Real-time Detection of Between-the-Eyes with a Circle Frequency Filter

Shinjiro Kawato and Nobuji Tetsutani
ATR Media Integration and Communications Research Laboratories
Seika-cho, Soraku-gun, Kyoto 619-0288, Japan
e-mail: {skawato|tetutani}@mic.atr.co.jp

Abstract

In this paper, we propose a method to extract the center point between the eyes or Between-the-Eyes in real time, which can be a good clue in detecting other face feature points. In this research, we put emphasis on real-time processing.

First, a skin-color region is extracted as a face candidate. Next, a circle-frequency filter, which we proposed before, is applied to this region to extract candidate points for Between-the-Eyes and also face rotation angles in the image plane at the candidate points. Each de-rotated local pattern at a candidate is compared with a template pattern of Between-the-Eyes, which is extracted from an image database of 400 faces, and a true candidate is selected.

We tested the algorithm with the face database, and the false detection rate was 1.5%. We implemented the system on a PC with a Pentium III 866MHz CPU. It ran at 27 frames/second without any special hardware.

1. Introduction

Gesture recognition is playing an important role in the advancement of human-computer interaction since it is providing a natural and efficient interface to computers. For head gesture recognition and/or face expression recognition, head and face detection and tracking in real time are the first steps.

We can take different strategies for face detection and face tracking. This is based on the finding that once a face is detected, the tracking appears easier. In this paper, we propose a real-time face detection method.

Feature-based tracking approaches for real-time 3D facial pose estimation are reported in [4][11]. In these papers, multiple feature points are tracked by template matching using a special correlation processor and the information is fed to a Kalman filter. Unfortunately, the papers fail to mention how the templates are made. Matsumoto extended this system to a stereo vision configuration and developed a new algorithm to make the system more robust and accurate enough to calculate the gaze direction [7]. However, template patterns still need to be made for each face before tracking. Our goal is to treat this subject more generally.

The general face location problem in an image seems to have been solved using skin-color information [10][2][8].

In a typical face region, many features are considered for face detection such as the eyes and/or their corners, the nostrils, and the mouth and/or its corners [3][9]. Among them, the shapes of the eyes and mouth can change drastically according to facial expression changes. The nostrils, in contrast, are very stable, but the observable face orientations relative to the camera direction are very limited. We therefor believe that these features are not suitable for head or facial gesture recognition.

We try to start with a part other than the eyes or the nostrils or the mouth. What is common to most people and easy to find for a wide range of face orientations? We claim that one possible candidate is the point between the eyes. We call this point “Between-the-Eyes”. Between-the-Eyes has dark parts on both sides including the eyes and eyebrows, and it is comparably bright on the upper side (forehead) and the lower side (nose bridge). This characteristic seems to be common to most people, and it can be seen for a wide range of face orientations. Moreover, the pattern around this point is fairly stable for any facial expression.

We earlier proposed an image filter, i.e., the circle-frequency filter (CF-filter), to detect Between-the-Eyes [5][6]. The CF-filter not only robustly extracts Between-the-Eyes, but also many other similar characteristic points. This means that we have to select the true Between-the-Eyes from among several candidates. In previous works [5][6], we tried to find eye-like regions on both sides of each candidate to confirm Between-the-Eyes. However the conditions for these eye-like regions ware too simple for real-time execution. As a result, the eye-brows or other hair parts were often taken as eye-like regions.

In this paper, we propose a template matching technique

for selecting the true Between-the-Eyes from among several candidates. In general, a template matching technique is weak in object rotations. With the CF-filter, however we can get not only an intensity output, but also a phase angle output. From this angle information, we can de-rotate the input image around each candidate point. This enables us to use a template matching technique.

We construct a gray level template for Between-the-Eyes from images of the ORL Database of Faces [1]. Our algorithm can detect Between-the-Eyes correctly for 394 face images out of 400 in the database. Of special note is that the images are monochrome (and still).

For real video images, we utilize skin color information for face candidate region extraction. Our experimental system on a Pentium III 866MHz PC can process 27 frames per second for 320×240 video image streams without any special hardware.

