Vision-Based Finger Action Recognition by Angle Detection and Contour Analysis

Daeho Lee and SeungGwan Lee

In this paper, we present a novel vision-based method of recognizing finger actions for use in electronic appliance interfaces. Human skin is first detected by color and consecutive motion information. Then, fingertips are detected by a novel scale-invariant angle detection based on a variable *k*-cosine. Fingertip tracking is implemented by detected region-based tracking. By analyzing the contour of the tracked fingertip, fingertip parameters, such as position, thickness, and direction, are calculated. Finger actions, such as moving, clicking, and pointing, are recognized by analyzing these fingertip parameters. Experimental results show that the proposed angle detection can correctly detect fingertips, and that the recognized actions can be used for the interface with electronic appliances.

Keywords: Fingertip detection, fingertip tracking, scale-invariant angle detection, human-machine interaction, machine vision.

I. Introduction

In vision-based augmented reality systems, a user's shape and actions are essentially detected from an image sequence acquired by a camera. Although the face, hand, finger, or entire body may be defined as the user's shape, hands and fingers are more appropriate for human-machine interaction because they and their associated gestures can be easily detected. For accuracy and precision, data gloves or special markers may be used to detect user actions [1]-[6]; however, these are not convenient for personal electronic appliances since they must be worn. Therefore, methods that can detect and track the movements of bare hands facilitate easy and cost-effective use [7]-[9].

To detect fingers or fingertips, hand regions must first be detected [9]-[15]. Skin color [9]-[11], edge [12], and temperature [13] have been used for the detection. Hand edges can easily be extracted only from uniform background images, and methods based on skin color may fail under complicated or dynamic lighting. To solve these problems, gradient, 3D, and motion information was used on extracted skin regions in [14]-[18]. The use of temperature is not very practical, owing to the expense of infrared cameras. Skeleton [13], dominant points [10], and polar coordinates have been used to detect fingers or fingertips from the extracted hand region; however, these methods encounter difficulty due to noise, scale, and hand direction. Methods that detect hand regions from grayscale images have been proposed [19], [20]; however, only uniform backgrounds were considered. Motion features may be used to track fingers [21], [22], but precise finger motions may not easily be tracked by region motion.

Real-time fingertip detection and tracking can be applied in electronic appliance interfaces such as remote control systems

Manuscript received June 8, 2010; revised Dec. 17, 2010; accepted Jan. 19, 2011.

Daeho Lee (phone: +82 31 201 2289, email: nize@khu.ac.kr) and SeungGwan Lee (email: leesg@khu.ac.kr) are with the Humanitas College, Kyung Hee University, Yongin, Gyeonggido, Rep. of Korea.

doi:10.4218/etrij.11.0110.0313

[10] and input devices [21]-[24]. The utility and convenience of such systems is enhanced by the ability to recognize actions, such as moving, clicking, and pointing. In [24], it was shown that a variety of interface actions can be recognized in mobile devices by 3D fingertip tracking using only one camera at a rate of 75 frames per second (fps).

This paper is aimed at developing a novel method which can recognize finger actions for electronic appliance interfaces and thereby detect and track fingertips with high performance and low-cost. The proposed method can detect fingertips of different hand sizes by scale-invariant angle detection and recognize interface actions by contour analysis of detected fingertips. The processing speed is sufficiently fast to make real-time operation practicable.

II. Overview of Proposed Method

The overall scheme of the proposed fingertip action recognition method is shown in Fig. 1. In each frame, skin and hand regions are first detected, and then fingertips are detected by variable *k*-cosines which can detect angles with scale invariants. Detected fingertips are tracked. Fingertip actions, such as moving, clicking, and pointing, are finally recognized using fingertip parameters (position, thickness, and direction) calculated from contour information, and these actions are transmitted as predefined human commands.

III. Hand Region Detection

Human skin is easily detected by color information. Determining a color range for skin detection is, however, very difficult because color intensities are dependent on the equipment and the environments. To solve this difficulty, we detect skin regions by background subtraction [25] and redness [26]. The skin regions are detected by

$$S_t(x,y) = \begin{cases} 1, & \text{if } \min\left(R_t(x,y)-G_t(x,y), R_t(x,y)-B_t(x,y)\right) > \tau_s \\ & \quad \wedge\ F_t(x,y), \\ 0, & \text{otherwise,} \end{cases} \quad (1)$$

where R_t, G_t, and B_t denote red, green, and blue components at frame t, respectively, τ_s is a threshold value, and $F_t(x, y)$ is the foreground region by subtracting the input image to the background image. By selecting a low τ_s, we can detect skin regions, including bright and dark regions. In Fig. 2(b), eliminated skin background regions by background subtraction are shown with back regions.

