SICE Annual Conference 2013
September 14-17, 2013, Nagoya University, Nagoya, Japan

Recognition of Finger–Pointing Direction Using Color Clustering and Image Segmentation

Hidetsugu Asano[1], Takeshi Nagayasu[2], Tatsuya Orimo[1], Kenji Terabayashi[3], Mutsumi Ohta[1] and Kazunori Umeda[4]

[1]R&D Division, PIONEER Corporation, Kanagawa, Japan
(Tel: +81-44-580-3211; E-mail: {hidetsugu_asano, tatsuya_orimo, mutsumi_oota}@post.pioneer.co.jp)
[2]Course of Precision Engineering, Chuo University, Tokyo, Japan
(Tel: +81-3-3817-1826; E-mail: nagayasu@sensor.mech.chuo-u.ac.jp)
[3]Department of Mechanical Engineering, Shizuoka University, Shizuoka, Japan
(Tel: +81-53-478-1036; E-mail: tera@eng.shizuoka.ac.jp)
[4]Department of Precision Mechanics, Chuo University, Tokyo, Japan
(Tel: +81-3-3817-1826; E-mail: umeda@mech.chuo-u.ac.jp)

Abstract: An accurate method of recognizing finger–pointing direction using multiple cameras is proposed. Input images are segmented by color clustering prior to shape analysis. Some restrictions in previous methods, such as narrowness of operation area and predefined skin color, are removed. Finger–pointing recognition in eight directions is achieved with an average of 90% recall and 93% precision.

Keywords: Intelligent Room, Gesture Interface, Human Machine Interface, Image Processing, Network Camera

1. INTRODUCTION

Human gestures are efficient for human–machine interfaces [1][2]. Dominguez et al. showed a finger–tracking method with a wearable camera [3], but a user must wear equipment. Sato et al. showed a method for tracking a user's hand in 3D and recognizing the hand's gesture with multiple cameras [4]. Fukumoto et al. [5] and Hu et al. [6] proposed hand–pointing estimation methods using orthogonal cameras. These methods had problems since the area where users can make gestures is limited. Irie et al. have proposed a gesture interface system to control any appliance from anywhere in the Intelligent Room simply by pointing a finger at a specific place [7]. This interface removed the restriction of area and has excellent usability, but it requires a predefined skin color. With this restriction, the system is not flexible enough to accommodate a variety of skin colors.

To remove this restriction, we propose a new method to recognize finger–pointing direction in the Intelligent Room without predefining a skin color. Input images are segmented with color clustering, and the shapes are analyzed to obtain the hand region. The finger–pointing direction is estimated using shape analysis of the hand region.

This paper is organized as follows: Section 2 describes the detailed algorithm, section 3 shows the experimental results, and section 4 presents the conclusion.

2. ALGORITHM

The Intelligent Room has functions to recognize human gestures and control appliances. It consists of an IR controller, a PC, and four cameras that are installed to eliminate blind areas and achieve 3D measurements.

The proposed method to recognize finger–pointing direction is carried out by the following steps:

- Detecting finger waving to specify the operator [8]
- Zooming in the operator's hand
- Recognizing finger–pointing direction from captured images (Fig. 1)

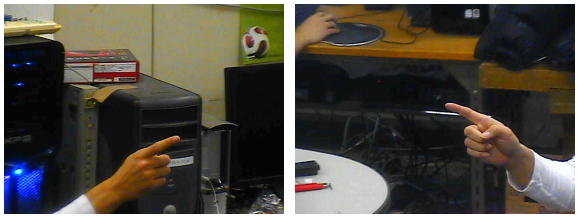


Fig. 1 Example of input images.

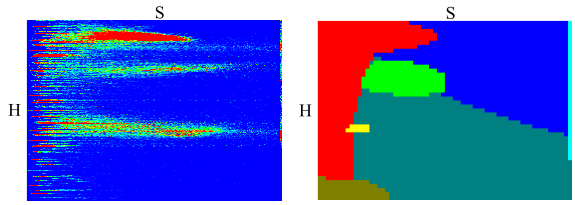


Fig. 2 (Left) 2D histogram in HS color space. Red shows high frequency, blue shows low frequency. (Right) Result of clustering. Each color shows a class.

To improve the usability for a variety of skin colors, the proposed method does not use predefined color. The algorithm consists of "color space segmentation," "image region segmentation," "hand shape analysis," "finger extraction," and "finger-pointing direction estimation."

