Appears in *Proc. of the IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, Austin, Texas, November 1994, pages 16-22.

DigitEyes: Vision-Based Hand Tracking for Human-Computer Interaction

James M. Rehg*

Dept. of Electrical and Computer Eng.
Carnegie Mellon University
Pittsburgh, PA 15213
jimr@cs.cmu.edu

Takeo Kanade

The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
tk@cs.cmu.edu

Abstract

Computer sensing of hand and limb motion is an important problem for applications in Human-Computer Interaction (HCI), virtual reality, and athletic performance measurement. Commercially available sensors are invasive, and require the user to wear gloves or targets. We have developed a noninvasive vision-based hand tracking system, called DigitEyes. *Employing a kinematic hand model, the* DigitEyes *system has demonstrated tracking performance at speeds of up to 10 Hz, using line and point features extracted from gray scale images of unadorned, unmarked hands. We describe an application of our sensor to a 3D mouse user-interface problem.*

1 Introduction

A "human sensor" capable of tracking a person's spatial motion using techniques from Computer Vision would be a powerful tool for human-computer interfaces. Such a sensor could be located in the user's environment (rather than on their person) and could operate under natural conditions of lighting and dress, providing a degree of convenience and flexibility that is currently unavailable. For the purpose of visual sensing, human hands and limbs can be modeled as *articulated mechanisms*, systems of rigid bodies connected together by joints with one or more degrees of freedom (DOF's). This model can be applied at a fine (visual) scale to describe hand motion, and at a coarser scale to describe the motion of the entire body. Based on this observation, we formulate human sensing as the real-time visual tracking of articulated kinematic chains.

Although many frameworks for human motion analysis are possible, our approach has four main advantages. First, by tracking all of the hand's DOF's, we provide the user with maximum flexibility for interface applications. (See [15, 6] for examples of interfaces requiring a whole-hand sensor.) In addition, our general modeling approach based on 3D kinematics makes it possible to track any subset of hand or body states with the same basic algorithm. Another benefit of full state tracking is invariance to unused hand motions. The motion of a particular finger, for example, can be recognized from its joint angles regardless of the pose of the palm relative to the camera. Finally, by modeling the hand kinematics in 3D we eliminate the need for application- or viewpoint-dependent user modeling.

The *DigitEyes* system treats hand tracking as a model-based sequential estimation problem: given a sequence of images and a hand model, we estimate the 3D hand configuration in each frame. All possible hand configurations are represented by vectors in a state space, which encodes the pose of the palm (six rotation and translation DOF's[1]) and the joint angles of the fingers (four states per finger, five for the thumb). Each hand configuration generates a set of image features, 2D lines and points, by projection through the camera model. A feature measurement process extracts these hand features from grey-scale images by detecting the occluding boundaries of finger links and tips. The state estimate for each image is computed by finding the state vector that best fits the measured features. Our basic tracking framework

*Supported by the NASA George Marshall Space Flight Center, GSRP Grant NGT-50559.

[1] We use quaternions to represent palm rotation, resulting in a model with four rotational states—- one more than the number of DOF's. Although quaternions are a nonminimal representation, they have the advantage of being free of singularities.

is similar to that of [4, 7, 16].

Articulated mechanisms are more difficult to track than the single rigid objects traditionally addressed in Computer Vision. Three major difficulties are the large size of the state space, nonlinearities in the state-to-feature mapping (called the measurement model), and self-occlusions. Finger articulations add an additional 21 states over the rigid body motion of the palm, significantly increasing the computational cost of estimation. These additional states are parameterized by joint angles, which introduce nonlinearities and kinematic singularities into the measurement model. Singularities arise when a small change in a given state has no effect on the image features. In addition to these problems, the fingers occlude each other and the palm during motion, making feature measurement difficult.

The *DigitEyes* system uses local search and leariztion to deal with the large state space and nonlinear measurement model. The key to our local, gradient-based approach to tracking is a high image acquisition rate (10 Hz), which limits the change in the hand state, and therefore image feature location, between frames. In the state space, we exploit this locality by linearizing the nonlinear state model around the previous estimate. Techniques from robotics provide for fast computation of the necessary kinematic Jacobian. Kinematic singularities are dealt with by stabilizing the state estimator. The resulting linear estimation problem is solved for each frame, producing a sequence of state corrections which are integrated over time to yield an estimated state trajectory.