To reduce the effects of noise in the S_t values, we apply a median filter and image labeling to detect isolated skin regions.

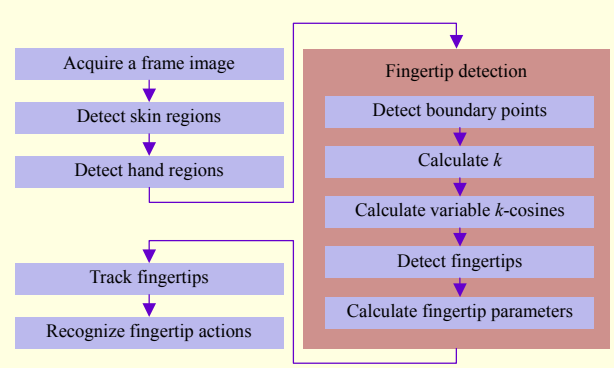
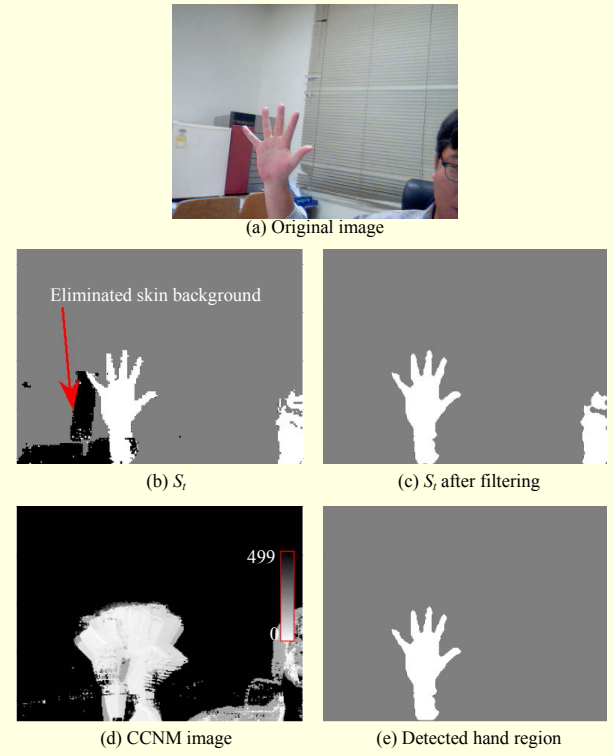


Fig. 1. Overall scheme of proposed finger action recognition.



(a) Original image
(b) S_t
(c) S_t after filtering
(d) CCNM image
(e) Detected hand region

Fig. 2. Hand region detection.

Figure 2(c) shows detected skin regions after filtering. However, three skin regions should be determined whether they are hand regions or not.

Since the hands being detected may be in motion, we use frame differences to determine hand regions. A simple frame difference, however, may be sensitive to motion. Thus, we use a consecutive count of non-movements (CCNM); this is defined by

$$C_t(x,y) = \begin{cases} 0, & \text{if } \left|I_t(x,y) - I_{t-1}(x,y)\right| > \tau_d, \\ C_{t-1}(x,y)+1, & \text{otherwise,} \end{cases} \quad (2)$$

where I_t denotes the intensity of the image at time t, and τ_d is a threshold value. Pixels associated with movement are characterized by small values of C_t. Therefore, hand regions are found by selecting skin regions having a large number of pixels with sufficiently small values of C_t as shown in Figs. 2(d) and (e).

IV. Fingertip Detection and Tracking

Once a hand region is detected, fingertips may be detected by high values of curvature because open fingers are usually sharp. Curvature is defined by tangential differentiation of a curve function: the curvature of the function $y = f(x)$ is given by

$$\kappa = \frac{y''}{(1+(y')^2)^{3/2}}. \quad (3)$$

To use the curvature of (1) on a digital planar curve, a neighbor size (that is, a differential of an analogue curve) for differentiation is first defined. If a digital planar curve formed with N points is defined by

$$C = \{p_0, p_1, ..., p_{i-1}, p_i = (x_i, y_i), p_{i+1}, ..., p_{N-1}\}, \quad (4)$$

the differential coefficient at p_i of C is usually calculated by $(y_i - y_{i-k})/(x_i - x_{i-k})$, where $k \geq 1$. The curvature calculated by this coefficient, however, is too sensitive to the size of k. Selecting k is therefore very difficult since we cannot know the shape of the target object or the noise level in advance.

