# Estimating the Direction of Pointing Gestures using Spatial Positions of the Eye and Fingertip

Hiroki Watanabe Hitoshi Hongo Mamoru Yasumoto Kazuhiko Yamamoto

HOIP, Research and Development Department
Softopia Japan and JST
Ogaki City, Gifu, Japan 503-8569

Faculty of Engineering
Gifu University
Gifu City, Gifu, Japan 501-1193

**Abstract**

*We propose a multi-camera system that can detect pointing in any direction and estimate its precise direction. In general, when people point at something, the target seems to overlap with their fingertip in their line of sight. Therefore, we regard the pointing direction as the imaginary line connecting the eye to the fingertip. First, multiple cameras detect the face region using skin color. Then, using the integrated information from the cameras, we determine which one captures the best frontal view of the face. This camera is labeled the center camera. Next, we select the center camera and its neighboring cameras to detect the spatial positions of the eye and fingertip in stereo mode. The corresponding eye points are established by calculating their correlations. The corresponding fingertip points are found as edge points in each hand region. Finally, we find the target along the imaginary line connecting the subject's eye and fingertip.*

*Using our system, we estimated the pointing direction. The experiment shows that our system achieves a mean error of 0.97° and a variance of 1.48 in all pointing directions.*

## 1 Introduction

Why is a vision interface necessary for computers? The reason is to facilitate smooth communication with people. Vision is an important means to understand information expressed through faces and gestures. This information plays a very important role in human-computer interactions[1][2] and should be processed as follows:

1. Information is acquired by observing human behavior.

2. Information is adapted to individual human characteristics.

3. Information is offered according to human methods of perception.

We are in the process of establishing a "Percept-room," which intelligently assists human-computer interactions. Human gestures are essential for interpreting what a person is doing. Among natural human gestures, the pointing gesture is an effective means for people to notify others of what they are interested in. A number of systems have been proposed in the past for human-computer interaction based on pointing gestures. Some systems required the user to wear a special glove or magnetic sensors[3][4][5]. Others using image processing had a limited range of direction[6] and required calibration for each user's hand shape and posture[7]. For intelligent environment applications, it is important to detect pointing gestures from any direction and to estimate their precise direction, because targets can exist anywhere in the Percept-room, and the system must not put strain on the user. Jojic et al. defined the pointing direction as the imaginary line passing through the upper region of the head and the tip of the arm and estimated it[8]. However, the result was not very precise.

In this work, we propose a multi-camera system that can detect pointing in any direction and estimate its precise direction. In general, when people point at something, the target seems to overlap with their fingertip in their line of sight. Therefore, we regard the pointing direction as the imaginary line connecting the eye to the fingertip.

## 2 System configuration

Figure 1 shows our current system configuration. The system consists of a studio and image input equipment. The studio occupies an area $4.8m^2$ enclosed in a simple background. Inside the studio, eight color video cameras, placed at an interval of 45° on a horizontal plane, are mounted at the same level. Their

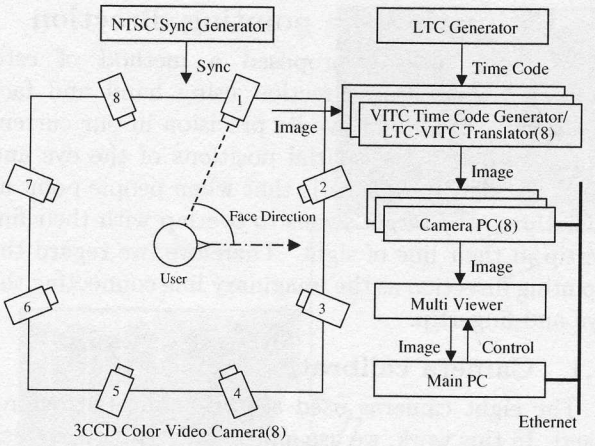



Figure 1: System configuration

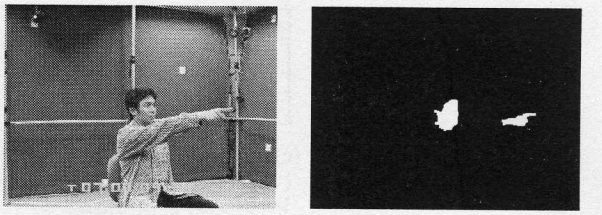
optical axes cross at the studio's center. Each camera is genlocked into a sync generator, and a time code is superimposed on their outputs. From each camera, 640 × 480, full color image data is captured at 30 fps, compressed into JPEG format, and stored in corresponding PCs. The multi-viewer combines the individual images into a single image. The main PC controls the multi-viewer and selects the input image.