As a result of the high image sampling rate, the change in hand features between frames is also small. For a given image, the state estimate from the previous frame is used to predict feature positions. Feature detectors, initialized to these predictions, exploit the symmetry of the finger links to extract lines and points and match them to the hand model. In the first image of the sequence, the user places his hand in a known starting configuration to initialize tracking. In the current system, each finger link detects its features independently of the others, which limits the sensor to hand motions without occlusions. We are extending our feature processing approach to remove this limitation.

In [12], we described the *DigitEyes* system in detail, and gave results of tracking a 27 DOF hand model from a two camera image sequence under orthographic projection. This paper describes an extension to perspective projection, and gives a detailed example of a user-interface based on our sensor: a 3D graphical mouse. While difficult problems still remain in tracking through occlusions and across complicated backgrounds, these results demonstrate the potential of our approach to vision-based human motion sensing.

2 Previous Work

Previous work on tracking general articulated objects includes [8, 16, 10, 9]. In [16], Yamamoto and Koshikawa describe a system for human body tracking using kinematic and geometric models. They give an example of tracking a single human arm and torso using optical flow features. Pentland and Horowitz [10] give an example of tracking the motion of a human figure using optical flow and an articulated deformable model. In a related approach, Metaxis and Terzopoulos [8] track articulated motion using deformable superquadric models. In [4], Dorner describes a system for interpreting American Sign Language from image sequences of a single hand. Dorner's system uses the full set of the hand's DOFs, and employs a glove with colored markers to simplify feature extraction. One of the earliest systems, by O'Rourke and Badler [9], analyzed human body motion using constraint propagation. None of these earlier approachs based on articulated models have demonstrated real-time tracking results for the full state of a complicated mechanism like the human hand, using natural image features.

In addition to previous work on articulated object tracking, many authors have applied general vision techniques to human motion analysis. In contrast to *DigitEyes*, these approachs analyze a subset of the total hand motion, such as a set of gestures [2, 13] or the rigid motion of the palm [1]. Darrell and Pentland describe a system for learning and recognizing dynamic hand gestures in [2]. Related work by Segen [13] takes a neural network approach to 2D hand gesture recognition. Both of these approachs work in real-time on unmarked hand images, but they don't produce 3D motion estimates and it would be difficult to apply them to problems like the 3D mouse interface in Subsect. 5. In [1], Blake et. al. describe a real-time contour tracking system that can follow the silhouette of a *rigidly* moving hand under an affine motion model.

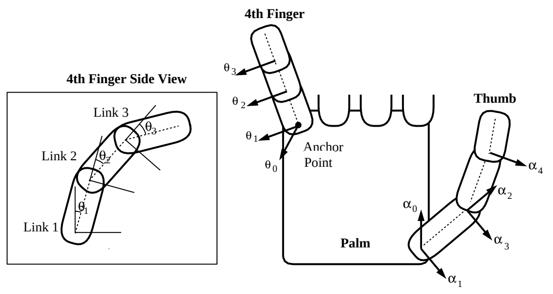

Figure 1: Kinematic models, illustrated for fourth finger and thumb. The arrows illustrate the joint axes for each link in the chain.

3 Hand Modeling for Visual Tracking

This section is a brief summary of our kinematic and feature models of the hand. See [12] for more detail. We use the Denavit-Hartenburg (DH) representation, widely used in robotics [14], to describe the hand kinematics. We make several simplifying assumptions in modeling the hand, which are illustrated in Fig. 1. First, we assume that each of the four fingers of the hand are planar mechanisms with four degrees of freedom (DOF). The abduction DOF moves the plane of the finger relative to the palm, while the remaining 3 DOF determine the finger's configuration within the plane. Each finger has an *anchor point*, which is the position of its base joint center in the frame of the palm, which is assumed to be rigid. Real fingers deviate from our modeling assumptions slightly, but we have found them to be adequate in practice.

Hand features consist of lines and points generated by the projection of the hand model into the image plane. Each finger link, modeled by a cylinder, generates a pair of lines in the image corresponding to its occlusion boundaries. The bisector of these lines, which contains the projection of the cylinder central axis, is used as the link feature. The link feature vector $[a\, b\, \rho]$ gives the parameters of the line equation $ax + by - \rho = 0$. Using the central axis line as the link feature eliminates the need to model the cylinder radius or the slope of the pair of lines relative to the central axis, which is often significant near the finger tips. We use the entire line because the endpoints are difficult to measure in practice. Fig. 2 shows two link feature lines extracted from the first two links of a finger. These features can be extracted from grey-scale hand images by a simple local operator, described in [12].