In general, k-cosines [27], [28] measure the included angle at a curve point as curvature, so they are well suited to the detection of dominant points. The k-vectors at p_i are defined as

$$\begin{aligned} \mathbf{a}_{ik} &= (x_i - x_{i-k},\ y_i - y_{i-k}), \\ \mathbf{b}_{ik} &= (x_i - x_{i+k},\ y_i - y_{i+k}). \end{aligned} \quad (5)$$

The k-cosine, which is the cosine of the angle between $\mathbf{a}_{ik}$ and $\mathbf{b}_{ik}$, is given by

$$\cos\theta_{ik} = \frac{\mathbf{a}_{ik} \cdot \mathbf{b}_{ik}}{|\mathbf{a}_{ik}||\mathbf{b}_{ik}|}. \quad (6)$$

Note that k-cosines are, like the curvatures calculated by (3), sensitive to the size of k. In Fig. 3, the k-cosines are also dependent on the k-value of $\mathbf{b}_{ik}$. When k is small, the direction of the k-vector is overly sensitive to noise (for example, $\mathbf{b}_{i1}$, $\mathbf{b}_{i2}$, and $\mathbf{b}_{i3}$ in Fig. 3). In Fig. 3, $\mathbf{b}_{i5}$ is the best choice for $\mathbf{b}_{ik}$, since the $\mathbf{b}_{ik}$ for $6 \leq k \leq 11$ does not correctly represent the curve boundary. For this selection of k-vector, we use the difference, that is, error, between the chord $\overline{p_i p_{i+k}}$ and the arc $\overset{\frown}{p_i p_{i+k}}$. We select the maximum k having the error less than the acceptable error. Then, p_{i+k}, where k is the selected value, is assigned as the end of the right k-vector. The left k-vector can

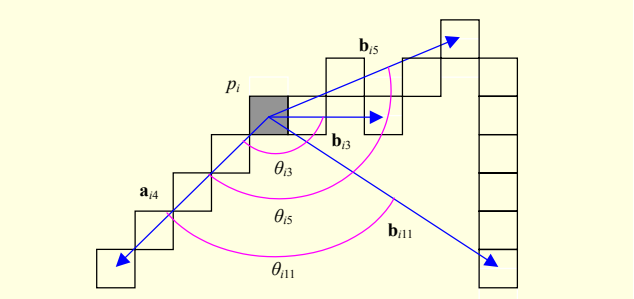


Fig. 3. Angle detection by different k-values.

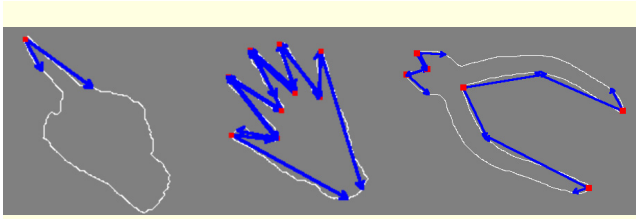


Fig. 4. Calculated variable k-vectors.

be also found in a similar manner. The error between the chord and the arc is calculated using the perpendicular distance by

$$\max_{t \in [1, k-1]} \sqrt{\frac{\left((x_{i+t} - x_i)(y_{i+k} - y_i) - (y_{i+t} - y_i)(x_{i+k} - x_i)\right)^2}{(x_i - x_{i+k})^2 + (y_i - y_{i+k})^2}}. \quad (7)$$

For noise suppression, the error is calculated by starting from the predefined minimum value.

The acceptable error may be a constant γ; however, we use $\gamma |t|$ in order to preserve scale invariance of the vector identified. Figure 4 shows examples of the identified k-vectors by the proposed method.

Convex points with acute angles represent fingertips. Thus, we detect convex contour points having high local maximum k-cosines; the convex point is determined by

$$d_i > (d_{i-s} + d_{i+s})/2, \quad (8)$$

where d_i denotes distance from the centroid of the hand region, and $s>0$. Figure 5 shows an example of fingertip detection. The detected fingertips and finger valleys are marked with red dots and cyan dots, respectively.