In section 2, we describe the circle-frequency filter and its application for Between-the-Eyes candidate extraction. In section 3, we describe our template matching algorithm for selecting the true Between-the-Eyes. In section 4, an implementation and in section 5, some experimental results are described. Section 6 concludes the paper.

2. Circle-Frequency Filter and Between-the-Eyes Candidate Extraction

Assume that f_k $(k = 0, ..., N-1)$ are pixel values along a circle of radius r centered at (x, y). Then, the CF-filter calculates intensity $f(x, y)$ and phase angle $\phi(x, y)$ as follows.

$$f(x,y) = \left(\sum_{k=0}^{N-1} f_k \cos(4\pi k/N)\right)^2 + \left(\sum_{k=0}^{N-1} f_k \sin(4\pi k/N)\right)^2 \quad (1)$$

$$\phi(x,y) = \tan^{-1}\left(\frac{\sum_{k=0}^{N-1} f_k \sin(4\pi k/N)}{\sum_{k=0}^{N-1} f_k \cos(4\pi k/N)}\right) \quad (2)$$

Here, f_k goes from the top $(x, y-r)$ along the circle in the counter-clockwise direction. Equation (1) gives the spectrum power of the frequency of two calculated by the discrete Fourier transform of data series f_k, and equation (2) gives its phase angle.

Between-the-Eyes has dark parts on both sides including the eyes and eyebrows, and it is comparably bright on the upper side (forehead) and the lower side (nose bridge). If we draw a circle of a certain radius centered at Between-the-Eyes, we can see two cycles of image intensities along this circle as bright – dark – bright – dark. When the center of the circle goes away from Between-the-Eyes, the two-cycle pattern gradually collapses. We can therefore expect the intensity output of the CF-filter to become the local maximum at Between-the-Eyes. When an image rotates in the image plane, the intensity output of the CF-filter does not change. This is a well-known Fourier transform characteristic. On the other hand, its phase angle output shifts exactly the same angle as the image rotation.

Figure 1 shows an example of f_ks at Between-the-Eyes (the white dot at the center). The radius of the circle is 7 pixels, and the 40 pixel values along it are plotted on the graph. The plot starts at the forehead and goes counter-clockwise. There are two cycles of dark and bright.

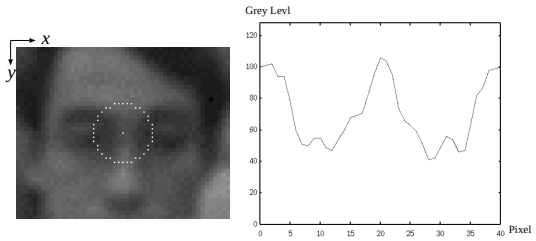


Figure 1. A plot of pixel values along a circle centered at Between-the-Eyes.

Figure 2 shows a CF-filtered image (in intensity) of Fig. 1. To display as an image, the gray level is normalized. At the peripheral, we cannot calculate equation (1), so we put the output as 0, where the width corresponds to the radius of the filter.

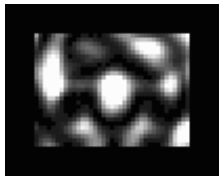


Figure 2. CF-Filtered image of Fig.1.

To evaluate the ability of the CF-filter in extracting Between-the-Eyes candidates, we applied it to face images of a database [1]. In this database, there are ten different images each of 40 distinct subjects. It is said that, for some of the subjects, the images were taken at different times, while varying the lighting, facial expressions (open/closed eyes, smiling/not smiling), and facial details (glasses/no glasses). All of the images are 8-bit monochrome and 92×112 in size.

First, each face image was placed over a large uniform background of gray level 128 so that equation (1) could be calculated at all points of the face image. After applying

a smoothing filter with 5×5 averaging, a $1/4$ size image was obtained by $1/2$ sub-sampling. The reason of the sub-sampling was because a characteristic pattern of Between-the-Eyes still remained in the reduced image and was recognizable. With the smaller image, we could apply a smaller CF-filter, which is an advantage for achieving a real-time system because of the shorter processing time.

To the smoothed reduced image, a CF-filter with a radius of 7 pixels was applied. Among the local maximum points of the filtered image, those points in the lower $1/3$ part were discarded where Between-the-Eyes is not likely to exist. Those points where the phase angle output was over ± 45 degrees were also discarded because we could not expect a lot of face rotations. Then, the top five points were selected as candidates for Between-the-Eyes.