2.1 Color Space Clustering

We assume the hand regions consist of a few colors and can be obtained using the results of color clustering. To make color classes from biased samples, Gaussian Mixture Model (GMM) fitting using EM algorithm

PR0001/13/0000-2029 ¥400

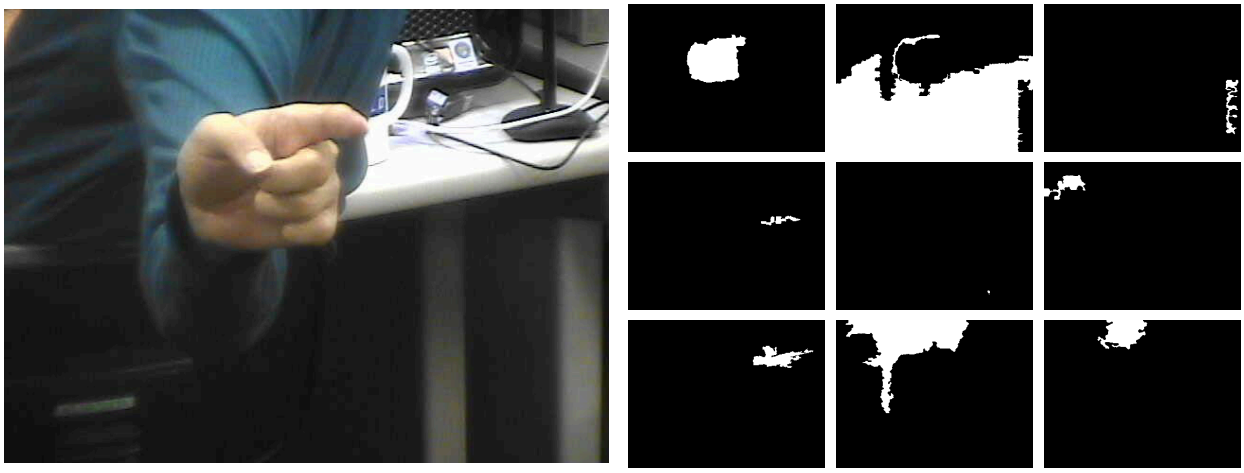


Fig. 3 Image segmentation results using every GMM class. (Left) Target image. (Right) Segmented images using each GMM class. In this example, number of initial GMM classes is 10, and one class degenerated.

is applied to a 2D histogram in a Hue-Saturation (HS) color space. One or two color classes may include a hand region. Fig. 2 shows a sample of a 2D histogram and a clustering result.

2.2 Image Segmentation

To obtain segmented regions, graph cut image segmentation [9] is applied to the image using every GMM class. We use the energy function in Rother et al. [10]. The largest and second largest regions are obtained as candidates for the hand region.

In this paper, the graph cut with only a single class is applied. If multiclass segmentation is applied, a versatile hand region extraction can be done, even when the user wears a multicolored glove. Fig. 3 shows the segmentation result.

2.3 Hand Shape Analysis

To analyze hand shapes, regions that include the user's hand are evaluated from the obtained regions. In this step, the following features of the region shape are evaluated:

- Number of pixels
- Standard deviation of the distance between region's centroid and contour pixels
- Rate at which the region's contour pixels are image contour

Under these conditions, evaluation function (1) is used.

$$E = \begin{cases} N(\frac{1}{\sigma_r} + \lambda O_r^2) & if \;\; N < M \\ 0 & otherwise \end{cases} \tag{1}$$

where N is the number of pixels in the region, σ_r is the standard deviation of the distance between the region's centroid and contour pixels, O_r is the rate at which the region's contour pixels are image border, λ is a weight, and M is the maximum number of pixels of the hand region. The region that has a maximum value of (1) is estimated as a hand region.

2.4 Finger Region Extraction

The finger region is extracted from the hand region. The finger region is assumed to be a narrow, elongated area in the hand region. Contour pixels are used for shape analysis. Finger region extraction is carried out by the following steps:

- To obtain a tangent line of every contour pixel, a least squares line fitting is applied to pixels with 3 neighboring contour pixels. A normal line is obtained from the tangent line, and the number of the pixels D that cross the normal line are counted (Fig. 4).
- Calculate the average μ_D and standard deviation σ_D of D (Fig. 5).
- Contour pixels under condition (2) are considered as a contour of the narrow region.

  $$D < \mu_D - a \times \sigma_D \tag{2}$$

  In equation (2), a is a parameter that controls how narrow the finger is. The hand region pixels on the normal line of the narrow region's contour pixels are filled. The largest filled region is obtained as a finger region.