To recognize fingertip actions and track fingertips, we define fingertip parameters as shown in Table 1. The position of the detected fingertip at time t is $f(t)$, and $\mathbf{v}(t)$ and $w(t)$ are the direction vector pointed by the finger that includes the detected fingertip at time t and the thickness of this finger, respectively. The relative distance of a tracked fingertip from the camera is calculated by $1/w(t)$.

w and $\mathbf{v}$ are easily calculated as shown in Fig. 6, where n_1

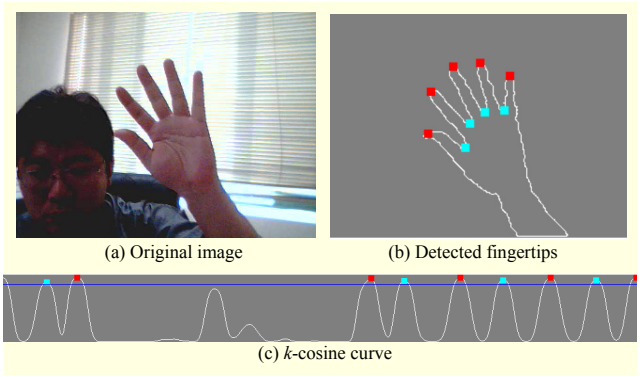


Fig. 5. Fingertip detection.

Table 1. Fingertip parameters.

Parameter	Description
$f = (f_x, f_y)$	Position of fingertip
w	Thickness of finger
$\mathbf{v} = (v_x, v_y)$	Direction vector pointed by finger

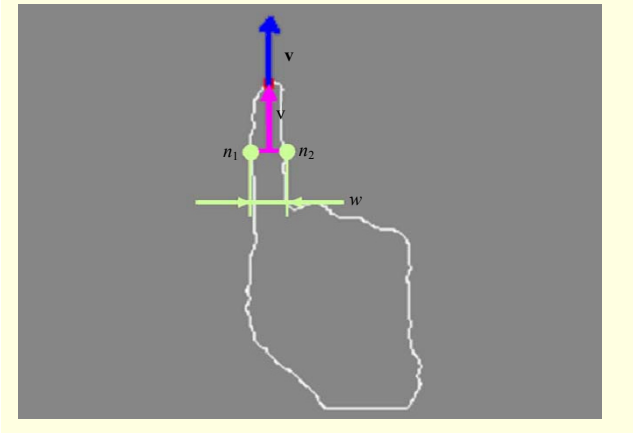


Fig. 6. Fingertip parameters.

and n_2 are the curve points located $\pm n$ neighbors from the detected fingertip. The distance between n_1 and n_2 is w, and $\mathbf{v}$ is calculated by

$$\mathbf{v} = (f_x, f_y) - (n_x, n_y), \tag{9}$$

where (n_x, n_y) is the center point between n_1 and n_2. Figure 7 shows examples of fingertip parameters for detected fingertips. The blue arrows point in the direction of $\mathbf{v}$, and their magnitude indicates w.

Because fingertips are detected in every frame, we implement region-based tracking, which is performed as follows. If the detected fingertip in the current frame cannot be associated with any corresponding tracked fingertips, the fingertip is initiated for new tracking. If there is no fingertip

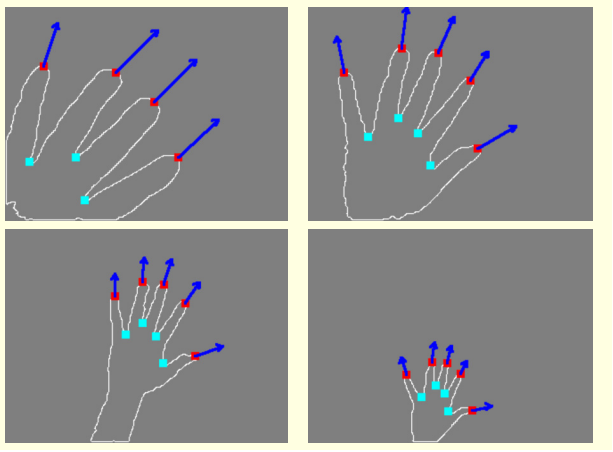


Fig. 7. Examples of fingertip parameters of detected fingertips.

associated with a tracked fingertip for n_t consecutive frames, tracking of the fingertip is terminated. The detected fingertip associated with each tracked fingertip is determined by a prediction filter based on the Kalman filter [29].

The prediction filter uses the fingertip parameters as the tracking states. Hence, fingertips that are closely tracked are obviously isolated, and any tracked fingertip which is not detected (in less than n_t consecutive frames) is tracked by the prediction.

V. Fingertip Action Detection

By the fingertip tracking, various fingertip actions, such as moving, clicking, and pointing, can be detected. The move action is determined simply by the f of the tracked fingertip.