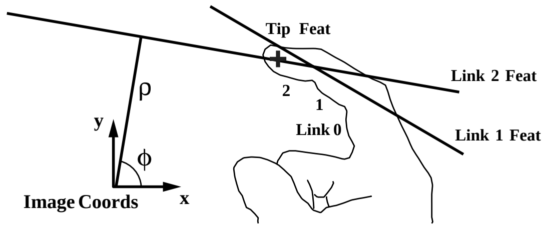

Figure 2: Features used in hand tracking are illustrated for finger links 1 and 2, and the tip. Each infinite line feature is the projection of the finger link central axis.

4 State Estimation for Articulated Mechanisms

We take a nonlinear least squares approach to state estimation. We form a residual in each frame that measures the difference between the measured image features and the projection of the estimated hand model into the image. The state correction is obtained by minimizing the squared magnitude of the residual vector through a modified Gauss-Newton (GN) algorithm [3]. Let $\mathbf{R}(\mathbf{q}_j)$ be the residual vector for image j. The GN state update equation is given by

$$\mathbf{q}_{j+1} = \mathbf{q}_j - [\mathbf{J}_j^t \mathbf{J}_j + \mathbf{S}]^{-1} \mathbf{J}_j^t \mathbf{R}_j \ , \qquad (1)$$

where $\mathbf{J}_j$ is the Jacobian matrix for the residual $\mathbf{R}_j$, both of which are evaluated at $\mathbf{q}_j$. The sources of nonlinearity in this model are trigonometric terms in the joint angles and inverse depth terms from perspective projection. $\mathbf{S}$ is a constant diagonal conditioning matrix used to stabilize the least squares solution in the presence of kinematic singularities. This approach is based on the rigid body tracking work of Lowe [7].

The remainder of this section gives the derivation of the link and tip residual Jacobians in both the orthographic and projective cases. The residual derivatives can be obtained by simple image plane projections of the gradient of 3D point positions with respect to the state. These gradients can be computed quickly using standard robotic techniques.

The residual measures the image plane distance between projected 3D hand geometry (lines and points) and detected features. It is zero when the hand model is perfectly aligned with the image data. The residual for the ith tip feature is a vector in the image plane defined by

$$\mathbf{v}_i(\mathbf{q}) = \mathbf{H}[\mathbf{P}_i(\mathbf{q})] - \mathbf{t}_i \ , \qquad (2)$$

where $\mathbf{P}_i = [X_i \, Y_i \, Z_i]$ is the 3D tip center (in the camera coordinate frame) as a function of the hand state, $\mathbf{H}[\cdot]$ gives the projection of a 3D point into the image plane, and $\mathbf{t}_i$ is the measured tip feature.

The link residual measures the deviation of points on the projected cylinder axis from the measured feature line. It is given by

$$l_i(\mathbf{q}) = \mathbf{m}^t \mathbf{H}[\mathbf{P}_i(\mathbf{q})] - \rho \ , \tag{3}$$

where $\mathbf{m} = [a \, b]^t$.

The point projection operator used in (2) and (3) can be written

$$\mathbf{H}[\mathbf{P}(\mathbf{q})] = \mathbf{C}\bar{\mathbf{P}}(\mathbf{q}) \ \text{ where } \mathbf{C} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \end{bmatrix} \tag{4}$$

is the intrinsic camera model (see [5]) and $\bar{\mathbf{P}}$ is the ideal point projection model. In the orthographic case, $\bar{\mathbf{P}} = [X \ \ Y \ \ 1]$, while in the projective case, $\bar{\mathbf{P}} = [X/Z \ Y/Z \ 1]$.

