings for exploring large data sets in an intuitive environment. The input consists of registered 3-D data. Bezier curves are used for trajectory analysis and classification of gestures. The method is robust and reliable: correct hand identification rate is 99.9% (from 1639 frames), modes of hand movements are correct 95.6% of the time, recognition rate (given the right mode) is 97.9%. It advances the state-of-the-art of human-computer interaction with a robust attachment- and marker-free gestural information processing for visualization. An application to gesture-controlled visualization is also presented. Future work includes defining a larger set of gestures based on the basic movements already analyzed, using two hands in manipulation, and creating actual hand models for detection of smaller, more subtle gestures.

Acknowledgments

We would like to thank Benjamin Lok at UNC Charlotte Virtual Environment Group for providing the example of virtual manipulation (Figure 5). We would like to thank the LLNL VIEWS Visualization project for Figure 1(a); the example data set appears as Figure 1(b) courtesy of Art Mirin of LLNL.

References

[1] J. K. Aggarwal and Q. Cai. Human motion analysis: A review. In *IEEE Nonrigid and Articulated Motion Workshop*, pages 90–102, San Juan, Puerto Rico, June 1997.

[2] A. F. Bobick and A. D. Wilson. A state-based approach to the representation and recognition of gesture. *IEEE Trans. on PAMI*, 19(12):1325–1337, December 1997.

[3] P. Hong, M. Turk, and T. S. Huang. Constructing finite state machines for fast gesture recognition. In *Proc. 15th ICPR*, volume 3, pages 691–694, Barcelona, Spain, September 2000.

[4] H. Hongo, M. Ohya, M. Yasumoto, and K. Yamamoto. Face and hand gesture recognition for human-computer interaction. In *Proc. 15th ICPR*, volume 2, pages 921 – 924, Barcelona, Spain, September 2000.

[5] C. Lee and Y. Xu. Trajectory fitting with smoothing splines using velocity information. In *IEEE Intern. Conf. on Robotics and Automation*, pages 2796–2801, vol.3, San Francisco, CA, April 2000.

[6] D. J. Moore, I. A. Essa, and M. H. Hayes III. Exploiting human actions and object context for recognition tasks. In *Proceedings of ICCV*, pages 80–86, Corfu, Greece, September 1999.

[7] V. I. Pavlovic, R. Sharma, and T. S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Trans. on PAMI*, 19(7):677–695, July 1997.

[8] L. Piegl and W. Tiller. *The NURBS Book*. Springer-Verlag, New York, 1997.

[9] Point Grey Research Inc. *Digiclops Stereo Vision System, User's guide and command reference*. Inc., Point Grey Research, Vancouver, BC, 2000.

[10] M. W. Powell and R. Murphy. Position estimation of micro-rovers using a spherical coordinate transform color segmenter. In *Proceedings of IEEE Workshop on Photometric Modeling for Computer Vision and Graphics*, pages 21–27, Fort Collins, CO, June 1999.

[11] L. V. Tsap and M. C. Shin. Improving quality and speed of range data for communication. In *Proc. IEEE International Workshop on Cues in Communication*.

[12] C. Vogler, H. Sun, and D. Metaxas. A framework for motion recognition with applications to American sign language and gait recognition. In *Proc. Workshop on Human Motion*, pages 33 – 38, Los Alamitos, CA, December 2001.

[13] Y. Yacoob and M. J. Black. Parameterized modeling and recognition of activities. *Journal of CVIU*, 73(2):232–247, 1999.

[14] M.-H. Yang and N. Ahuja. Recognizing hand gestures using motion trajectories. In *Proceedings of CVPR*, volume 1, pages 466–472, Fort Collins, CO, June 1999.