In our experiment, a subject sits in the middle of a studio enclosed in a simple background. Applying these limitations, the system estimates the pointing direction as follows: first, the eight cameras detect the face and hand regions using color data. The skin color region in the middle of the image is the candidate for the face region. Then, the facial direction shown in each camera is estimated with the facial direction feature classes. Next, we estimate the precise facial direction with the integrated information from the cameras and determine which camera captures the best frontal view of the face. This camera is labeled the center camera. Then, we select the center camera and its neighboring cameras and detect the spatial positions of the eye and fingertip in stereo mode. Finally, the target is identified on the imaginary line that connects the positions of the eye and fingertip.

## 3 Skin color detection

We detect the face and hand regions using skin color data. The subjects we experimented on were primarily of Asian descent. To detect skin color regions from the input image, we use the CIE-L*u*v* color space[9]. Our detection method is described as follows:

1. A two-dimensional u*v* histogram is made from the input image.



(a)Original image (b)Detected skin color regions

Figure 2: Skin color detection

2. From the two-dimensional histogram, we select the color that covers the maximum number of pixels within the range of skin colors as the reference skin color.

3. The deviation from the reference skin color is calculated for each pixel in the input image.

4. We compose a histogram of the results obtained from step 3.

5. From the histogram, we determine a threshold using the method described by Otsu[10].

6. The skin color regions are detected from the input image using the threshold.

Figure 2(a) is the original image, and Figure 2(b) shows the detected skin color regions. The skin color region in the middle of the image is the candidate for the face region. The skin regions around the face region are the candidates for the hand region.

## 4 Capturing the frontal view of the face

Since the target exists in front of the face, we want to capture the frontal view of face. We employ four directional features and linear discriminant analysis in order to determine the camera that captures the best frontal view of the face[11].

### 4.1 Extracting the four directional features

The four directional features are extracted as follows:

1. We generate four edge images from the detected face region by applying Prewitt's operator in four directions (horizontally, vertically, and two diagonally).

2. Each edge image is normalized to a resolution of 8 × 8 pixels.

3. A 256-dimensional feature vector is constructed from these four edge images.

(a)Horizontal (b)Vertical (c)Diagonally (top right to bottom left) (d)Diagonally (top left to bottom right)

Figure 3: Four directional features

An example of the four directional features is shown in Figure 3. The four directional features are able to preserve the edge direction data at a low resolution. Consequently, these features are resistant to deformation and noise and can be processed quickly.

### 4.2 Linear discriminant analysis

Linear discriminant analysis is used to compose a discriminant space in which the intra-class variance is small and the inter-class distance is large. This analysis yields the coefficient matrix $\boldsymbol{A}$, which is the optimal linear projection of the known training data. The feature classes are composed from a linear discriminant analysis of the four directional features for each facial image. This method is described as follows:

1. The feature vector $\boldsymbol{x}$ extracted from the facial image is transformed into $\boldsymbol{y}$ using the coefficient matrix $\boldsymbol{A}$ obtained through linear discriminant analysis.

2. There exists a facial direction class $C_j (j = 1..16)$, in which $j$ is divided into 22.5° intervals. The distance $D_j$ between $\boldsymbol{y}$ and the mean vector $\overline{\boldsymbol{y}}_j$ of $C_j$ is calculated by equation(1).

$$D_j = |\boldsymbol{y} - \overline{\boldsymbol{y}}_j|^2 \tag{1}$$

3. The direction with the minimum $D_j$ value gives the approximate facial direction.

### 4.3 Coordinating multiple cameras

In the real world, occlusions between the face and camera may cause errors in estimation. Therefore, we use a multiple camera estimation method that is effective in this case[11]. By coordinating the information from all cameras, the facial direction is determined so that the evaluation value $F$ defined by equation(2) is largest.

$$F(j) = \sum_{q=1}^{8} \frac{1}{{D_j}^{(q)} + 1} \tag{2}$$

${D_j}^{(q)}$ is the distance obtained for the $q$-th camera. Once the facial direction is estimated, the pointing range is determined by the camera that captures the best frontal view of the face.