To calculate the tip and link Jacobians, we will need $\partial\bar{\mathbf{P}}/\partial\mathbf{q}$. In the orthographic case, we have

$$\frac{\partial\bar{\mathbf{P}}}{\partial\mathbf{q}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{\partial\mathbf{P}}{\partial\mathbf{q}} = \mathbf{A}_o \frac{\partial\mathbf{P}}{\partial\mathbf{q}} \ , \tag{5}$$

while in the perspective case we have

$$\frac{\partial\bar{\mathbf{P}}}{\partial\mathbf{q}} = \begin{bmatrix} 1/Z & 0 & -X/Z^2 \\ 0 & 1/Z & -Y/Z^2 \\ 0 & 0 & 0 \end{bmatrix} \frac{\partial\mathbf{P}}{\partial\mathbf{q}} = \mathbf{A}_p \frac{\partial\mathbf{P}}{\partial\mathbf{q}} \ . \tag{6}$$

To calculate the tip Jacobian, we differentiate (2) with respect to the state vector, obtaining

$$\frac{\partial\mathbf{v}_i(\mathbf{q})}{\partial\mathbf{q}} = \mathbf{C}\frac{\partial\bar{\mathbf{P}}_i(\mathbf{q})}{\partial\mathbf{q}} = \mathbf{H}\frac{\partial\mathbf{P}_i(\mathbf{q})}{\partial\mathbf{q}} \ , \tag{7}$$

where $\mathbf{H} = \mathbf{C}\mathbf{A}_{o/p}$ is a 2x3 gradient projection matrix and the choice of $\mathbf{A}_o$ or $\mathbf{A}_p$ is determined by the projection model.

Similarly, to calculate the link Jacobian we differentiate (3), obtaining

$$\frac{\partial l_i(\mathbf{q})}{\partial\mathbf{q}} = \tilde{\mathbf{m}}^t \frac{\partial\mathbf{P}_i(\mathbf{q})}{\partial\mathbf{q}} \ , \tag{8}$$

where $\tilde{\mathbf{m}} = \mathbf{m}^t\mathbf{C}\mathbf{A}_{o/p}$. Substituting (4), (5), and (6), we obtain:

$$\tilde{\mathbf{m}}_o = [a\alpha_u \ b\alpha_v \ 0] \tag{9}$$

$$\tilde{\mathbf{m}}_p = [a\alpha_u/Z \quad b\alpha_v/Z \quad -(a\alpha_u X + b\alpha_v Y)/Z^2] \ . \tag{10}$$

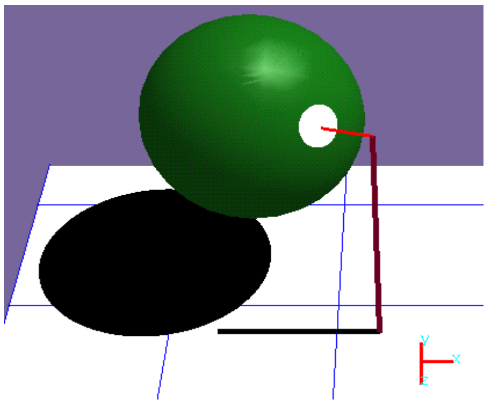

Figure 3: A sample graphical environment for a 3D mouse.

The gradient vector for link i constitutes one row of the total Jacobian matrix. Geometrically, it is formed by projecting the *kinematic Jacobian* for points on the link, $\partial\mathbf{P}_i(\mathbf{q})/\partial\mathbf{q}$, in the direction of the feature edge normal.

The kinematic Jacobians in (7) and (8) are composed of terms of the form $\partial\mathbf{P}_i/\partial\mathbf{q}_j$, which arise frequently in robot control and can be obtained directly from the kinematic model. See [11] for more details.

5 3D Graphical Mouse: Experimental Results

We describe in detail an experiment with a 3D graphical mouse user-interface based on the *DigitEyes* sensor and a simplified, eight state hand model. Figure 3 shows the simple 3D graphical environment used in the interface. It consists of a ground plane, a 3D cursor (drawn as a pole, with the cursor at the top), and a spherical object (being manipulated.) Shadows generate additional depth cues. The interface problem is to provide the user with control of the cursor's three DOF's.