The finger region can be extracted with this approach. When compared to the result that is extracted with morphology, our method can extract the finger region accurately (Fig. 6).

To reject the case that the pointing finger is occluded or is inside the hand region, or the case that the hand region estimation failed, the finger region should satisfy the conditions (3) to (5).

$$L_D \leq n_L \tag{3}$$

$$\mu_D \geq n_\mu \tag{4}$$

$$\sigma_D \geq n_\sigma \tag{5}$$

L_D is a length of the hand region contour and n_L, n_μ, and n_σ are thresholds for L_D, μ_D, and σ_D respectively. If the extracted region does not satisfy these conditions, the image is assumed to contain no finger region.

Fig. 4 An example of a tangent line and a normal line for contour pixels.

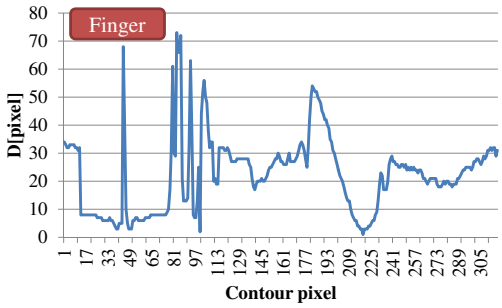


Fig. 5 D for each contour pixel. D is small at the finger region.The peak in the finger region shows a fingertip.

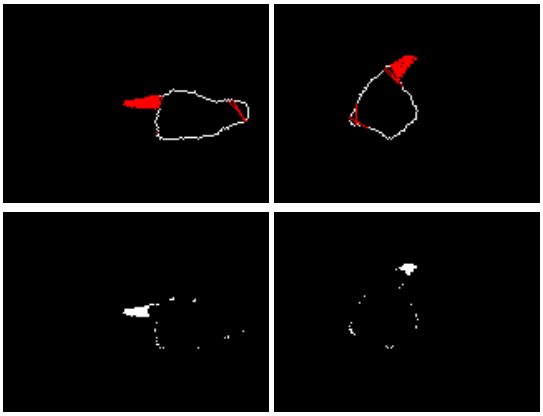


Fig. 6 (Upper) Finger extraction results using our method. (Lower) Results with morphology [7].

2.5 Finger–Pointing Direction Estimation

To estimate finger–pointing direction in the image, the least squares line fitting is applied to pixels in the finger region. By combining the results of multiple cameras, the finger–pointing direction in the 3D space can be estimated [7]. Since the proposed method works on each input image, the result may be unstable because of fitting error or failure of region estimation. To stabilize these results, a 13–tap median filter is applied.

3. EXPERIMENT

We implemented our algorithm on Intel®Core ™i7 975 and NVIDIA®GeForce®GTX580. Pan–Tilt–Zoom cameras Axis 233D were used and installed at the four corners in a room. Positions of installed cameras are shown in Table 1. The room size was 6.9m by 7.8m. Input images were resized from 640×480 to 160×120.

In the proposed algorithm, "color space clustering," "image segmentation," and "hand shape analysis" are time-consuming. Since a dynamic image change is rare, these steps are not needed for all input images. On the other hand, it is desirable to estimate finger–pointing direction for every frame. For this reason, these steps were implemented separately and run on different threads. The former steps were processed every 100ms to update color class information, which took 50ms. The latter steps took 5ms for each camera. GMM and graph cut were processed on a GPU; others were processed on a CPU. Parameters of this experiment are shown in Table 3.

Table 1 Camera positions [m]

	X	Y	Z
Camera1	0.59	0.53	2.11
Camera2	5.67	0.37	2.18
Camera3	0.13	7.15	2.47
Camera4	6.22	5.78	2.39

Fig. 7 gives the results of the hand region estimation and shows that the proposed method can extract the hand region accurately. Fig. 8 shows the result of the finger–pointing direction estimation. The proposed method can estimate finger–pointing direction accurately when a finger region is taken clearly, and it can reject results when the finger is occluded or inside the hand region. Fig. 9, 10 show the results when the user wore a short sleeve. The proposed method can estimate direction regardless of user's clothes. Fig. 11 shows the results when the user wore a colored glove. Our method does not use predefined color restriction, so it can extract the hand region even if the user wears a colored glove.