## 5 Estimating the pointing direction

We have already proposed a method of estimating the pointing direction using hand and face positions[12]. To improve its precision in our current work, we detect the spatial positions of the eye and fingertip. The basic idea is that when people point at something, the target seems to overlap with their fingertip in their line of sight. Therefore, we regard the pointing direction as the imaginary line connecting the eye and fingertip.

### 5.1 Camera calibration

The eight cameras used are all calibrated beforehand. In this work, we use a projective camera model. This model maps a 3D point $\boldsymbol{M} = [X, Y, Z]^t$ to a 2D image point $\boldsymbol{m} = [x, y]^t$ through a $3 \times 4$ projection matrix $\boldsymbol{P}$,

$$s\hat{\boldsymbol{m}} = \boldsymbol{P}\hat{\boldsymbol{M}} \tag{3}$$

$$\boldsymbol{P} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix}$$

where $s$ is a nonzero scale factor and the notation $\hat{\boldsymbol{p}}$ is such that if $\boldsymbol{p} = [x, y, ...]^t$ then $\hat{\boldsymbol{p}} = [x, y, ..., 1]^t$. Furthermore, $\boldsymbol{P}$ is given by

$$\boldsymbol{P} = \boldsymbol{A}[\boldsymbol{R}, \boldsymbol{t}] \tag{4}$$

where $\boldsymbol{A}$ is the $3 \times 3$ intrinsic parameter matrix, $\boldsymbol{R}$ is the $3 \times 3$ rotation matrix, and $\boldsymbol{t}$ is the $3 \times 1$ translation vector defining the rigid displacement between the world coordinate system and the camera coordinate system. All of these matrices can be computed with good accuracy by means of a classical calibration method as suggested by Faugeras[13].

By eliminating the scalar $s$ associated with the projection equation(3) as well as the point $\boldsymbol{M}$, we obtain an equation relating a pair of projections of the same 3D point:

$$\hat{\boldsymbol{m}}_1 \boldsymbol{F} \hat{\boldsymbol{m}}_2 = 0 \tag{5}$$

where $\boldsymbol{F}$ is the $3 \times 3$ fundamental matrix that provides the epipolar line of equation(5), and $m_1$ and $m_2$ are points in images $I_1$ and $I_2$. We use these equations to detect the spatial position of the eye and fingertip.

### 5.2 Camera selection

To use stereo vision, we select three appropriate cameras. First, we identify the camera that captures the best frontal view of face and label that camera the center camera. Next, its neighboring cameras are selected and labeled the right and left cameras. The