The *DigitEyes* solution is based on a partial hand model, consisting of the first and fourth fingers of the hand, along with the thumb. The palm is constrained to lie in the plane of the table (3 DOF) used in the interface. The first finger has 3 articulated DOFs, while the fourth finger and thumb each have a single DOF allowing them to rotate in the plan of the table (abduct). The hand model is illustrated in Fig. 4, and a corresponding intensity image is given in Fig. 5. A single camera oriented at approximately 45 degrees to the table top acquires the images used in

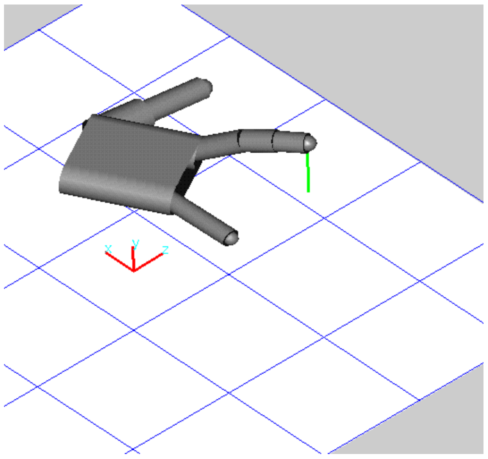

Figure 4: The hand model used in the 3D mouse application is shown in its estimated state from frame 200 in the motion sequence. The vertical line shows the height of the tip above the ground plane.

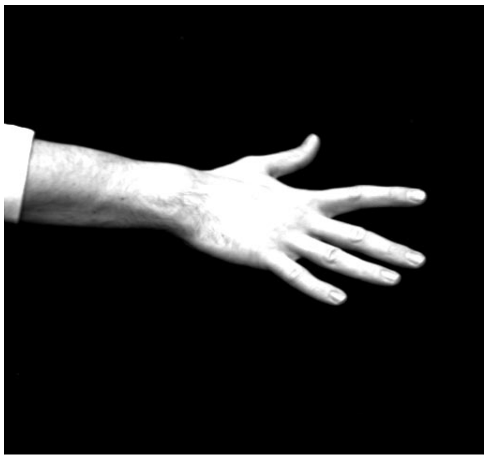

Figure 5: The input hand image from frame 200 in the motion sequence.

tracking. The palm position on the table controls the base position of the pole, while the height of the index finger above the table controls the height of the cursor. This particular mapping has the important advantage of decoupling the controlled DOF's, while making it possible to operate them simultaneously. The fourth finger and thumb have abduction DOF's in the plane, and are used as "buttons".

Figures 6 – 10 give experimental results from a 500 frame motion sequence. Figures 7, 8, and 9 show the estimated hand state for each frame. Frames were acquired at 100 ms sampling intervals. The pole heights and base positions derived from the estimated hand states are depicted in Fig. 10. The motion sequence has four phases. In the first phase (frame 0 to 150), the user's finger is raised and lowered twice, producing two peaks in the pole height, with a small variation in the estimated pole position. Second, around frame 150 the finger is raised again and kept elevated, while the thumb is actuated, as for a "button event". The actuation period is from frame 150 to frame 200, and results in some change in the pole height, but negligible change in pole position. Third, from 200 to 350, the pole height is held constant while the pole position is varied. Finally, from 350 to the end of the sequence all states are varied simultaneously. Figures 4 and 5 show an input image and its associated state estimate for frame 200. Sample mouse pole positions throughout the sequence are illustrated in Fig. 6.

These results demonstrate fairly good decoupling between the desired states and a useful dynamic range of motion. The largest coupling error occurs around frame 150 when the pole height drops as the thumb is actuated. For simplicity, the mouse state is generated from the hand state by a simple scaling and coordinate change, which tends to amplify sensor noise. Both of these problems could be alleviated by more sophisticated processing of the sensor output.

This example illustrates an important advantage of hand tracking with kinematic models: absolute 3D distances (such as finger height above a table) can be measured from a single camera image. The ability to recover 3D spatial quantities from hand motion is one of the advantages our system has over approachs based on gesture recognition.

In the above experiment, *DigitEyes* was implemented on a 68040-based image processing platform called IC40. State estimates were transfered by Ethernet to a Silicon Graphics Indigo 2 for model display and user-interface demonstrations. The IC40 board delivers digitized images to processor memory at video rate with no computational overhead, removing an important I/O bottleneck.

6 Conclusion

We have presented the *DigitEyes* framework for visual hand tracking and its application to a 3D mouse user-interface problem. We are currently extending these results in two ways. First, we are modifying our feature extraction process to handle occlusions and complicated backgrounds. Second, we are analyzing the observability requirements of articulated object tracking.

The authors would like to thank Omead Amidi and Yuji Mesaki for help with the IC40 hardware environment, and the first reviewer for his many detailed comments.

