Real-Time Fingertip Tracking and Gesture Recognition

Kenji Oka and Yoichi Sato
University of Tokyo

Hideki Koike
University of Electro-Communications, Tokyo

Augmented desk interfaces and other virtual reality systems depend on accurate, real-time hand and fingertip tracking for seamless integration between real objects and associated digital information. We introduce a method for discerning fingertip locations in image frames and measuring fingertips trajectories across image frames. We also propose a mechanism for combining direct manipulation and symbolic gestures based on multiple fingertip motions.

Our method uses a filtering technique, in addition to detecting fingertips in each image frame, to predict fingertip locations in successive image frames and to examine the correspondences between the predicted locations and detected fingertips. This lets us obtain multiple fingertips' trajectories in real time and improves fingertip tracking. This method can track multiple fingertips reliably even on a complex background under changing lighting conditions without invasive devices or color markers.

Our hand and fingertip tracking method, developed for augmented desk interface systems, reliably tracks multiple fingertips and hand gestures against complex backgrounds and under dynamic lighting conditions without any markers.

Distinguishing the thumb lets us differentiate manipulative (extended thumb) from symbolic (folded thumb) gestures. We base this on the observation that users generally use only a thumb and forefinger in fine manipulation. The method then uses the Hidden Markov Model (HMM),[1] which interprets hand and finger motions as symbolic events based on a probabilistic framework, to recognize symbolic gestures for application to interactive systems. Other researchers have used HMM to recognize body, hand, and finger motions.[2,3]

Augmented desk interfaces

Several augmented desk interface systems have been developed recently.[4,5] One of the earliest attempts in this domain, DigitalDesk,[6] uses a charge-coupled device (CCD) camera and a video projector to let users operate projected desktop applications using a fingertip. Inspired by DigitalDesk, we've developed an augmented desk interface system, EnhancedDesk[7] (Figure 1), that lets users perform tasks by manipulating both physical and electronically displayed objects simultaneously with their own hands and fingers.

Figure 2 shows an application of our proposed tracking and gesture recognition methods.[8] This two-handed drawing tool assigns different roles to each hand. After selecting radial menus with the left hand, users draw objects or select objects to be manipulated with the right hand. For example, to color an object, a user selects the color menu with the left hand and indicates the object to be colored with the right hand (Figure 2a). The system also uses gesture recognition to let users draw objects such as circles, ellipses, triangles, and rectangles and directly manipulate them using the right hand and fingers (Figure 2b).

Real-time fingertip tracking

This work evolves from other vision-based hand and finger tracking methods (see the "Related Work" sidebar on p. 66), including our earlier multiple-fingertip tracking method.[9]

Detecting multiple fingertips in an image frame

We must first extract multiple fingertips in each input image frame in real time.

Extracting hand regions. Extracting hands based on color image segmentation or background subtraction often fails when the scene has a complicated background and dynamic lighting. We therefore use an infrared camera adjusted to measure a temperature range approximating human body temperature (30 to 34 degrees C). This raises pixel values corresponding to human skin above that for other pixels (Figure 3a). Therefore, even with complex backgrounds and changing light, our system easily identifies image regions corresponding to human skin by binarizing the input image with a proper threshold value. Because hand tempera-

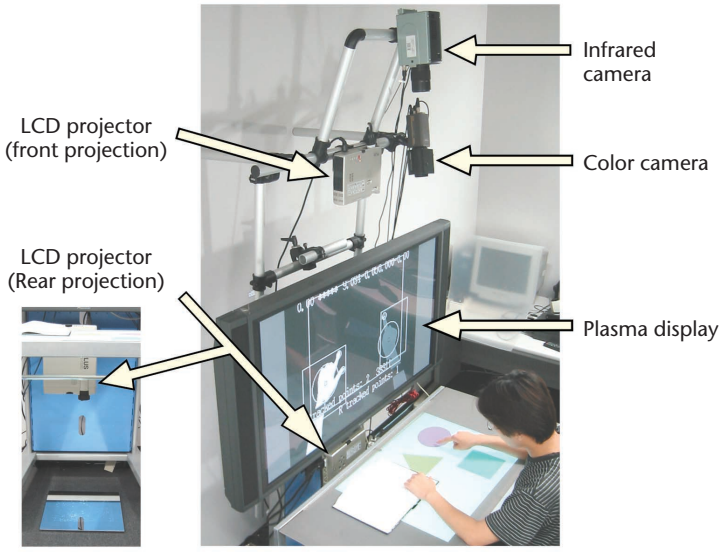
0272-1716/02/$17.00 © 2002 IEEE

ture varies somewhat among people, our system determines an appropriate threshold value for image binarization during initialization by examining the histogram of an image of a user's hand placed open on a desk. It similarly obtains other parameters such as approximate hand and fingertip sizes.