Finger pointing is made from 5 different points in 8 directions parallel to the XY plane (Fig. 12, Table 2). In our experiment, 150 samples were estimated in each direction. Table 4 shows the average angular errors and standard deviations. Table 5 shows 8–directional recognition results. The total average is 90% recall and 93% precision. This confirms that our method can recognize finger–pointing direction correctly. Angular error is high in some points, especially in cases where the image of the finger is not taken clearly or an inverse direction is falsely recognized.

Fig. 13 shows the time series of the estimation results from pos.E to dir.1 and 8. In dir.1, the proposed method can estimate the direction stably and precisely because every camera can capture the finger clearly. In dir.8, the result is unstable. This is because the finger is directed to the camera, and only two cameras can capture the finger. Fig. 14 shows the estimation errors projected on YZ plane in 1m from pos.E pointed to dir.1. From this result, most of the errors are under 0.4m in Y axis and 0.2m in Z axis.

4. CONCLUSION

We have proposed a method to recognize finger–pointing direction without using a predefined color. The method consists of "color space segmentation," "image region segmentation," "hand shape analysis," "finger extraction," and "finger–pointing direction estimation." The proposed method works with 90% recall and 93% preci-

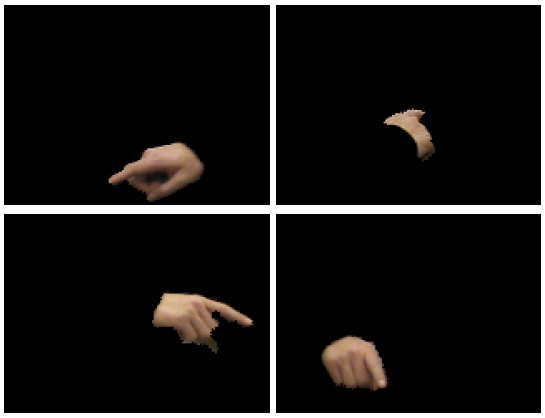


Fig. 7 Results of hand region estimation with long sleeve.

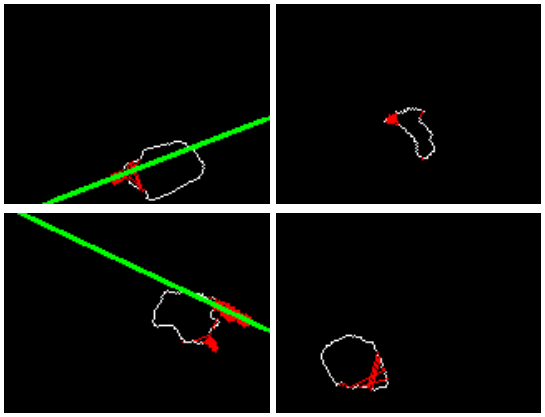


Fig. 8 Results of finger–pointing direction estimation from Fig. 7. Red regions are candidates of the finger, green lines are the estimated finger–pointing direction.

Table 2 Experimental positions [m]

Position	X	Y	Z
A	2.30	2.60	1.00
B	4.60	2.60	1.00
C	2.30	5.20	1.00
D	4.60	5.20	1.00
E	3.45	3.90	1.00

Table 3 Parameters in the experiment.

Number of GMM classes	10
λ	0.1
M[pixel]	$width \times height/2$
a	0.75
N_L[pixel]	300
N_μ[pixel]	14
N_σ[pixel]	10

sion on average. Future work will focus on improving the stability and evaluating the usability of this interface.

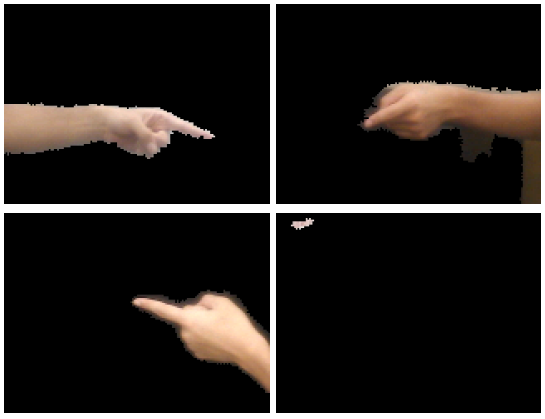


Fig. 9 Results of hand region estimation with short sleeve.