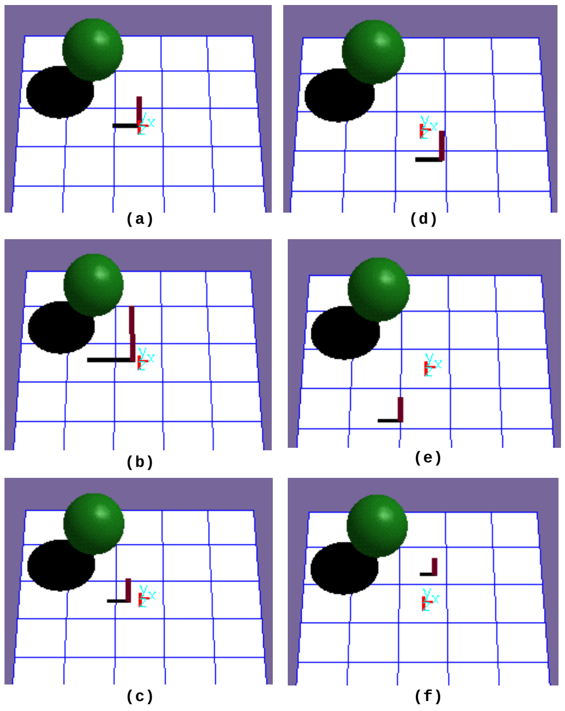

Figure 6: The mouse pole cursor at six positions during the motion sequence. The pole is the vertical line with a horizontal shadow, and is the only thing moving in the sequence. Samples were taken at frames 0, 30, 75, 260, 300, and 370 (chosen to illustrate the range of motion).

References

[1] A. Blake, R. Curwen, and A. Zisserman. A framework for spatiotemporal control in the tracking of visual contours. *Int. J. Computer Vision*, 11(2):127–145, 1993.

[2] T. Darrell and A. Pentland. Space-time gestures. In *Looking at People Workshop*, Chambery, France, 1993.

[3] J. Dennis and R. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, NJ, 1983.

[4] B. Dorner. Hand shape identification and tracking for sign language interpretation. In *Looking at People Workshop*, Chambery, France, 1993.

[5] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, 1993.

[6] S. B. Kang and K. Ikeuchi. Grasp recognition using the contact web. In *Proc. IEEE/RSJ Int. Conf. on Int. Robots and Sys.*, Raleigh, NC, 1992.

[7] D. Lowe. Robust model-based motion tracking through the integration of search and estimation. *Int. J. Computer Vision*, 8(2):113–122, 1992.

[8] D. Metaxis and D. Terzopoulos. Shape and nonrigid motion estimation through physics-based synthesis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(6):580–591, 1993.

[9] J. O'Rourke and N. Badler. Model-based image analysis of human motion using constraint propagation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2(6):522–536, 1980.

[10] A. Pentland and B. Horowitz. Recovery of nonrigid motion and structure. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(7):730–742, 1991.

[11] J. Rehg and T. Kanade. Digiteyes: Vision-based human hand tracking. Technical Report CMU-CS-TR-93-220, Carnegie Mellon Univ. School of Comp. Sci., 1993.

[12] J. Rehg and T. Kanade. Visual tracking of high dof articulated structures: an application to human hand tracking. In J. Eklundh, editor, *Proc. of Third European Conf. on Computer Vision*, volume 2, pages 35–46, Stockholm, Sweden, 1994. Springer-Verlag.

[13] J. Segen. Gest: A learning computer vision system that recognizes hand gestures. In R. Michalski and G. Tecuci, editors, *Machine Learning IV*. Morgan Kaufmann, 1993.

[14] M. Spong. *Robot Dynamics and Control*. John Wiley and Sons, 1989.

[15] D. Sturman. *Whole-Hand Input*. PhD thesis, Massachusetts Institute of Technology, 1992.

[16] M. Yamamoto and K. Koshikawa. Human motion analysis based on a robot arm model. In *IEEE Conf. Comput. Vis. and Pattern Rec.*, pages 664–665, 1991.