Next, we checked each candidate to see if it was in a certain rectangular area, which is depicted in Fig. 3. Its bottom edge was on a line tangent to the bottom edges of the eyes, its width corresponded to the distance between the eye corners, and its height was a half of its width.

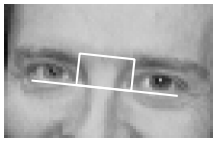


Figure 3. Area of Between-the-Eyes.

As a result, in 398 images out of 400, Between-the-Eyes was extracted as a candidate. This showed the effectiveness of the CF-filter in extracting Between-the-Eyes. Figure 4 shows examples of extracted candidates. In some cases, less than five points remain. Two cases of failure are shown in Fig. 5. For nine other images of the same persons, Between-the-Eyes was successfully extracted.

3. Selection of Between-the-Eyes

To select the true Between-the-Eyes among candidates, we apply a weighted template matching technique. In general, template matching techniques have a weak point in object rotation. We cannot apply it in a simple manner in consideration of the inclination of the face. As mentioned in the previous section, the CF-filter outputs not only intensity values but also phase angles corresponding to rotation angles. Therefore, we can de-rotate an input image at each candidate point. After the de-rotation, we can compare patterns with a template.

To make a template pattern of Between-the-Eyes, we used the above 398 face images from the database, from which our program could successfully extract Between-the-Eyes as a candidate. The candidate selection was done manually, but de-rotation of each image was done by a program, and 33×17 pixel patterns were extracted from the sub-sampled images. Figure 6 shows some examples of extracted patterns.

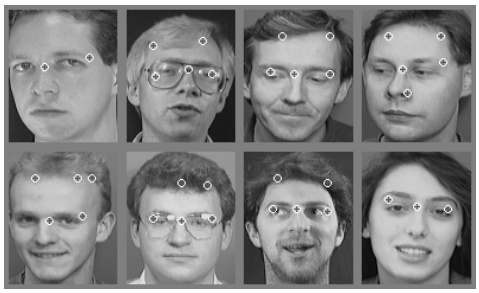
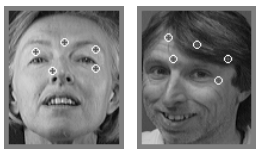


Figure 4. Examples of faces and extracted local maximum points from CF-filtered images.



(a) (b)

Figure 5. Face images in which Between-the-Eyes was not extracted as local maximum points from CF-filtered images.

Then, the gray levels of each sample pattern were normalized so that the average gray level was zero and the variance was 1. Symmetry patterns to the vertical center line were added as sample patterns, because they were also Between-the-Eyes patterns. Next, we calculated an average pattern of 796 samples, and the variance at each pixel position. Figure 7 shows the average pattern and the variance pattern. To show the average pattern as an image, the gray level was converted so that the average level was 128 and the standard deviation was 64. To show the variance pattern, each value was multiplied by 255. Of course, both patterns were symmetrical.

From the view point of the calculation load, the smaller the template, the better. We discarded three columns on both sides of the pattern because their variances were very large. We also discarded the top and bottom rows, because the pattern continued to retain the characteristic of Between-the-Eyes. We should evaluate the right half and the left half independently, because lighting conditions are likely to be different between the right and left halves of faces. Therefore we discarded the center column, and divided the average pattern into symmetric left and right patterns. After all, we kept a 13×15 pixel pattern for the left template, and a symmetry pattern for the right template. We also kept the corresponding variance values.

Figure 6. Examples of extracted Between-the-Eyes patterns from a database.

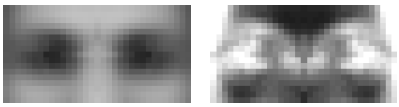


Figure 7. Average pattern of Between-the-Eyes and its variance with respect to each pixel.

The candidate evaluation process is as follows.

1. Calculate the phase angle output of the CF-filter at the candidate point using the original resolution image to get a more accurate rotation angle.

2. De-rotate the sub-sampled image at the candidate point, and then extract a 27×15 pixel pattern.

3. Normalize the gray levels of the left half and right half independently so that the average is 128 and standard deviation is 64.