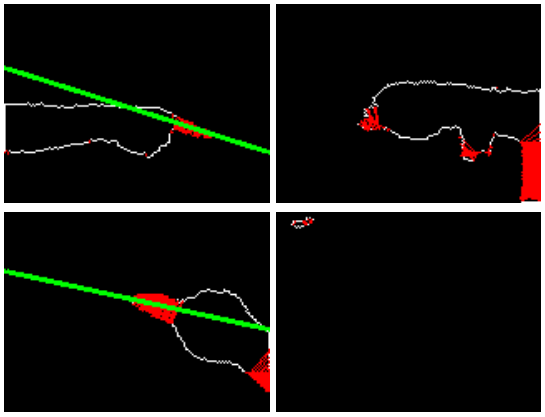


Fig. 10 Results of finger–pointing direction estimation from Fig. 9.

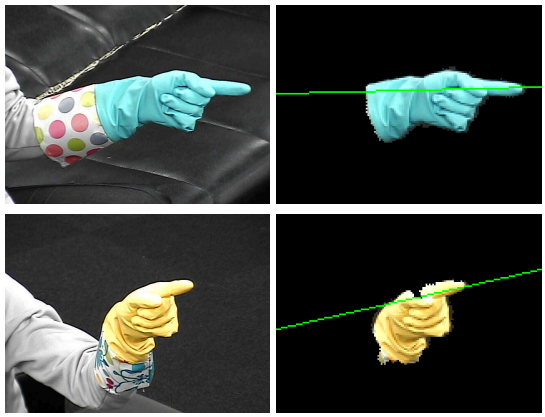


Fig. 11 Results when user wore a colored glove.

REFERENCES

[1] T. Moeslund, and E. Granum, "A survey of computer vision–based human motion capture", *CVIU*, 81, 3, 2001.

[2] V. I. Pavlovic, R. Sharma, and T. S. Huang, "Visual interpretation of hand gestures for human–computer interaction: a review", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.19, pp.677–695, 1997.

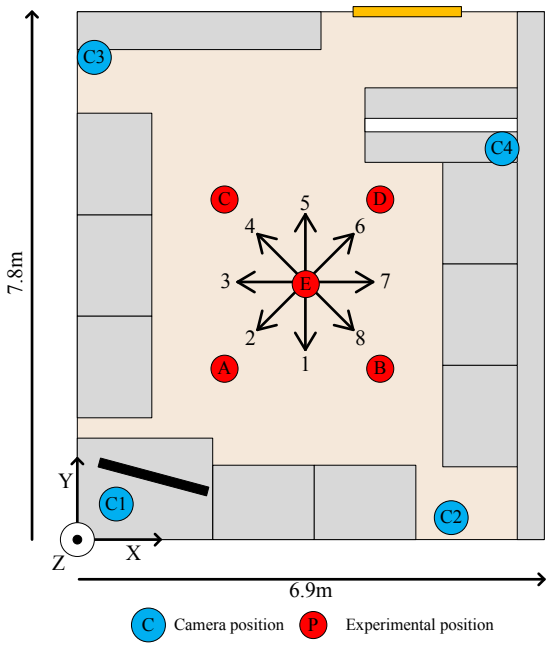
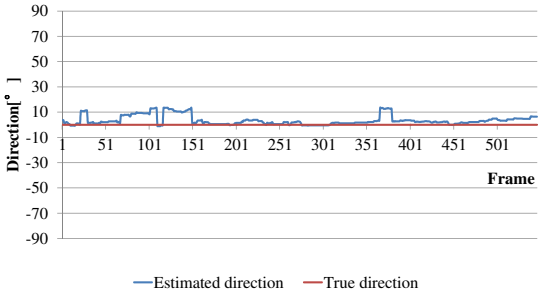
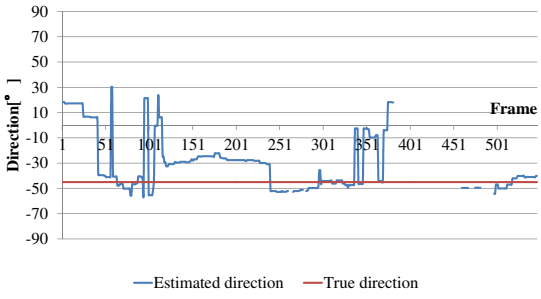


Fig. 12 Points and directions in experiments.



(a) Finger–pointing from pos.E to dir.1



(b) Finger–pointing from pos.E to dir.8

Fig. 13 The time series of the estimation results.