We strictly define clicking actions to reduce false alarms. When a clicking action is forcefully performed, the index finger disappears for a moment, and then reappears within the same parameters; thus, this action is recognized by tracking information. When a tracked finger reappears, the clicking action is recognized by the following condition:

$$\begin{aligned}&\left(\tau_1 < t_p < \tau_2\right) \wedge \left(\max\left|f(t-i) - f(t)\right| < \tau_f\right) \\ &\qquad \wedge \left(\max\left|\mathbf{v}(t-i) - \mathbf{v}(t)\right| < \tau_v\right),\end{aligned} \tag{10}$$

where t_p is the duration of the time the finger disappears before reappearing, the range of i is some duration before disappearing, and τ_1, τ_2, τ_f, and τ_v are threshold values.

The pointing action can be detected by all fingertip parameters. When a fingertip points at something, none of parameters vary significantly for a moment since the finger does not move during a pointing action.

VI. Experimental Results

The proposed method was implemented in Visual C++, and

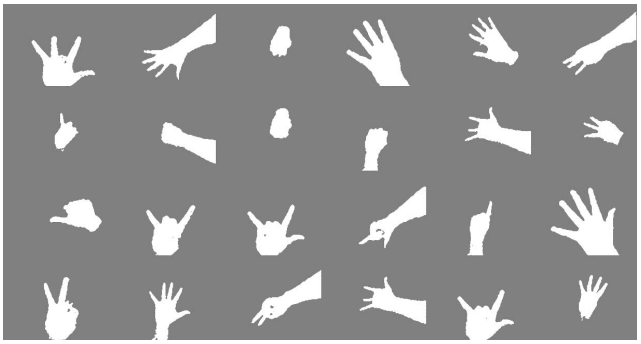


Fig. 8. Test data examples for fingertip detection.

Table 2. Detection result of fingertip counting.

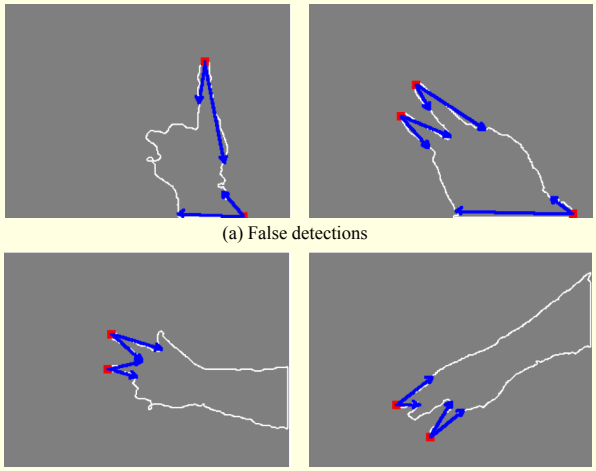
Number of fingertips	Correct/total (accuracy rates: %)	
	Ref. [10]	Proposed method
0	87/88 (98.86)	88/88 (100.00)
1	126/131 (96.18)	129/131 (98.47)
2	223/252 (88.49)	249/252 (98.81)
3	286/331 (86.40)	327/331 (98.79)
4	208/273 (76.19)	267/273 (97.80)
5	195/258 (75.58)	252/258 (97.67)
Total	1,125/1,333 (84.40)	1,312/1,333 (98.42)

tested on a Pentium PC (Core™ 2 Duo, 2.40 GHz). The test image sequences were grabbed from a Web camera (LifeCam VX-1000) with a resolution of 320×240.

There are many methods for detection of fingertips; however, the performance of these methods is often overestimated using their exclusive data for promotional reasons. Thus, we compared our method with the method proposed in [10] using a single set of test data. The method proposed in [10] used a *k*-cosine with fixed *k*-values to count fingertips. The test images were collected from a Web camera, and segmented by background subtraction and redness. In addition, the images include various gestures involving hands (for example, counting, finger gestures, and sign language), hand orientations, and hands with/without wrists as shown in Fig. 8.

The proposed method by scale-invariant angle detection is more robust than the previous method in [10] as shown in Table 2. The only false detections were observed at wrist boundaries as shown in Fig. 9(a). When fingers with short lengths were shown, the fingers were not detected (Fig. 9(b)) since we calculate the error (chord and arc) by starting from the predefined minimum value for noise suppression.

The detection result of the proposed method is more accurate than other fingertip detection using template matching [9], where the success ratio of fingertip detection was reported as 90.5% (complicated background and many fingers).



(a) False detections

(b) Misdetections

Fig. 9. False detections and misdetections.