4. At each pixel, calculate the square of the difference between the template and the candidate gray level, and multiply the inverse of the variance. Left-disparity and right-disparity are sum-up of them of the left half and right half.

5. If both of the disparities are less than a predefined threshold D, this candidate can be Between-the-Eyes. If no candidate satisfies this condition, the decision is "Give-up".

6. If only one candidate has left- and right-disparities less than D, it is Between-the-Eyes. If two or more candidates have such left- and right-disparities, the one that has the smallest total of disparities is Between-the-Eyes.

Figure 8 shows the numbers of true detections, give-ups, and false detections when the above evaluation process was applied to the 400 face images of the above database, while changing D from 150 to 600.

Under the severe condition that D is less than 200, the number of give-ups is larger than that of true detections, and there is no false detection. As D increases, the give-up count decreases and the true detection count increases monotonically. False detection appears at $D = 250$, but its count does not increase so much. At $D = 600$, the number of true detections is 394, the give-up count is 0, and the false detection count is 6.

Candidate patterns and selected patterns in the case of six false detections at $D = 600$ are shown in Fig. 9.

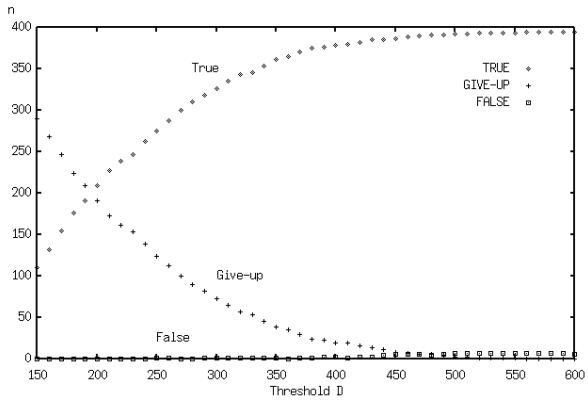


Figure 8. Detection rate vs. threshold value.

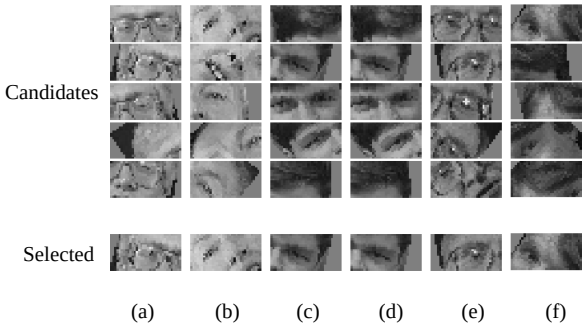


Figure 9. Examples of false detections.

4. Implementation

We implemented the system on a PC with a Pentium III 866MHz CPU. In the case of the database images, the face regions had already been segmented, and the backgrounds were nearly uniform. Now, our system can work in typical office environments without any background control.

If we can extract a face region, we can apply the algorithm we tested on the database images. A difference lies in the region size, which changes in every case. Here, we expect one face in a scene, and we use skin-color information for the extraction of the face region.

Many different color spaces have been proposed for skin-color extraction. We use a modified normalized rg space, namely (a, b) calculated by the following equations [5] because of its light calculation load in color conversion from the RGB space, where R, G, and B are the red, green, and blue components obtained from a color video camera.

$$r = R/(R+G+B), \quad (3)$$

$$g = G/(R+G+B), \quad (4)$$

$$a = r + g/2, \quad (5)$$

$$b = \sqrt{3}g/2. \quad (6)$$

Any color is projected onto a regular triangle area defined by $(0,0)$, $(1,0)$, $(1/2,\sqrt{3}/2)$. We digitize each coordinate of a,b in increments of 0.01 to form a discrete color space.

Our skin-color model is a color histogram of a face image. We take a face image, cut off very dark pixels by a program, cut off non-skin parts by hand, and then make a color histogram of the remaining pixels in the discrete (a,b) color space. Those colors where the histogram is over a threshold T are defined to be the skin-color. T is determined by hand, by looking at the extracted results of the skin-color regions.