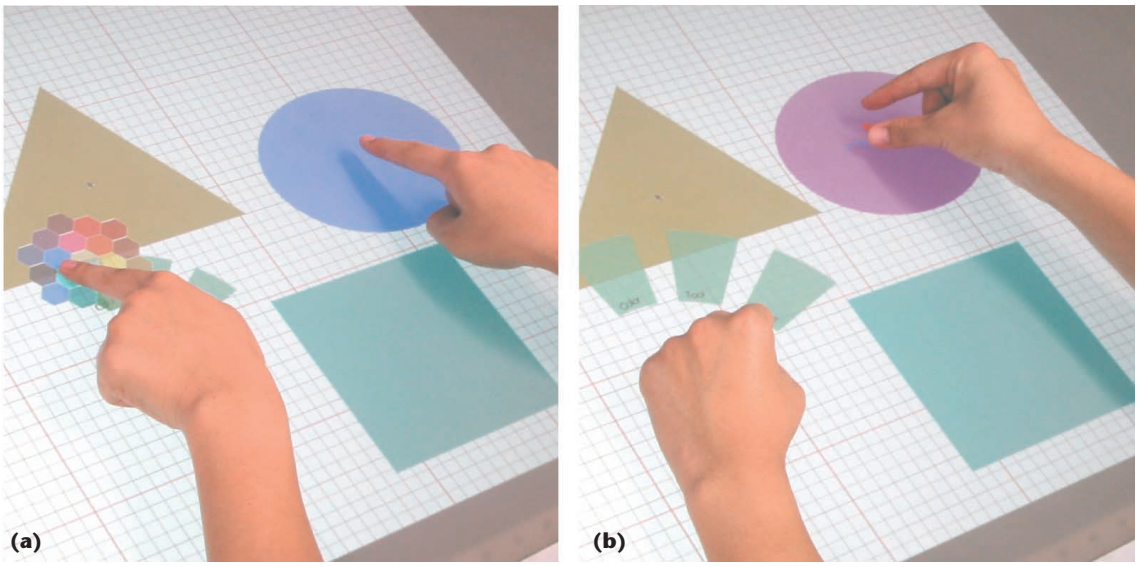
We then remove small regions from the binarized image and select the two largest regions to obtain an image of both hands.

Finding fingertips. Once we've found a user's arm regions, including hands, in an input image, we search for fingertips within those regions. This search process is more computationally expensive than arm extraction, so we define search windows for the fingertips rather than searching the entire arm region.

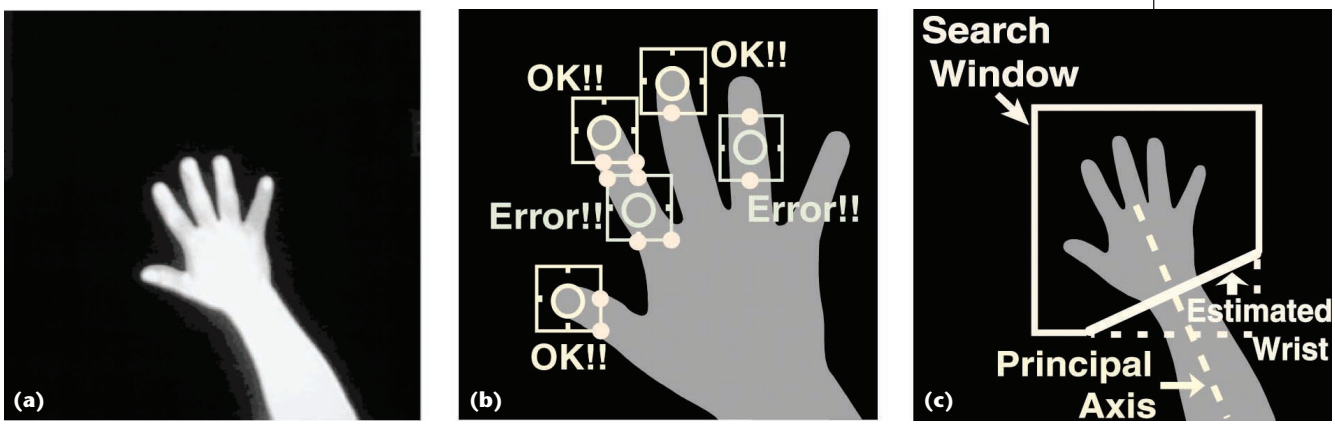
We determine a search window based on arm orientation, which is estimated as the extracted arm region's principal axis from the image moments up to the second order. We then set a fixed-size search window corresponding to the user's hand size so that it includes a hand part of the arm region based on the arm's orientation. The approximate distance from the infrared camera to a user's hand should determine the search



1 EnhancedDesk, an augmented desk interface system, applies fingertip tracking and gesture recognition to let users manipulate physical and virtual objects.



2 EnhancedDesk's two-handed drawing system.



3 Fingertip detection.

Related Work

Augmented reality systems can use a tracked hand's or fingertip's position as input for direct manipulation. For instance, some researchers have used their tracking techniques for drawing or for 3D graphic object manipulation.[1-4]

Many researchers have studied and used glove-based devices to measure hand location and shape, especially for virtual reality. In general, glove-based devices measure hand postures and locations with high accuracy and speed, but they aren't suitable for some applications because the cables connected to them restrict hand motion.

This has led to research on and adoption of computer vision techniques. One approach uses markers attached to a user's hands or fingertips to facilitate their detection. While markers help in more reliably detecting hands and fingers, they present obstacles to natural interaction similar to glove-based devices. Another approach extracts image regions corresponding to human skin by either color segmentation or background image subtraction. Because human skin isn't uniformly colored and changes significantly under different lighting conditions, such methods often produce unreliable segmentation of human skin regions. Methods based on background image subtraction also prove unreliable when applied to images with a complex background.