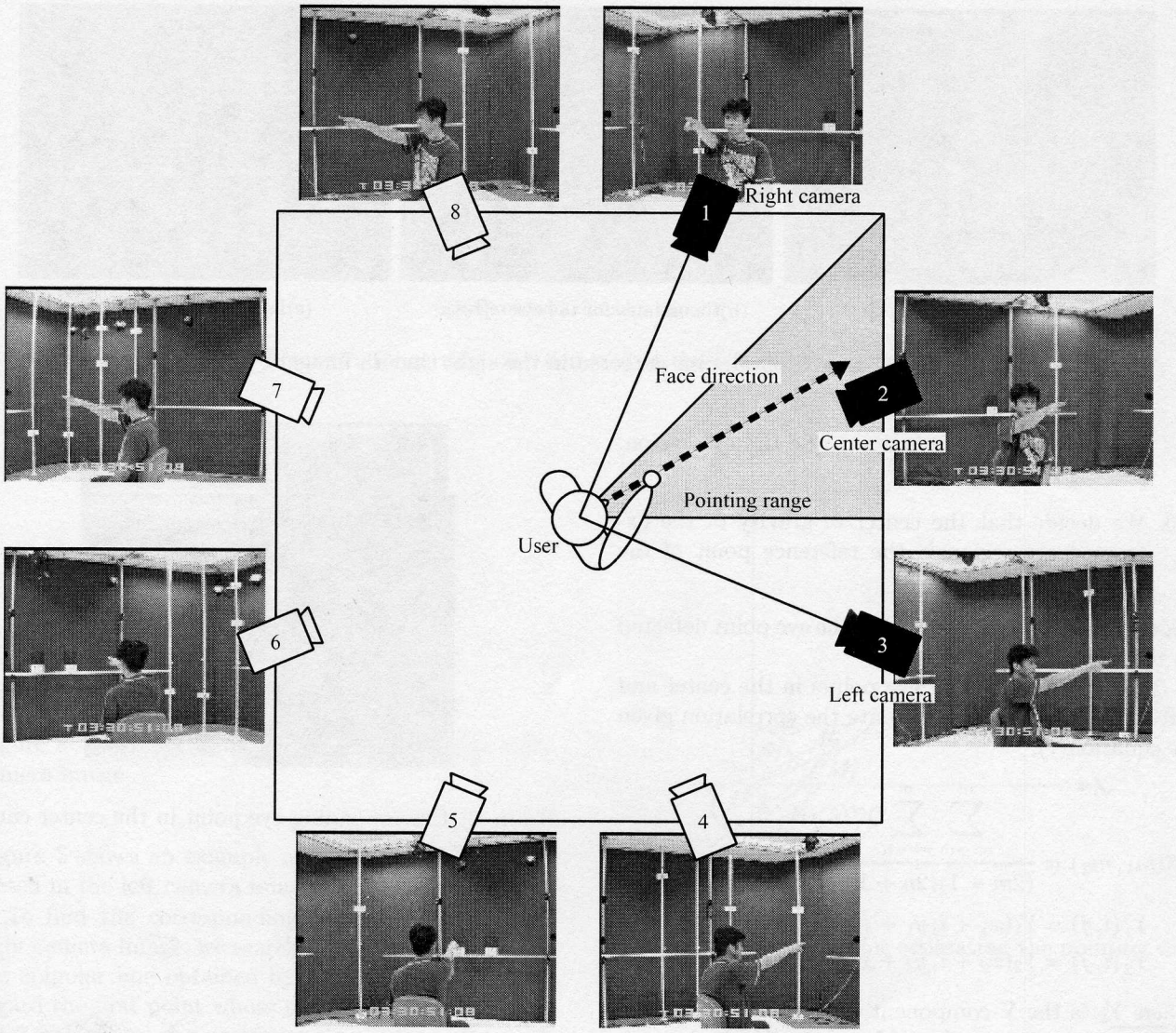



Figure 4: Selected cameras

pointing range is set to $\pm 22.5°$ from the center camera. Figure 4 shows an example of the selected cameras. The three images that are captured by the center camera and its neighboring cameras are scaled to $320 \times 240$ pixels and assembled into a mosaic by the multi-viewer.

### 5.3 Detecting the spatial position of the eye

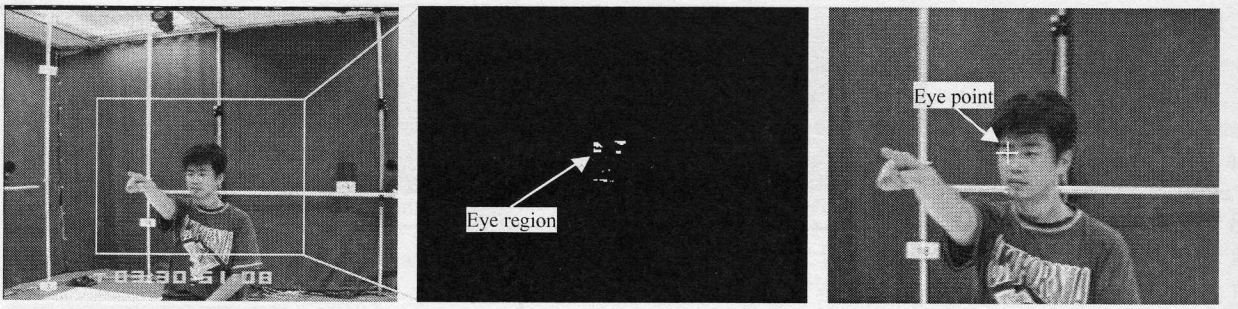
In order to extract a reference point for the eye, we select the image from the right camera because it contains no overlaps between the hand regions and the face. In this work, we use the YIQ color space to extract the eye regions. Each component of the YIQ color space is calculated by equation(6).