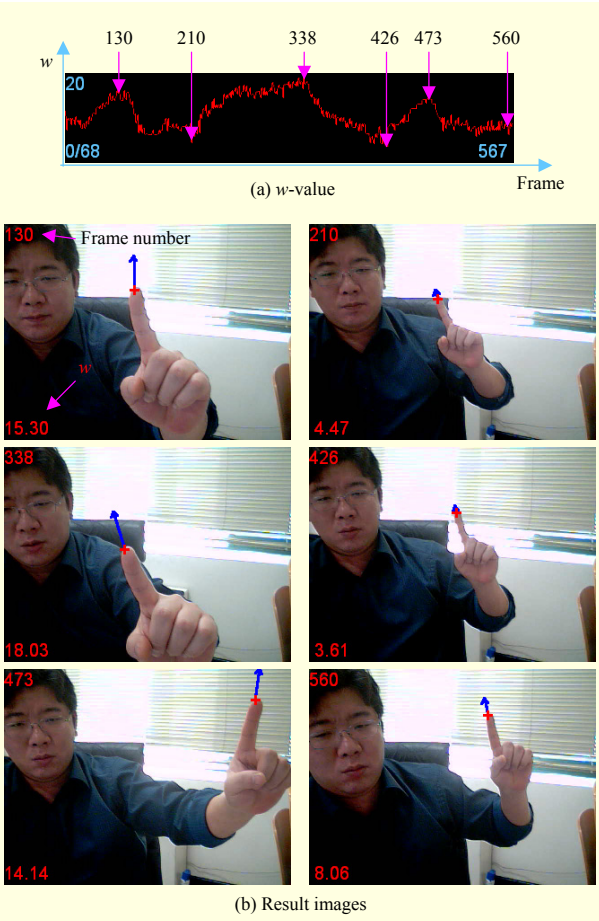


Fig. 10. Fingertip detection and tracking results.

Figure 10 shows fingertip detection and tracking results. Figure 10(a) shows the *w*-value corresponding to each frame. In Fig. 10(b), detected fingertips are marked with crosses, the blue arrows point in the direction of **v**, and the magnitude

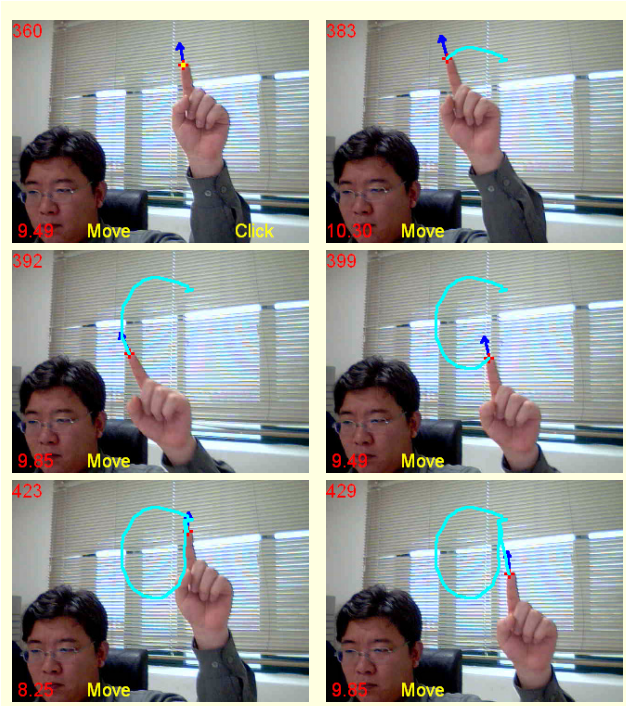


Fig. 11. Drawing example by fingertip tracking.

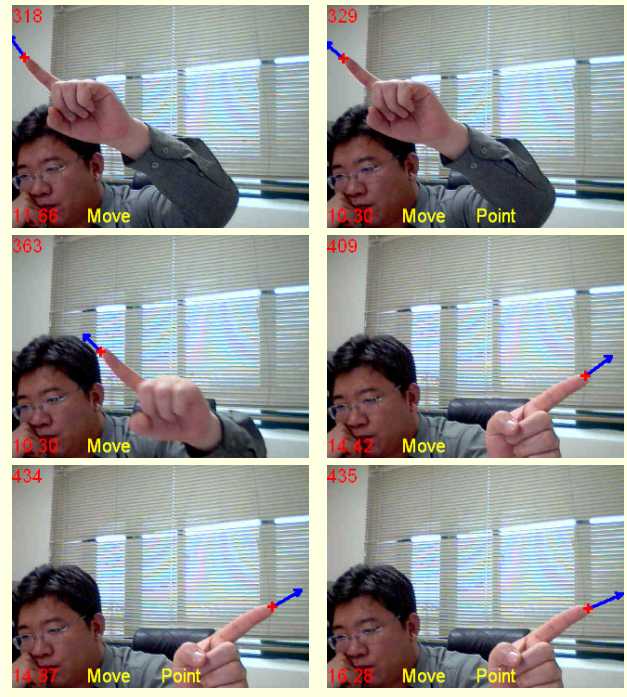


Fig. 12. Pointing examples by fingertip parameters.