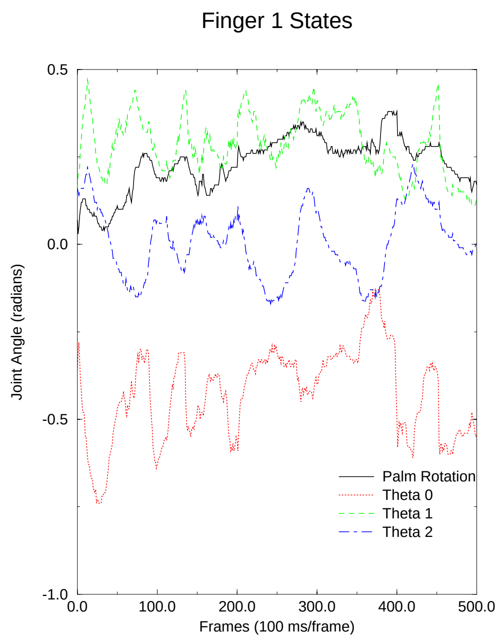

Figure 7: Palm rotation and first finger joint angles for mouse pole hand model.

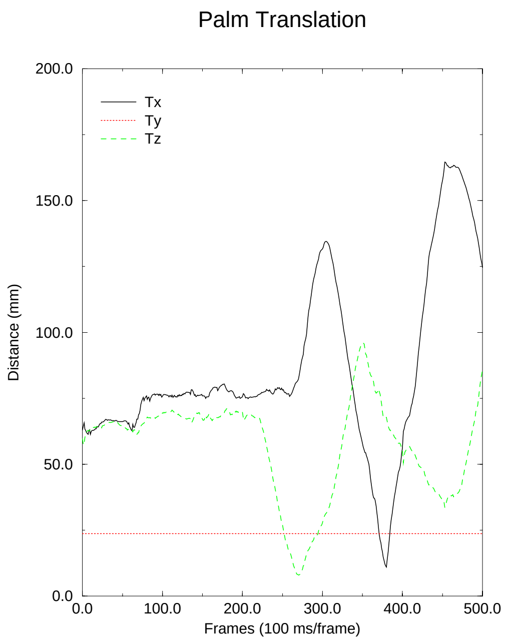

Figure 9: Estimated translation states for mouse pole hand model. The Y axis motion is constrained to zero due to tabletop.

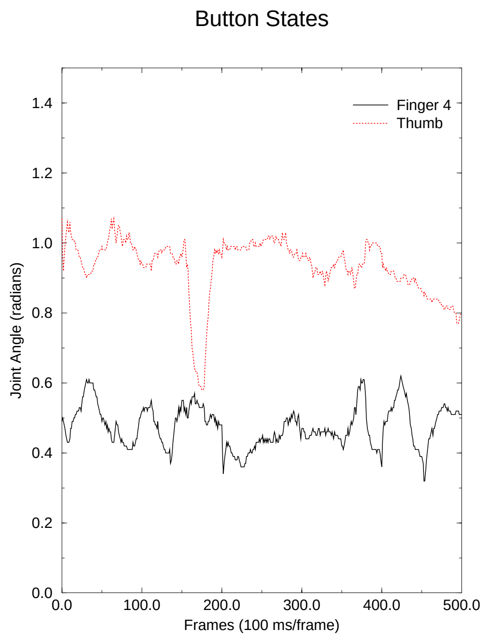

Figure 8: Joint angles for thumb and fourth finger of mouse pole hand model. Note the “button event” signaled by the thumb motion around frame 175.

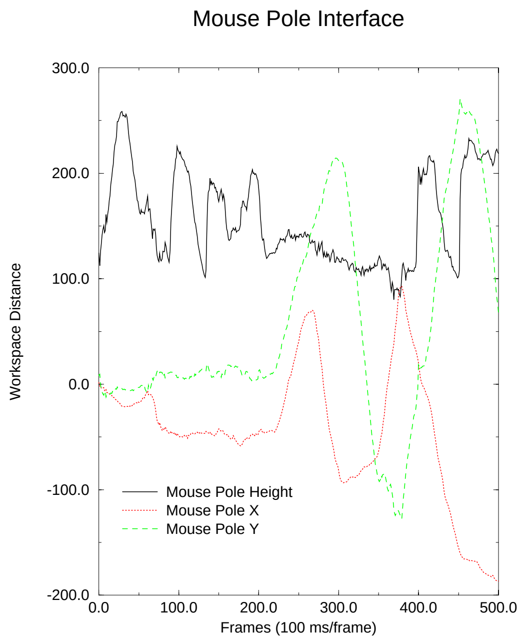

Figure 10: Mouse pole states derived from the *DigitEyes* sensor during the motion sequence.