After a system identifies image regions in input images, it can analyze the regions to estimate hand posture. Researchers have developed several techniques to estimate pointing directions of one or multiple fingertips based on 2D hand or fingertip geometrical features.[1,2] Another approach used in hand gesture analysis uses a 3D human hand model. To determine the model's posture, this approach matches the model to a hand image obtained by one or more cameras.[3,5-7] Using a 3D human hand model solves the problem of self-occlusion, but these methods don't work well for natural or intuitive interactions because they're too computationally expensive for real-time processing and require controlled environments with a relatively simple background.

Pavlovic et al. provide a comprehensive survey of hand tracking methods and gesture analysis algorithms.[8]

References

1. M. Fukumoto, Y. Suenaga, and K. Mase, "Finger-pointer: Pointing Interface by Image Processing," *Computer and Graphics*, vol. 18, no. 5, 1994, pp. 633-642.
2. J. Segan and S. Kumar, "Shadow Gestures: 3D Hand Pose Estimation Using a Single Camera," *Proc. IEEE Conf. Computer Vision and Pattern Recognition* (CVPR 99), IEEE Press, Piscataway, N.J., 1999, pp. 479-485.
3. A. Utsumi and J. Ohya, "Multiple-Hand-Gesture Tracking Using Multiple Cameras," *Proc. IEEE Conf. Computer Vision and Pattern Recognition* (CVPR 99), IEEE Press, Piscataway, N.J., 1999, pp. 473-478.
4. J. Crowley, F. Berard, and J. Coutaz, "Finger Tracking as an Input Device for Augmented Reality," *Proc. IEEE Int'l Workshop Automatic Face and Gesture Recognition* (FG 95), IEEE Press, Piscataway, N.J., 1995, pp. 195-200.
5. J. Rehg and T. Kanade, "Model-Based Tracking of Self-Occluding Articulated Objects," *Proc. IEEE Int'l Conf. Computer Vision* (ICCV 95), IEEE Press, Piscataway, N.J., 1995, pp. 612-617.
6. N. Shimada et al., "Hand Gesture Estimation and Model Refinement Using Monocular Camera-Ambiguity Limitation by Inequality Constraints," *Proc. 3rd IEEE Int'l Conf. Automatic Face and Gesture Recognition* (FG 98), IEEE Press, Piscataway, N.J., 1998, pp. 268-273.
7. Y. Wu, J. Lin, and T. Huang, "Capturing Natural Hand Articulation," *Proc. IEEE Int'l Conf. Computer Vision* (ICCV 01), vol. 2, IEEE Press, Piscataway, N.J., 2001, pp. 426-432.
8. V. Pavlovic, R. Sharma, and T. Huang, "Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, July 1997, pp. 677-695.

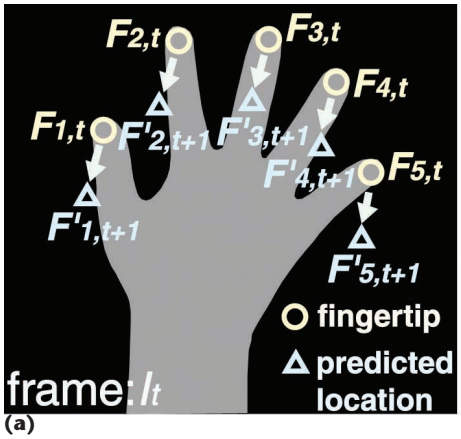
window's size. However, we found that a fixed-size search window works reliably because the distance from the infrared camera to a user's hand on the augmented desk interface system remains relatively constant.

We then search for fingertips within the new window. A cylinder with a hemispherical cap approximates finger shape, and the projected finger shape in an input image appears to be a rectangle with a semicircle at its tip, so we can search for a fingertip based on its geometrical features. Our method uses normalized correlation with a template of a properly sized circle corresponding to a user's fingertip size.

Although a semicircle reasonably approximates projected fingertip shape, we must consider false detection from the template matching and must also find a sufficiently large number of candidates. Our current implementation selects 20 candidates with the highest matching scores inside each search window, a sample we consider large enough to include all true fingertips.

Once we've selected the fingertip candidates, we remove false candidates using two methods. We remove multiple matching around the fingertip's true location by suppressing neighbor candidates around a candidate with the highest matching score. We then remove matching that occurs in the finger's middle by examining surrounding pixels around a matched template's center. If multiple diagonal pixels lie inside the hand region, we consider the candidate not part of a fingertip and therefore discard it (Figure 3b).

Finding a palm's center. In our method, the center of a user's hand is given as the point whose distance to the closest region boundary is the maximum. This makes the hand's center insensitive to changes such as opening and closing of the hand. We compute this location by a morphological erosion operation of an extracted hand region. First, we obtain a rough shape of the user's palm by cutting out the hand region at the estimated wrist. We assume the wrist's location is at the predetermined distance from the top of the search window and perpendicular to the hand region's principal direction (Figure 3c).