indicates w. The top left numbers are frame numbers, and the bottom left values denote w. Because w was stably calculated, the relative value of $1/w$ can be used for the relative distance from the camera.

Fig. 13. Clicking example by fingertip parameters.

Figures 11, 12, and 13 show fingertip tracking and parameter determination for example tasks of drawing, pointing, and clicking, respectively. The drawing was reset by clicking at the 360th frame in Fig. 11, and an 'a' character was drawn by the index finger. Pointing actions were recognized at the 329th, 434th, and 435th frames in Fig. 12. A clicking was recognized at the 330th frame in Fig. 13, and the third menu was selected. The clicking action can be used for human-machine interaction.

The average processing time per frame is 16.65 ms, which includes preprocessing, fingertip detection and tracking, and recognition of fingertip actions. This processing time is fast enough for real-time interfaces.

Quantitative recognition results of interface actions are very sensitive to user experience since a more frequent user may be more adept. Our method, however, has advantages over other methods, including the ability to use low-cost cameras, robust fingertip detection regardless of hand scale, and detectability of a wide variety of interface actions.

VII. Conclusion

We presented a novel vision-based real-time method of recognizing fingertip actions for electronic appliance interfaces. Hand regions are first detected by redness. To eliminate false detections, skin-like regions associated with movement are selected. Scale-invariant angle detection based on k-cosines is used to find fingertips. This method yields robust and accurate performance regardless of hand scale. By analyzing the position and contour of the detected fingertip, various interface actions, such as moving, clicking, and pointing, are recognized. Since fingertips are correctly detected and tracked, fingertip parameters, which can measure position, direction, and distance, may be stably calculated. Therefore, the proposed method may lead to a new generation of interfaces.

References

[1] G. Hillebrand and K. Zuerl, "Finger Tracking for Virtual Agents," *LNCS*, vol. 4722, 2007, pp. 417-419.

[2] G. Hillebrand et al., "Inverse Kinematic Infrared Optical Finger Tracking," *Int. Conf. Humans and Computers*, 2006.

[3] P.G. Kry et al., "HandNavigator: Hands-On Interaction for Desktop Virtual Reality," *ACM Symp. Virtual Reality Software Technol.*, Oct. 2008, pp. 53-60.

[4] R.Y. Wang and J. Popovic, "Real-Time Hand-Tracking with a Color Glove," *Int. Conf. Computer Graphics Interactive Techn.*, 2009.

[5] S.H. Lee, J. Choi, and J. Park, "Interactive e-Learning System using Pattern Recognition and Augmented Reality," *IEEE Trans. Consumer Electron.*, vol. 55, no. 2, May, 2009, pp. 883-890.

[6] F. Duca, J. Fredriksson, and M. Fjeld, "Real-Time 3D Hand Interaction: Single Webcam Low-Cost Approach," *IEEE Workshop Trends Issues Tracking Virtual Environments*, 2007, pp. 1-5.

[7] J. Segen and S. Kumar, "Look Ma, No Mouse!" *Commun. ACM*, vol. 43, no. 7, July 2000, pp. 102-109.

[8] J. Segen and S. Kumar, "GestureVR: Vision-Based 3D Hand Interface for Spatial Interaction," *Proc. 6th ACM Int. Conf. Multimedia*, 1998, pp. 455-464.

[9] J.M. Kim and W.K. Lee, "Hand Shape Recognition Using Fingertips," *5th Int. Conf. Fuzzy Syst. Knowledge Discovery*, 2008, pp. 44-48.

[10] D. Lee and Y. Park, "Vision-Based Remote Control System by Motion Detection and Open Finger Counting," *IEEE Trans. Consumer Electron.*, vol. 55, no. 4, Nov. 2009, pp. 2308-2313.

[11] K. Oka, Y. Sato, and H. Koike, "Real-Time Fingertip Tracking and Gesture Recognition," *IEEE Comp. Graph. Appl.*, vol. 22, no. 6, Dec. 2002, pp. 64-71.