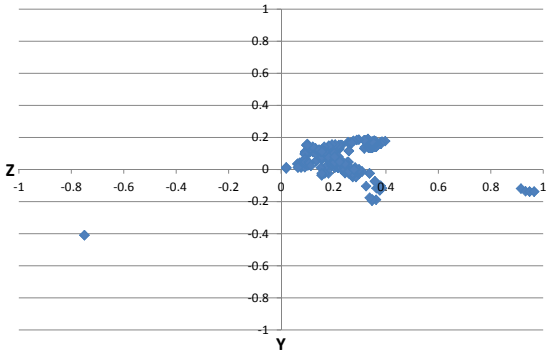


Fig. 14 Estimation errors projected on YZ plane in 1m from pos.E pointed to dir.1 [m]

pointer": Pointing interface by image processing", *Computer & Graphics*, Vol.18, No.5, 1994.

[6] K. Hu, S. J. Canavan, and L. Yin, "Hand Pointing Estimation for Human Computer Interaction Based on Two Orthogonal–Views", *ICPR2010*, 2010.

[7] K. Irie, N. Wakamura, and K. Umeda, "Construction of an intelligent room based on gesture recognition: operation of electric appliances with hand gestures", *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004)*, Vol.1, pp.193–198, 2004.

[8] K. Terabayashi, H. Asano, T. Nagayasu, T. Orimo, and K. Umeda, "Detection of small–waving hand by distributed camera system", *International Conference on Network Sensing Systems (INSS2012)*, 2012.

[9] Y. Boykov, and M. Jolly, "Interactive graph cuts for optimal boundary and region segmentation of objects in N–D images", *In Proc. IEEE Int. Conf. on Computer Vision*, 2001.

[10] C. Rother, V. Kolmogorv, and A. Blake, " "GrabCut": interactive foreground extraction using iterated graph cuts", *ACM Trans. Graphics (SIGGRAPH 2004)*, Vol.23, No.3, pp.309–314, 2004.

[3] S. M. Dominguez, T. Keaton, and A.H. Sayed, "A robust finger tracking method for multimodal wearable computer interfacing", *IEEE Trans. on Multimedia*, Vol.8, No.5, pp.956–972, 2006.

[4] Y. Sato, M. Saito, and H. Koike, "Real–time input of 3D pose and gestures of a user's hand and its applications for HCI", *Proc. of IEEE Virtual Reality Conference*, pp.79–86, 2001.

[5] M.Fukumoto, Y. Suenaga, and K. Mase, " "Finger–

Table 4 Average angular error and standard deviation. [deg]

		Direction							
		1	2	3	4	5	6	7	8
Pos.A	Average error	9.15	5.43	9.77	4.48	2.15	9.81	11.35	6.93
	S. D.	1.57	1.00	13.69	3.26	2.82	1.04	7.89	1.05
Pos.B	Average error	17.19	5.79	14.13	6.74	11.44	8.97	18.57	10.28
	S. D.	8.51	2.22	8.52	4.09	7.08	7.43	11.77	6.61
Pos.C	Average error	8.56	7.04	7.95	8.92	7.87	13.02	16.32	11.42
	S. D.	4.79	1.78	4.88	3.88	1.75	6.20	17.61	0.72
Pos.D	Average error	9.15	3.01	15.11	15.32	4.72	11.32	14.35	13.18
	S. D.	12.34	2.59	11.48	20.61	1.76	1.73	8.35	8.16
Pos.E	Average error	13.86	3.26	7.38	3.50	3.02	4.09	6.48	27.52
	S. D.	6.86	9.55	5.38	1.34	2.30	1.99	4.68	24.72

Table 5 Recognition results in 8–direction.

		Direction							
		1	2	3	4	5	6	7	8
Pos.A	Recall	0.99	1.00	0.82	0.89	1.00	1.00	0.97	0.71
	Precision	0.99	1.00	0.82	1.00	1.00	1.00	0.97	1.00
Pos.B	Recall	0.65	0.95	0.81	1.00	0.98	0.83	0.53	0.97
	Precision	0.71	1.00	0.81	1.00	0.98	0.83	1.00	0.97
Pos.C	Recall	0.92	1.00	0.95	1.00	0.98	0.83	0.85	1.00
	Precision	0.92	1.00	0.95	1.00	0.98	0.83	0.85	1.00
Pos.D	Recall	0.90	1.00	0.83	0.82	1.00	1.00	0.83	0.65
	Precision	0.90	1.00	0.83	0.82	1.00	1.00	0.83	0.65
Pos.E	Recall	0.98	0.97	1.00	0.95	0.87	1.00	1.00	0.62
	Precision	0.98	0.97	1.00	1.00	0.87	1.00	1.00	0.62