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.522 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (6)$$

The extraction method is described as follows:

1. The regions having a Y component that is smaller than $Y_{min}$ are extracted from the detected face region. $Y_{min}$ is experimentally determined.

2. Since the extracted regions include not only the eyes but also eyebrows, nostrils, etc., we consider the second white area from the top and the first




(a)Original image (b)Candidates for the eye regions (c)Detected eye point

Figure 5: Eye point detected in the right camera image

from the left of the image to be the eye region. See Figure 5(b).

3. We decide that the center of gravity of the extracted eye region is the reference point of the eye.

Figure 5(c) shows an example of the eye point detected in the right camera image.

To find corresponding eye points in the center and left camera images, we calculate the correlation given by equation(7),

$$S(m_1, m_2) = \frac{\sum_{i=-m}^{m} \sum_{j=-n}^{n} Y_1'(i,j) Y_2'(i,j)}{(2m+1)(2n+1)\sqrt{\sigma^2(Y_1)\sigma^2(Y_2)}} \quad (7)$$

$$Y_1'(i,j) = Y_1(x_1+i, y_1+j) - \overline{Y_1(x_1,y_1)}$$

$$Y_2'(i,j) = Y_2(x_2+i, y_2+j) - \overline{Y_2(x_2,y_2)}$$

where $Y_k$ is the Y component of image $I_k$ $(k = 1,2)$, $\overline{Y_k(x,y)}$ is the mean value of $Y_k$, and $\sigma(Y_k)$ is the variance. We determine two correlations, $S_1$ and $S_2$. $S_1$ is calculated between the right and center camera images, and $S_2$ is calculated between the right and left camera images. We select the point that has the maximum correlation out of $S_1$ and $S_2$. In this case, since $S_1$ is larger than $S_2$, the corresponding eye point is detected in the center camera image as shown in Figure 6.

Finally, $M_e$, the spatial position of the eye, is calculated by equation(8),

$$\boldsymbol{M} = \boldsymbol{B}^{+}\boldsymbol{b} \quad (8)$$

where

$$\boldsymbol{B} = \begin{bmatrix} x_1p_{31} - p_{11} & x_1p_{32} - p_{12} & x_1p_{33} - p_{13} \\ y_1p_{31} - p_{21} & y_1p_{32} - p_{22} & y_1p_{33} - p_{23} \\ x_2p'_{31} - p'_{11} & x_2p'_{32} - p'_{12} & x_2p'_{33} - p'_{13} \\ y_2p'_{31} - p'_{11} & y_2p'_{32} - p'_{12} & y_2p'_{33} - p'_{13} \end{bmatrix}$$

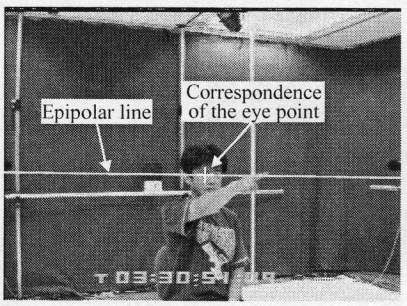



Figure 6: Corresponding eye point in the center camera image

$$\boldsymbol{b} = \begin{bmatrix} p_{14} - x_1p_{34} \\ p_{24} - y_1p_{34} \\ p'_{14} - x_2p'_{34} \\ p'_{24} - y_2p'_{34} \end{bmatrix}.$$

Equation(8) is derived from equation(3).