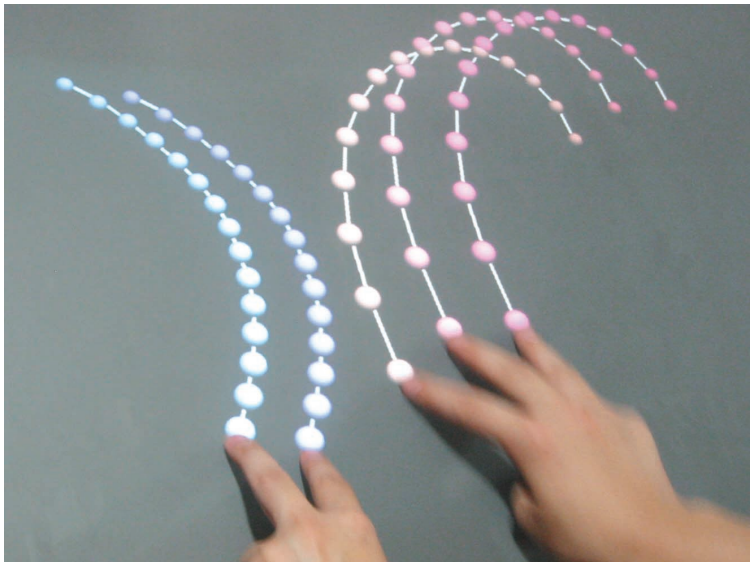
4 Taking fingertip correspondences: (a) detecting fingertips and (b) comparing detected and predicted fingertip locations to determine trajectories.

We then apply a morphological erosion operator to the obtained shape until the region becomes smaller than a predetermined threshold value. This yields a small region at the palm's center. Finally, the center of the hand region is given as the resulting region's center of mass.

Measuring fingertip trajectories

We obtain multiple fingertip trajectories by taking correspondences of detected fingertips between successive image frames.

Determining trajectories. Suppose that we detect n_t fingertips in the tth image frame I_t. We refer to these n_t fingertips' locations as $F_{i,t}(i = 1,2, \ldots, n_t)$ as Figure 4a shows. First, we predict the locations $F'_{i,t+1}$ of fingertips in the next frame I_{t+1}. Then we compare the locations $F_{j,t+1}$ ($j = 1,2, \ldots, n_{t+1}$) of n_{t+1} fingertips detected in the $t + 1$th image frame I_{t+1} with the predicted location $F'_{i,t+1}$ (Figure 4b). Finding the best combination among these two sets of fingertips lets us reliably determine multiple fingertip trajectories in real time (Figure 5).



5 Measuring fingertip trajectories.

Predicting fingertip locations. We use the Kalman filter to predict fingertip locations in one image frame based on their locations detected in the previous frame. We apply this process separately for each fingertip.

First, we measure each fingertip's location and velocity in each image frame. Hence we define the state vector as $\mathbf{x_t}$

$$\mathbf{x_t} = \left(x(t), y(t), v_x(t), v_y(t)\right)^{\mathrm{T}} \tag{1}$$

where $x(t)$, $y(t)$, $v_x(t)$, $v_y(t)$ shows the location of fingertip $(x(t), y(t))$ and the velocity of fingertip $(v_x(t), v_y(t))$ in tth image frame. We define the observation vector $\mathbf{y_t}$ to represent the location of the fingertip detected in the tth frame. The state vector $\mathbf{x_t}$ and observation vector $\mathbf{y_t}$ are related as the following basic system equation:

$$\mathbf{x_{t+1}} = \boldsymbol{F}\boldsymbol{x_t} + \boldsymbol{G}\boldsymbol{w_t} \tag{2}$$

$$\mathbf{y_t} = \boldsymbol{H}x_t + v_t \tag{3}$$

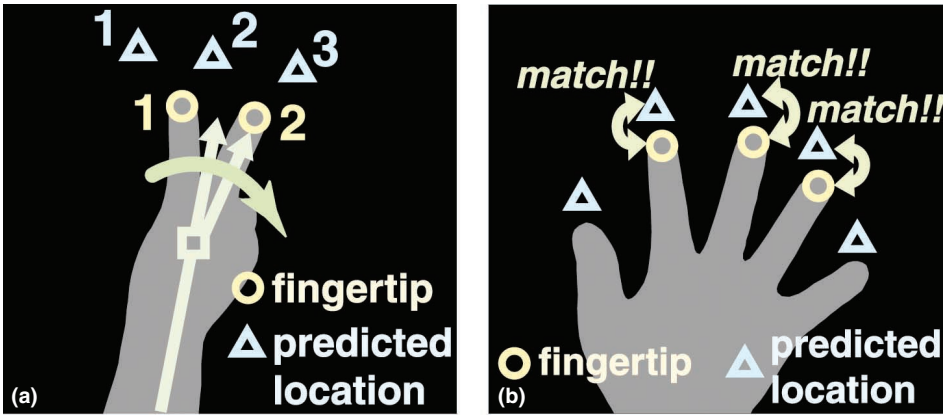
where $\boldsymbol{F}$ is the state transition matrix, $\boldsymbol{G}$ is the driving matrix, $\boldsymbol{H}$ is the observation matrix, w_t is system noise added to the velocity of the state vector $\mathbf{x_t}$, and v_t is the observation noise—that is, error between real and detected location.

Here we assume approximately uniform straight motion for each fingertip between two successive image frames because the frame interval ΔT is short. Then, $\boldsymbol{F}$, $\boldsymbol{G}$, and $\boldsymbol{H}$ are given as follows:

$$\boldsymbol{F} = \begin{bmatrix} 1 & 0 & \Delta T & 0 \\ 0 & 1 & 0 & \Delta T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

$$\boldsymbol{G} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{\mathrm{T}} \tag{5}$$

$$\boldsymbol{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{6}$$



6 Correspondences of detected and predicted fingertips: (a) fingertip order and (b) incorrect thumb and finger detection.