In actual face region extraction, to cope with signal noise, we divide the original image into 8×8 pixel blocks, and for each block, if more than half of the pixels are skin-color we assume the block to be a skin-color block; otherwise, a non-skin color block. Then, a face candidate region is defined as a rectangle that includes all of the skin-color blocks.

A monochrome image and not a color image is required to apply a circle-frequency filter. There is a well-known linear combination equation of R,G,B for color to monochrome conversion. However we use an R image as a monochrome image to save on the conversion calculation time. For skin colors, the R component is relatively large, and so no problems result.

Figure 10 shows the general process of our system.

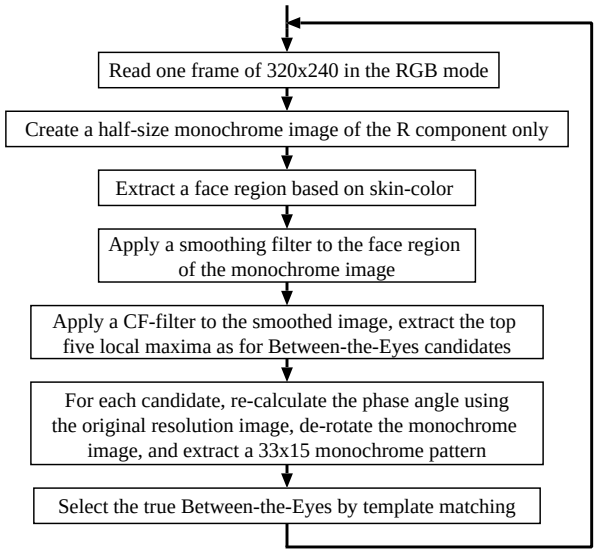


Figure 10. General flow.

5. Experiment

We use a commercial NTSC video camera and a video capture board. No other special H/W is implemented.

Figure 11 shows a monitor image of our system. The lower $2/3$ part is a color input image (320×240), although

Figure 11. Example of a result of experimental system.

the left 41 columns are overlayed showing the candidate patterns of Between-the-Eyes and selected one. The upper left image is a sub-sampled R component image. A smoothing filter is applied to the face candidate region extracted from the original image as a skin-color region.

The upper right image shows the intensity output of the CF-filter applied to the face candidate region of the upper left image. When the phase angle output exceeds ± 45 degrees, the intensity output is replaced with zero. The appropriate gray level conversion is applied to show it as an image.

The top five local maximum points in the CF-filtered image are extracted as candidates of Between-the-Eyes. Their positions are shown on the original input image by dots of overlayed graphics. There is a small segment hanging from each dot to show the phase angle output of the CF-filter at that point by its direction. The perpendicular direction is zero degrees.

The five de-rotated candidate patterns are shown in the middle of the left side, and the selected one is shown below them. The original face inclines to the right about 20 degrees. In the selected pattern, however, the eyes are aligned almost horizontally. This shows the effectiveness of de-rotation based on the phase angle output of the CF-filter.

A circle with a radius of 30 pixels centered at the selected candidate point is overlayed on the original image. By watching this circle, we can see if the result is true, false, or give-up (no circle).

For the faces in the database, the true detection rate increased when the threshold D became higher. In actual cases, however, even if a skin-color region is extracted, a face may not exist. In such cases with a large threshold D, the system obviously detects some point other than Between-the-Eyes. Refering to the graph of Fig. 8, the

threshold $D = 400$ is selected. It is in-between the cross point of true detection and give-up ($D = 200$) and the point ($D = 600$) where the detection rate saturates.

Our system runs at the rate of 27 frames per second, or 90% of the NTSC frame rate.

In the experiment, we noticed that our algorithm is pretty robust against scale changes as long as face image is frontal. The circles shown in Fig. 11 and Fig. 12 are of the same size. The faces in Fig. 12 differ in scale by almost two times, and yet our system can detect Between-the-Eyes in both cases. One reason for this characteristic is that the CF-filter detects a simple two-cycle pattern and this pattern keeps at Between-the-Eyes even when the scale changes. Another reason is that the template pattern in Fig. 7 does not have clear edge patterns in it, and it is used for pattern selection, but not pattern detection nor positioning. Therefore the threshold D can be very loose.