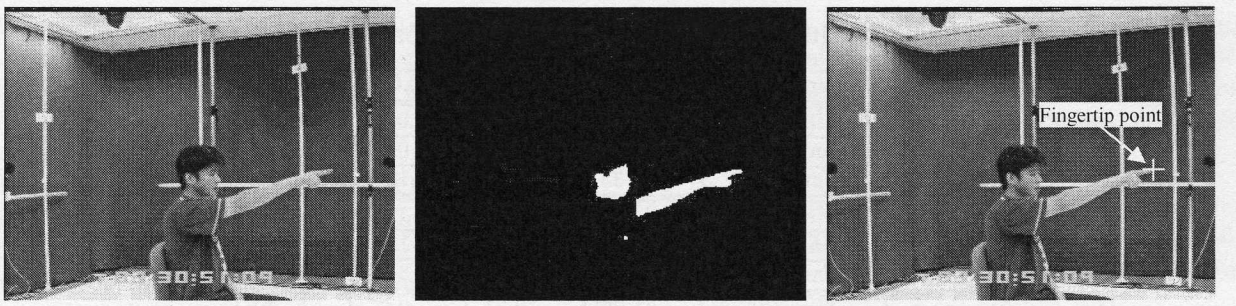
### 5.4 Detecting the spatial position of the fingertip

In order to extract a reference point for the fingertip, we select the left camera image. In this case, the fingertip appears as an edge point in the hand region. This makes extraction easy. The extraction method is described as follows:

1. The hand region is detected using an aspect ratio because the original region may include the arm. This region is labeled the new hand region.

2. The center of gravity of the new hand region is computed.

3. We select the farthest edge point from the center of gravity as the fingertip point.




(a)Original image (b)Detected skin color regions (c)Detected fingertip point

Figure 7: Fingertip point detected in the left camera image

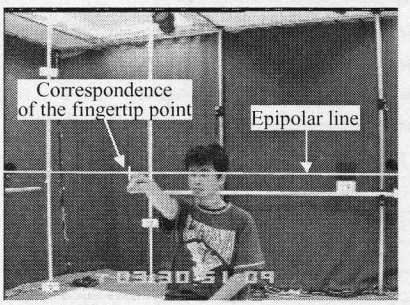



Figure 8: Corresponding fingertip point in the right camera image

Figure 7 shows an example of the fingertip point detected in the left camera image.

To find the corresponding fingertip point in the right camera image, we search from left to right along the epipolar line obtained by equation(5). Then, we regard the first point whose color lies within the skin color palette as the corresponding point. Figure 8 shows the point that corresponds to the fingertip point in Figure 7(c).

Finally, $M_f$, the spatial position of the fingertip, is calculated by equation(8).

### 5.5 Coordinates for estimating the pointing direction

Figure 9 shows the coordinate system used for estimating the pointing direction in the horizontal direction. We set up the eye position $M_e$, fingertip position $M_f$, target position $T$, and camera $C_q$ whose optical axis passes through the origin $O$. The pointing direction $\theta_p$ is calculated as the angle between the Y axis and the line $M_eM_f$. However, $\theta_p$ dose not remain constant when a person changes position. To detect which target a person is pointing to, the target direction $\theta_a$ is estimated

$$\theta_a = \phi_c - \phi_t \tag{9}$$

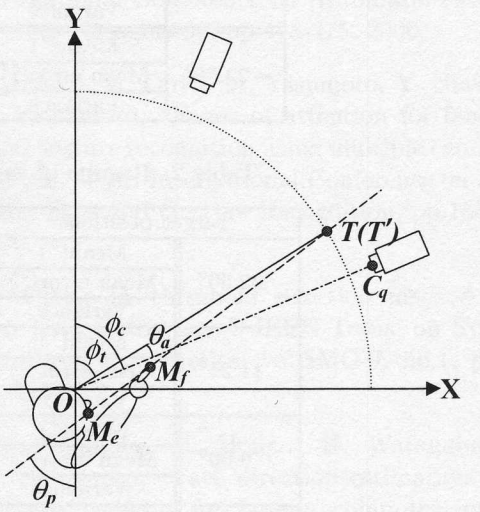



Figure 9: Coordinates for estimating the pointing direction

where $\phi_c$ is the angle between the Y-axis and the line $OC_q$, and $\phi_t$ is the angle between the Y-axis and the line $OT'$. The position $T'$ is the intersection of the extension of the line $M_eM_f$ and the circle whose center is $O$ and radius is $|OT|$. Similarly, the vertical direction of the target is estimated as the angle between the line $OC_q$ and the line $OT'$.