[12] S.C. Crampton and M. Betke, "Counting Fingers in Real Time: A Webcam-Based Human-Computer Interface with Game Applications," *Proc. Conf. Universal Access Human-Comput. Interaction*, 2003, pp. 1357-1361.

[13] J. MacLean et al., "Fast Hand Gesture Recognition for Real-Time Teleconferencing Applications," *Int. Workshop Recognition, Analysis Tracking of Faces Gestures Real Time Syst.*, 2001, pp. 133-140.

[14] R. Belaroussi and M. Milgram, "A Real Time Fingers Detection by Symmetry Transform Using a Two Cameras System," *LNCS*, vol. 5359, 2008, pp. 703-712.

[15] O. Gallo, S.M. Arteaga, and J.E. Davis, "A Camera-Based Pointing Interface for Mobile Devices," *IEEE Int. Conf. Image Processing*, 2008, pp. 1420-1423.

[16] B. Stenger et al., "Model-Based Hand Tracking Using a Hierarchical Bayesian Filter," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 28, no. 9, Sept. 2006, pp. 1372-1384.

[17] J. Alon et al., "A Unified Framework for Gesture Recognition and Spatiotemporal Gesture Segmentation," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 31, no. 9, Sept. 2009, pp. 1685-1699.

[18] H. Ying et al., "Fingertip Detection and Tracking Using 2D and 3D Information," *Proc. 7th World Congress Intell. Control Autom.*, 2008, pp. 1149-1152.

[19] C. Nolker and H. Ritter, "Detection of Fingertips in Human Hand Movement Sequences," *Gesture Sign Language Human-Computer Interaction*, 1997, pp. 209-218.

[20] A.M. Burns and B. Mazzarino. "Finger Tracking Methods Using EyesWeb," *LNCS*, vol. 3881, 2006, pp. 156-167.

[21] J. Hannuksela et al., "Motion-Based Finger Tracking for User Interface with Mobile Devices," *IET 4th European Conf. Visual Media Production*, 2007.

[22] A.A. Argyros and M.I.A. Lourakis, "Vision-Based Interpretation of Hand Gestures for Remote Control of a Computer Mouse," *LNCS*, vol. 3979, 2006, pp. 40-51.

[23] J. Hannuksela et al., "Adaptive Motion-Based Gesture Recognition Interface for Mobile Phones," *LNCS*, vol. 5008, 2008, pp. 271-280.

[24] Y. Hirobe and T. Niikura, "Vision-Based Input Interface for Mobile Devices with High-Speed Fingertip Tracking," *ACM Symp. User Interface, Software Technol.*, 2009, pp. 7-8.

[25] Z. Zivkovic, "Improved Adaptive Gaussian Mixture Model for Background Subtraction," *Int. Conf. Pattern Recog.*, 2004, pp. 28-31.

[26] Y.J. Lee and D.H. Lee, "Research on Detecting Face and Hands for Motion-Based Game Using Web Camera," *Int. Conf. Security Technol.*, 2008, pp. 7-12.

[27] A. Rosenfeld and E. Johnston, "Angle Detection on Digital Curves," *IEEE Trans. Comp.*, vol. 22, no. 9, Sept. 1973, pp. 875-878.

[28] A. Rosenfeld and J.S. Weszka, "An Improved Method of Angle Detection on Digital Curves," *IEEE Trans. Comp.*, vol. 24, no. 9, Sept. 1975, pp. 940-941.

[29] R.E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *J. Basic Eng.*, vol. 82-D, 1960, pp. 35-45.

Daeho Lee received his BS, MS, and PhD in electronic engineering from Kyung Hee University, Rep. of Korea, in 1998, 2001, and 2005, respectively. He was an assistant professor in the Faculty of General Education at Kyung Hee University from September 2005 to February 2011. Since March 2011, he has been an assistant professor at Kyung Hee University, in Humanitas College. His research interests include computer vision, pattern recognition, image processing, digital signal processing, intelligent transportation systems, human-computer interaction, and computer games.

SeungGwan Lee received the BS, MS, and PhD from the Department of Computer Engineering at Kyung Hee University, in 1997, 1999, and 2004, respectively. He was a visiting professor in the School of Computer Science and Information Engineering at Catholic University from 2004 to 2006. He is an assistant professor in the Humanitas College at Kyung Hee University. His research interests include artificial intelligence, meta-search algorithms, multiagents, ubiquitous computing, image processing, and robot soccer.