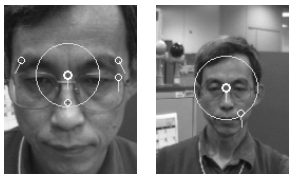


Figure 12. Robustness against scale changes.

When hair covers the forehead, our algorithm does not work, because the two-cycle pattern shown in Fig. 1 does not appear at Between-the-Eyes. In addition, black frame glasses often cause our system to fail.

6. Conclusion

Face feature point detection is a first step towards face gesture recognition. We proposed a method to detect Between-the-Eyes in a video stream in real time.

To efficiently detect Between-the-Eyes, we use a circle-frequency filter. Its intensity output becomes a local maximum at Between-the-Eyes, and its phase angle output denotes a rotation angle in the image plane. Therefore, we can easily find candidates for Between-the-Eyes, and de-rotate the input face image at the candidate point. This enables us to use a template matching technique to select the true Between-the-Eyes among all candidates.

We tested our algorithm with a face image database [1]. For 400 faces, a true detection rate of 98.5% was achieved.

In an application to a real video stream, we implemented a skin-color detection technique to extract a face candidate region. In the color to monochrome conversion for the CF-filter calculation, we used the R component only, to reduce the calculation load. Our system ran at a rate of 27 frames/second on a PC with a Pentium III 866MHz CPU without any special hardware.

The radius of the CF-filter was fixed to 7 pixels. If the filter size were to be made smaller than this, the resolution of the phase angle output would get too coarse. If it were to be made lager, the filtering process would take too long. For large-scale changes of the face, we think we should prepare multi-resolution images, instead of applying various filter sizes.

We plan to develop a face tracking system based on the current detection technique in the near future.

Acknowledgment

We give credit to AT&T Laboratories Cambridge for letting us use "The ORL Database of Faces"[1].

References

[1] AT&T Laboratories Cambridge. "The ORL Database of Faces". *http://www.cam-orl.co.uk/facedatabase.html*.

[2] Q. Chen, H. Wu, T. Fukumoto, and M. Yachida. 3D head pose estimation without feature tracking. *Proc. IEEE 3rd Int. Conf. on Automatic Face and Gesture Recognition*, pages 88–93, 1998.

[3] K. Fukui and O. Yamaguchi. Facial feature point extraction method based on combination of shape extraction and pattern matching. *Systems and Computers in Japan*, 29(6):49–58, 1998.

[4] J. Heinzmann and A. Zelinsky. 3-D facial pose and gaze point estimation using a robust real-time tracking paradigm. *Proc. IEEE 3rd Int. Conf. on Automatic Face and Gesture Recognition*, pages 142–147, 1998.

[5] S. Kawato and J. Ohya. Real-time detection of nodding and head-shaking by directly detecting and tracking "between-eyes". *Proc. IEEE 4th Int. Conf. on Automatic Face and Gesture Recognition*, pages 40–45, 2000.

[6] S. Kawato and J. Ohya. Two-step approach for real-time eye tracking with a new filtering technique. *Proc. Int. Conf. on System, Man & Cybernetics*, pages 1366–1371, 2000.

[7] Y. Matsumoto and A. Zelinsky. An algorithm for real-time stereo vision implementation. *Proc. IEEE 4th Int. Conf. on Automatic Face and Gesture Recognition*, pages 499–504, 2000.

[8] J. C. Terrillon, M. David, and S. Akamatsu. Automatic detection of human faces in natural scene images by use of a skin color model and of invariant moments. *Proc. IEEE 3rd Int. Conf. on Automatic Face and Gesture Recognition*, pages 112–117, 1998.

[9] J. Yang, R. Stiefelhagen, U. Meier, and A. Waibel. Real-time face and facial feature tracking and applications. *Proc. Workshop on Audio-Visual Speech Processing*, pages 79–84, 1998.

[10] J. Yang and A. Waibel. A real-time face tracker. *Proc. 3rd IEEE Workshop on Application of Computer Vision*, pages 142–147, 1996.

[11] A. Zelinsky and J. Heinzmann. Real-time visual recognition of facial gestures for human-computer interaction. *Proc. IEEE 2nd Int. Conf. on Automatic Face and Gesture Recognition*, pages 351–356, 1996.