The resolution of this system is 0.3°. That is, one pixel shift of the eye/fingertip position causes an angle variation of 0.3°.

## 6 Experiment

In this experiment, subjects look at specific targets and point to them. The results show the estimated angle $\theta_a$ to 20 targets. The targets were placed at four angles in the horizontal direction (−22.5°, −11.25°, 0°, and 11.25°) and five angles in the vertical direction (−22.5°, −11.25°, 0°, 11.25°, and 22.5°). When

Table 1: Results of target estimation in the horizontal direction

| Target positions | | −22.50° | −11.25° | 0.00° | 11.25° |
|---|---|---|---|---|---|
| 22.50° | Mean(°) | −23.01 | −11.59 | 0.10 | 12.05 |
| | Mean error(°) | 1.02 | 0.99 | 0.94 | 1.08 |
| | Variance | 1.33 | 1.41 | 1.32 | 1.19 |
| 11.25° | Mean(°) | −22.76 | −11.71 | 1.60 | 9.80 |
| | Mean error(°) | 1.03 | 0.72 | 1.92 | 1.58 |
| | Variance | 1.33 | 0.73 | 3.98 | 1.76 |
| 0.00° | Mean(°) | −21.82 | −12.41 | 0.65 | 11.40 |
| | Mean error(°) | 0.80 | 1.16 | 1.14 | 0.77 |
| | Variance | 0.82 | 0.41 | 1.89 | 0.89 |
| −11.25° | Mean(°) | −22.61 | −10.92 | −0.56 | 11.10 |
| | Mean error(°) | 0.89 | 1.02 | 0.93 | 0.82 |
| | Variance | 1.42 | 1.35 | 0.98 | 1.05 |
| −22.50° | Mean(°) | −22.04 | −12.64 | −0.31 | 10.91 |
| | Mean error(°) | 0.82 | 1.39 | 0.75 | 1.28 |
| | Variance | 0.80 | 1.41 | 0.80 | 2.93 |

Table 2: Results of target estimation in the vertical direction

| Target positions | | −22.50° | −11.25° | 0.00° | 11.25° |
|---|---|---|---|---|---|
| 22.50° | Mean(°) | 21.64 | 21.21 | 21.96 | 22.40 |
| | Mean error(°) | 1.13 | 1.29 | 0.73 | 0.81 |
| | Variance | 1.61 | 0.41 | 0.61 | 1.20 |
| 11.25° | Mean(°) | 11.08 | 10.91 | 10.79 | 10.54 |
| | Mean error(°) | 0.72 | 0.76 | 1.03 | 1.08 |
| | Variance | 0.88 | 0.79 | 1.22 | 1.39 |
| 0.00° | Mean(°) | −0.40 | −0.18 | −0.32 | 0.29 |
| | Mean error(°) | 0.88 | 0.81 | 0.49 | 0.65 |
| | Variance | 1.15 | 1.11 | 0.27 | 0.56 |
| −11.25° | Mean(°) | −11.30 | −10.83 | −11.77 | −11.20 |
| | Mean error(°) | 0.74 | 0.69 | 0.91 | 0.91 |
| | Variance | 0.87 | 0.55 | 1.26 | 1.21 |
| −22.50° | Mean(°) | −22.08 | −21.70 | −21.83 | −23.01 |
| | Mean error(°) | 1.51 | 1.10 | 1.02 | 0.78 |
| | Variance | 3.24 | 1.12 | 1.61 | 0.73 |

the target is camera $C_q$, the horizontal angle is 0° and the vertical angle is 0°. In order to counteract pointing deviations between subjects, we had each subject point to camera $C_q$ first. The experimental results are adjusted based on this estimated angle. The experimental results for 12 right-handed subjects are summarized in Tables 1 and 2.

In Table 1, the values in the first row indicate the target horizontal angles and those in rows 2–6 indicate the mean of the estimated angle, mean error, and variance in the horizontal direction. The total mean error in the horizontal direction is 1.04°, and the total variance is 1.75. In Table 2, the values in the first column indicate the target vertical angles and those in columns 3–6 indicate the mean of the estimated angle, mean error, and variance in the vertical direction. The total mean error in the vertical direction is 0.89°, and the total variance is 1.24. The total mean error in all directions is 0.97°, and the total variance is 1.48.