The (x, y) coordinates of the state vector $\mathbf{x_t}$ coincide with those of the observation vector $\mathbf{y_t}$ defined with respect to the image coordinate system. This is for simplicity of discussion without loss of generality. The observation matrix $\boldsymbol{H}$ should be in an appropriate form, depending on the transformation between the world coordinate system defined in the work space—for example, a desktop of our augmented desk interface system—and the image coordinate system.

Also, we assume that both the system noise $\mathbf{w_t}$ and the observation noise $\mathbf{v_t}$ are constant Gaussian noise with a zero mean. Thus the covariance matrix for $\mathbf{w_t}$ and $\mathbf{v_t}$ becomes $\sigma_w^2 \boldsymbol{I}_{2\times2}$ and $\sigma_v^2 \boldsymbol{I}_{2\times2}$ respectively, where $\boldsymbol{I}_{2\times2}$ represents a 2×2 identity matrix. This is a rather coarse approximation, and those two noise components should be estimated for each image frame based on some clue such as a matching score for normalized correlation for template matching. We plan to study this in the future.

Finally, we formulate a Kalman filter as

$$\boldsymbol{K}_t = \tilde{\boldsymbol{P}}_t \boldsymbol{H}^{\mathrm{T}} \left(\boldsymbol{I}_{2\times2} + \boldsymbol{H}\tilde{\boldsymbol{P}}_t \boldsymbol{H}^{\mathrm{T}} \right)^{-1} \tag{7}$$

$$\tilde{\mathbf{x}}_{\mathbf{t}+1} = \boldsymbol{F}\left\{ \tilde{\mathbf{x}}_{\mathbf{t}} + \boldsymbol{K}_t \left(\mathbf{y_t} - \boldsymbol{H}\tilde{\mathbf{x}}_{\mathbf{t}} \right) \right\} \tag{8}$$

$$\tilde{\boldsymbol{P}}_{t+1} = \boldsymbol{F}\left(\tilde{\boldsymbol{P}}_t - \boldsymbol{K}_t \boldsymbol{H}\tilde{\boldsymbol{P}}_t \right)\boldsymbol{F}^{\mathrm{T}} + \frac{\sigma_w^2}{\sigma_v^2}\Lambda \tag{9}$$

where $\tilde{\mathbf{x}}_{\mathbf{t}}$ equals $\hat{\mathbf{x}}_{\mathbf{t}|t-1}$, the estimated value of $\mathbf{x_t}$ from $y_0, \ldots, y_{t-1}$, $\tilde{\boldsymbol{P}}_t$ equals $\hat{\Sigma}_{\boldsymbol{t}|\boldsymbol{t}-1}/\sigma_v^2$, $\hat{\Sigma}_{\boldsymbol{t}|\boldsymbol{t}-1}$ represents the covariance matrix of estimation error of $\hat{\mathbf{x}}_{\mathbf{t}|\mathbf{t}-1}$, $\boldsymbol{K}_t$ is Kalman gain, and Λ equals $\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}$.

Then the predicted location of the fingertip in the $t+1$th image frame is given as $(x(t+1), y(t+1))$ of $\tilde{\mathbf{x}}_{\mathbf{t}+1}$. If we need a predicted location after more than one image frame, we can calculate the predicted location as follows:

$$\hat{x}_{t+m|t} = \boldsymbol{F}^m \left\{ \tilde{x}_t + \boldsymbol{K}_t \left(\mathbf{y_t} - \boldsymbol{H}\tilde{x}_t \right) \right\} \tag{10}$$

$$\hat{\boldsymbol{P}}_{t+m|t} = \boldsymbol{F}^m \left(\tilde{\boldsymbol{P}}_t - \boldsymbol{K}_t \boldsymbol{H}\tilde{\boldsymbol{P}}_t \right)\left(\boldsymbol{F}^{\mathrm{T}} \right)^m + \frac{\sigma_w^2}{\sigma_v^2} \sum_{l=0}^{m-1} \boldsymbol{F}^l \Lambda \left(\boldsymbol{F}^{\mathrm{T}} \right)^l \tag{11}$$

where $\hat{\mathbf{x}}_{\mathbf{t}+m|t}$ is the estimated value of $\mathbf{x}_{\mathbf{t}+m}$ from $\boldsymbol{y_0}, \ldots, \boldsymbol{y_t}$, $\hat{\boldsymbol{P}}_{t+m|t}$ equals $\hat{\Sigma}_{\boldsymbol{t}+\boldsymbol{m}|\boldsymbol{t}}/\sigma_v^2$, and $\hat{\Sigma}_{\boldsymbol{t}+\boldsymbol{m}|\boldsymbol{t}}$ represents the covariance matrix of estimation error of $\hat{\mathbf{x}}_{\mathbf{t}+m|t}$.

Fingertip correspondences between successive frames. For each image frame, we detect fingertips as described earlier and examine correspondences between the detected fingertips' locations and the predicted fingertip locations from Equation 8 or 10.

More precisely, we compute the sum of the square of distances between a detected fingertip and a predicted fingertip for all possible combinations and consider the combination with the least sum to be the most reasonable.

To avoid a high computational cost for examining all possible combinations, we reduce the number of combinations by considering the clockwise (or counterclockwise) fingertip order around the hand's center (Figure 6a). In other words, we assume the fingertip order in input images doesn't change. For instance, in Figure 6a, we consider only three combinations:

- O1–Δ1 and O2–Δ2
- O1–Δ1 and O2–Δ3
- O1–Δ2 and O2–Δ3

This reduces the maximum possible combinations from ${}_5P_5$ to ${}_5C_5$.

Occasionally, the system doesn't detect one or more fingertips in an input image frame. Figure 6b illustrates an example where an error prevents detection of the thumb and little finger. To improve our method's reliability for tracking multiple fingertips, we use a missing fingertip's predicted location to continue tracking it. If we find no fingertip corresponding to the predicted one, we examine the element (1, 1) of the covariance matrix $\tilde{\boldsymbol{P}}_{t+1}$ in Equation 9 for the predicted fingertip. This element represents the ambiguity of the predicted fingertip's location—if it's smaller than a predetermined ambiguity threshold, we consider the fingertip to be undetected because of an image frame error. We then use the fingertip's predicted location as its true location and continue tracking it.

If the element (1, 1) of the covariance matrix exceeds a predetermined threshold, we determine that the fingertip prediction is unreliable and terminate its track-

ing. Our current implementation fixes an experimentally chosen ambiguity threshold.

If we detect more fingertips than predicted, we start tracking a fingertip that doesn't correspond to any of the predictions. We treat its trajectory as that of a new fingertip after the predicted fingertip location's ambiguity falls below a predetermined threshold.

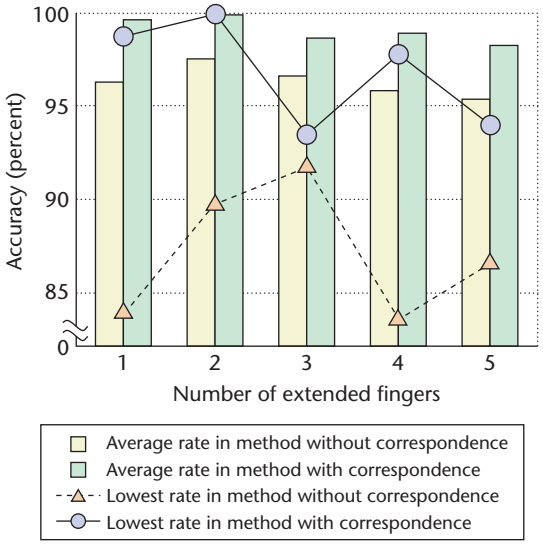
Evaluating the tracking method

To test our method, we experimentally evaluated the reliability improvement by considering fingertip correspondences between successive image frames using seven test subjects. Our tracking system consists of a Linux-based PC with Intel Pentium III 500-MHz and Hitachi IP5000 image processing board, and a Nikon Laird-S270 infrared camera.

We asked test subjects to move their hands naturally on our augmented desk interface system while keeping the number of extended fingers constant in each trial. In the first trial, subjects moved their hands with one extended finger for 30 seconds, then extended two, three, four, and finally five fingers. Each trial lasted 30 seconds and produced about 900 image frames. To ensure fair comparison, we first recorded the infrared camera output using a video recorder, then applied our method to the recorded video.

We compared our tracking method's reliability with and without correspondences between successive image frames. Figure 7 shows the results, with bar charts indicating the average rate that the number of tracked fingertips was correct and line charts indicating the lowest rate among seven test subjects.

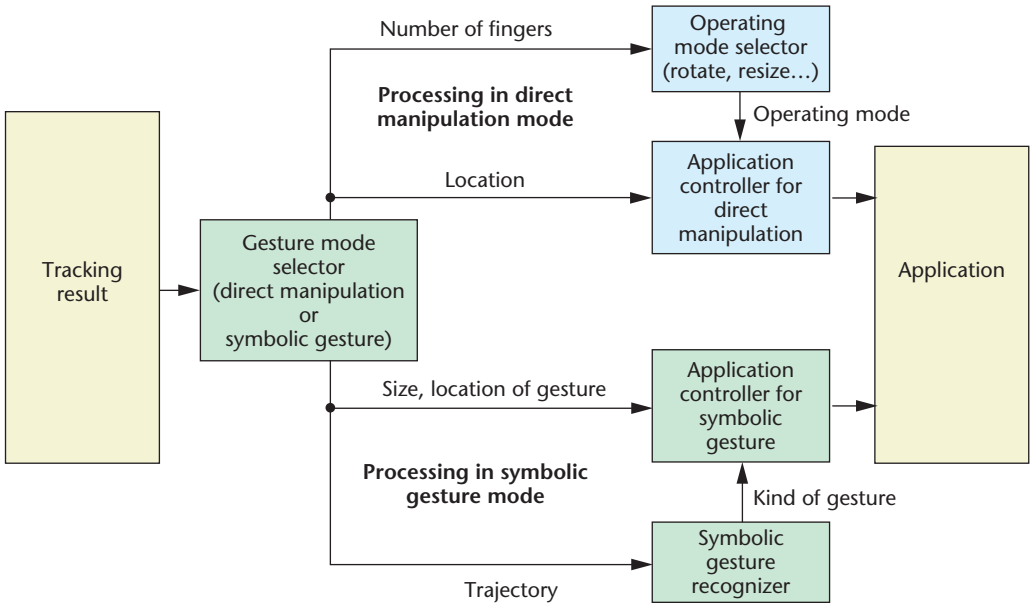
As Figure 7 shows, tracking reliability improves significantly when we account for fingertip correspondences between image frames. In particular, tracking accuracy approaches 100 percent for one or two fingers, and the lowest rate also improves. Our method reliably tracks multiple fingertips and could prove useful in real-time human–computer interaction applications.