We intend to use the pointing gesture to specify and control home appliances in the "Percept-room." Therefore, these results show that our proposed method has sufficient precision for our purpose.

## 7 Conclusion

We developed a system that can detect pointing in any direction and estimate its precise direction. To decide the pointing range, the eight cameras estimate the facial direction since the target always exists in front of the face. Then, to estimate the pointing di-

rection, three cameras detect the spatial positions of the eye and fingertip since the target exists along the imaginary line connecting these two points. Therefore, with this method, user initialization consists of pointing at the lens of the camera just once.

The experimental results show that the mean error is 0.97°, and the variance is 1.48. Thus, the spatial resolution is about 5cm at a distance of 3m from the user. We concluded that we can distinguish between targets when they are more than 5cm apart.

We plan to develop a real-time system in an actual environment. For that purpose, two main issues should be addressed in future research. One is skin color detection. In an actual environment, noise may occur in the images not only because the background is complex but also because other persons may exist in the room. This would require a robust algorithm for image thresholding and determining geometric relations between skin color regions. The other is pointing gesture detection. There are many kinds of human gestures. We will address the issue of differentiating certain pointing gestures from others.

## Acknowledgements

We are deeply indebted to Mr. Joshua Cole of the Research and Development Department of Softopia Japan, who collaborated in writing this paper.

## References

[1] M. Kaneko and O. Hasegawa, "Processing of face images and its applications," IEICE Trans. Information and Systems, vol.E82-D, no.3, pp.589-600, 1999.

[2] A. Pentland, "Looking at people: Sensing for ubiquitous and wearable computing," IEEE Trans. Pattern Analysis and Machine Intelligence, vol.22, no.1, pp.107-119, 2000.

[3] R. A. Bolt, " 'Put-that-there': Voice and gesture at the graphics interface," ACM-SIGGRAPH, vol.14, no.3, pp.262-270, 1980

[4] D. Wiemer and S. G. Ganapathy, "A synthetic visual environment with hand gesturing and voice input," In Proc. of the ACM Conference on Human Factors in Computing Systems, pp.235-240, 1989

[5] R. Cipolla, Y. Okamoto and Y. Kuno, "Qualitative visual interpretation of 3D hand gestures using motion parallax," In Proc. IAPR Workshop on Machine Vision Applications, pp.477-482, 1992.

[6] R. Cipolla, P. A. Hadfield and N. J. Hollinghurst, "Uncalibrated stereo vision with pointing for a man-machine interface," In Proc. IAPR Workshop on Machine Vision Applications, pp.163-166, 1994.

[7] M. Fukumoto, K. Mase and Y. Suenaga, "Real-time detection of pointing actions for a glove-free interface," In Proc. IAPR Workshop on Machine Vision Applications, pp.473-476, 1992.

[8] N. Jojic, B. Brumitt, B. Meyers, S. Harris and T. Huang, "Detection and estimation of pointing gestures in dense disparity maps," In Proc. of 4th International Conference on Automatic Face and Gesture Recognition, pp.468-475, 2000.

[9] H. Hongo, M. Ohya, M. Yasumoto, Y. Niwa and K. Yamamoto, "Focus of attention for face and hand gesture recognition using multiple cameras," In Proc. of 4th International Conference on Automatic Face and Gesture Recognition, pp.156-161, 2000.

[10] N. Otsu, "A threshold selection method from gray-level histograms," IEEE Trans. on System, Man, and Cybernetics, vol.SMC-9, no.1, pp.62-66, 1979.

[11] M. Yasumoto, H. Hongo, H. Watanabe and K. Yamamoto, "Face direction estimation using multiple cameras for human computer interaction," In Proc. of 3rd International Conference on Multimodal Interfaces, pp.222-229, 2000.

[12] H. Watanabe, H. Hongo, M. Yasumoto and K. Yamamoto, "Detection and estimation of omni-directional pointing gestures using multiple cameras," In Proc. IAPR Workshop on Machine Vision Applications, pp345-348, 2000.

[13] O. Faugeras, "Three-Dimensional Computer Vision: A Geometric Viewpoint," MIT Press, Cambridge, MA 1993.