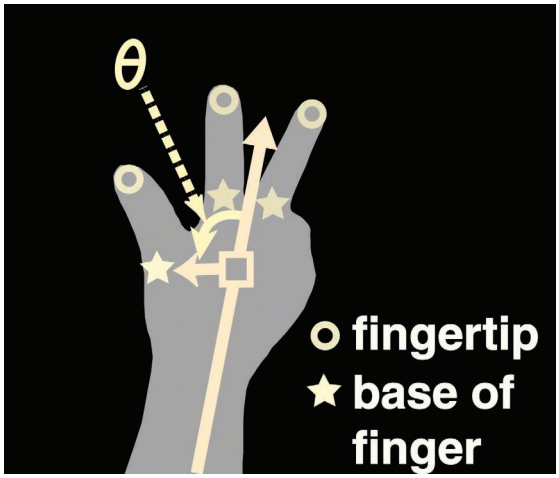
7 Finger tracking evaluation.

Gesture recognition

Our tracking method also works well for gesture recognition and lets us achieve interactions based on symbolic gestures while we perform direct manipulation with our hands and fingers using the mechanism shown in Figure 8. First, our system determines from measured fingertip trajectories whether a user's hand motions represent direct manipulation or symbolic gestures. For direct manipulation, it then selects operating modes such as rotate, move, or resize based on the distance between two fingertips or the number of extended fingers, and controls the selected modes' parameters. For symbolic gestures, the system recognizes gesture types using a



8 Interaction based on direct manipulation and symbolic gestures.



9 Definition of the angle θ between finger direction and arm orientation, for thumb detection.

symbolic gesture recognizer in addition to recognizing gesture locations and sizes based on trajectories.

To distinguish symbolic gestures from direct manipulation, our system locates a thumb in the measured trajectories. As described earlier, we regard gestures with an extended thumb as direct manipulation and those with a bent thumb as symbolic gestures, and use the HMM to recognize the segmented symbolic gestures. Our gesture recognition system should prove useful for augmented desk interface applications such as the drawing tool shown in Figure 2.

Detecting a thumb

Our method for distinguishing a thumb from the other tracked fingertips uses the angle θ between finger direction—the direction from the hand's center to the finger's base—and arm orientation (Figure 9). We use the finger's base because it's more stable than the tip even if the finger moves.

In the initialization stage, we define the thumb's standard angle θ_T and that of the forefinger θ_F ($\theta_T > \theta_F$). First, we apply the morphological process and the image subtraction to a binarized hand image to extract finger regions. We regard the end of the extracted finger opposite the fingertip as the base of the finger and calculate θ.

Here we define θ_k as θ in the kth frame from the finger trajectory's origin, and the current frame is the Nth frame from the origin. Then, the score s_T, which represents a thumb's likelihood, is given as follows:

$$s'_T(k) = \begin{cases} 1.0 & \text{if } \theta_k > \theta_T \\ \dfrac{\theta_k - \theta_F}{\theta_T - \theta_F} & \text{if } \theta_F \le \theta_k \le \theta_T \\ 0.0 & \text{if } \theta_k < \theta_F \end{cases} \quad (12)$$

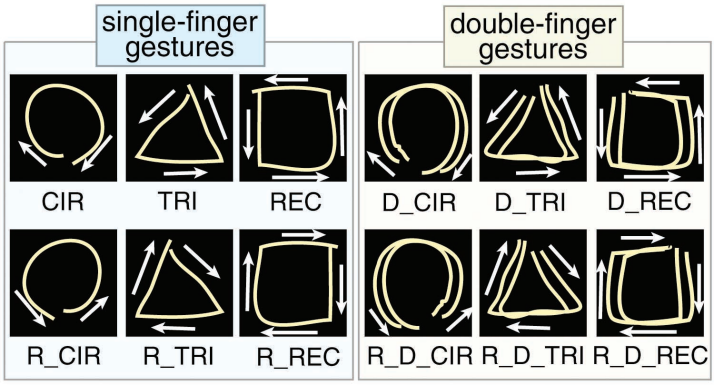
$$s_T = \frac{\sum_{k=1}^{N} s'_T(k)}{N} \quad (13)$$

If s_T exceeds 0.5, we regard the finger as a thumb.

To evaluate this method's reliability, we performed three kinds of tests mimicking actual desktop work: drawing with only a thumb, picking up with a thumb and forefinger, and drawing with only a forefinger. Table 1 shows the results, demonstrating that the method reliably distinguishes the thumb from other hand parts.

Symbolic gesture recognition

Like other recognition techniques,[2,3] our symbolic gesture recognition system uses HMM. The input to HMM consists of two components for recognizing multiple fingertip trajectories: the number of tracked fingertips and a discrete code from 1 to 16 that represents the direction of the tracked fingertips' average motions. It's unlikely that we would move each of our fingers independently unless we consciously tried to do so. Thus, we decided to use the direction of multiple fingertips' average motions instead of each fingertip's direction. We used code 17 to represent no motion.



10 Symbolic gesture examples.

We tested our recognition system using 12 kinds of hand gestures with the fingertip trajectories shown in Figure 10. As a training data set for each gesture, we used 80 hand gestures made by a single person to initialize HMM. Six other people also participated in testing. For each trial, a test subject made one of 12 gestures 20 times at arbitrary locations and with arbitrary sizes. Table 2 shows this experiment's results,

Table 1. Evaluating thumb distinction.

Task	Drawing with Thumb Only	Picking Up with Thumb and Forefinger	Drawing with Forefinger Only
Average (percent)	98.2	99.4	98.3
Standard deviation (percent)	3.6	0.8	4.6

indicating average accuracy and standard deviation for single-finger and double-finger gestures.

Our system offers reliable, near-perfect recognition of single-finger gestures and high accuracy for double-finger gestures. Our gesture recognition system proves suitable for natural interactions using hand gestures.

Table 2. Evaluating gesture recognition.

Gesture Type	Single-Finger	Double-Finger
Average (percent)	99.2	97.5
Standard deviation (percent)	0.5	1.8

Future work

We plan to improve our tracking method's reliability by incorporating additional sensors. Although using an infrared camera had some advantages, it didn't work well on cold hands. We'll solve this problem by using a color camera in addition to the infrared camera.

We also plan to extend our system for 3D tracking. Currently, our tracking method is limited to 2D motion on a desktop. Although this is enough for our augmented desk interface system, other application types require interaction based on 3D hand and finger motion. We'll therefore investigate a practical 3D hand and finger tracking technique using multiple cameras. ■

References

1. L. Rabiner and B. Juang, "An Introduction to Hidden Markov Models," *IEEE Acoustic Signal and Speech Processing (ASSP)*, vol. 3, no. 1, Jan. 1986, pp. 4-16.
2. T. Starner and A. Pentland, "Visual Recognition of American Sign Language Using Hidden Markov Models," *Proc. IEEE Int'l Workshop Automatic Face and Gesture Recognition* (FG 95), IEEE Press, Piscataway, N.J., 1995, pp. 189-194.
3. J. Martin and J. Durand, "Automatic Handwriting Gestures Recognition Using Hidden Markov Models," *Proc. 4th IEEE Int'l Conf. Automatic Face and Gesture Recognition* (FG 2000), IEEE Press, Piscataway, N.J., 2000, pp. 403-409.
4. J. Underkoffler and H. Ishii, "Illuminating Light: An Optical Design Tool with a Luminous-Tangible Interface," *Proc. ACM Conf. Human Factors and Computing Systems* (CHI 98), ACM Press, New York, 1998, pp. 542-549.
5. J. Rekimoto and M. Saito, "Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments," *Proc. ACM Conf. Human Factors and Computing Systems* (CHI 99), ACM Press, New York, 1999, pp. 378-385.
6. P. Wellner, "Interacting with Paper on the DigitalDesk," *Comm. ACM*, vol. 36, no. 7, July 1993, pp. 87-96.
7. H. Koike et al., "Interactive Textbook and Interactive Venn Diagram: Natural and Intuitive Interface on Augmented Desk System," *Proc. ACM Conf. Human Factors and Computing Systems* (CHI 2000), ACM Press, New York, 2000, pp. 121-128.
8. X. Chen et al., "Two-Handed Drawing on Augmented Desk System," *Proc. Int'l Working Conf. Advanced Visual Interfaces* (AVI 2002), ACM Press, New York, 2002, pp. 219-222.
9. Y. Sato, Y. Kobayashi, and H. Koike, "Fast Tracking of Hands and Fingertips in Infrared Images for Augmented Desk Interface," *Proc. 4th IEEE Int'l Conf. Automatic Face and Gesture Recognition* (FG 2000), IEEE Press, Piscataway, N.J., 2000, pp. 462-467.

Kenji Oka *is a PhD candidate at the University of Tokyo Graduate School of Information Science and Technology. His research interests include human–computer interaction and computer vision, particularly perceptual user interfaces and human behavior understanding. He received BS and MS degrees in information and communication engineering from the University of Tokyo.*

Yoichi Sato *is an associate professor at the University of Tokyo Institute of Industrial Science. His primary research interests are in computer vision (physics-based vision, image-based modeling), human–computer interaction (perceptual user interfaces), and augmented reality. He received a BS in mechanical engineering from the University of Tokyo and an MS and PhD in robotics from the School of Computer Science, Carnegie Mellon University. He is a member of IEEE and ACM.*

Hideki Koike *is an associate professor at the Graduate School of Information Systems, University of Electro-Communications, Tokyo. His research interests include information visualization and vision-based human–computer interaction for perceptual user interfaces. He received a BS in mechanical engineering and an MS and Dr.Eng. in information engineering from the University of Tokyo. He is a member of IEEE and ACM.*

Readers may contact Kenji Oka at the Institute of Industrial Science, University of Tokyo, 4-6-1 Komaba Meguro-ku, Tokyo 153-8505, Japan, email oka@iis.u-tokyo.ac.jp.

For further information on this or any other computing topic, please visit our Digital Library at http://computer.org/publications/dlib.