

# A Real-Time Eye-Tracking System for Autostereoscopic Displays

Chan-Hung Su<sup>1,2</sup> (蘇展弘), Yong-Sheng Chen<sup>1</sup> (陳永昇),

Yi-Ping Hung<sup>1</sup> (洪一平), Chu-Song Chen<sup>1</sup> (陳祝嵩)

<sup>1</sup>Institute of Information Science, Academia Sinica, Taipei, Taiwan, R.O.C.

<sup>2</sup>Department of Computer Science and Information Engineering,  
National Taiwan University, Taipei, Taiwan, R.O.C.

Tel: (02)2788-3799 ext. 1310, Fax: (02)2651-8660  
e-mail:song@iis.sinica.edu.tw

## Abstract

In this paper, we present a real-time eye-tracking system which can be employed by autostereoscopic display systems. An autostereoscopic display system can provide users great enjoyment of stereo visualization without the uncomfortable and inconvenient drawbacks of wearing stereo glasses or Head-Mounted Displays. In order to render stereo video with respect to the user's viewpoints and to accurately project the stereo video onto the user's eyes, the left and right eye positions of the user, who is allowed to move around freely, have to be obtained when the user is watching the autostereoscopic display. The proposed real-time tracking techniques can efficiently provide the position of the user's eye in the images. These techniques comprise of (1) face detection by using an eigenspace-based face detection module, (2) fast template matching for tracking four motion parameters (X and Y translation, scaling, and rotation) of the user's face, and (3) eye detection in the obtained face region. According to our implementation on a PC with a Pentium III 700 MHz CPU, the frame rate of the eye tracking process can achieve 30 Hz.

*Keywords:* autostereoscopic display system, face detection, eigenface, object tracking, template matching

## 1. Introduction

Due to the need for more efficient and friendly human computer interface (HCI), studies on face processing have been rapidly expanding in recent years. One useful component for HCI is the stereoscopic display for providing users with a stereo visual environment. Conventionally, users of a virtual reality system have to wear stereo glasses or Head-Mounted

Displays (HMDs) on their heads, which will make them feel less comfortable.

Recently, researchers began to develop autostereoscopic display systems [1],[2] which can provide great enjoyment of stereo visualization without the requirement of wearing any special devices. In a look-around system, as shown in Fig.1, a user can move around freely in front of the autostereoscopic display to watch the stereo video from various points of views. The eye-tracking component of the autostereoscopic display system is used for tracking the left and right eye positions of the user. Thus, the autostereoscopic display system can render the stereo video with respect to the view points of the user and project the left and right channels of the stereo video to the two eyes of the user respectively. One kind of the tracking methods requires that the user to wear some special sensors, such as infrared sensors or reflectors, ultrasonic wave receivers, and electromagnetic wave sensors. However, these kinds of active sensing methods are uncomfortable and inconvenient. Moreover, these sensors cannot be mounted on the eyes and thus the obtained positions are not the exact eye positions. Therefore, video-based methods are preferable for tracking users in a passive manner. For example, Azarbayejani et al. [3] developed interactive graphics systems; they used a camera to observe a user. Image features of the user can be extracted and can help to track the motion of user's head. The graphics systems can then be controlled by the tracking results. Instead of using features as in [3], another research uses skin-color information for tracking the user's facial region [4], [5], [6]. Shirai [7] also has developed a 3D human tracking system by using optical flow and depth information observed from an active camera head. Recently, Stauffer and Grimson [8]

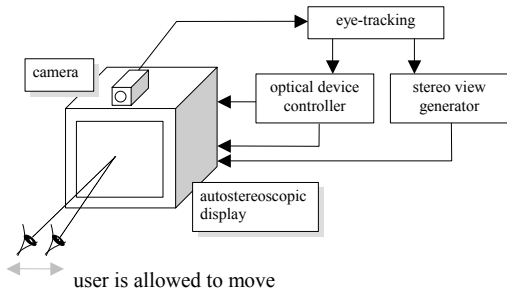


Figure 1: A look-around system.

developed tracking techniques for a visual monitoring system based on an adaptive background subtraction method.

In this work, we develop a real-time eye-tracking system for autostereoscopic display systems. To avoid the drawback of wearing sensors or marks, we use a camera observing a user for tracking his/her eye positions in the acquired image sequence. In the initial stage, we use the eigenface method [10] to detect the face position of the user. The lighting compensation technique is utilized to accommodate various lighting conditions. Then, we employ the template matching technique to track the four motion parameters (X and Y translation, scaling, and rotation) of the user's face in the following images. To meet the real-time requirements, we apply a fast template matching algorithm, called the winner-update algorithm [11], [12], to speed up the tracking process. Once the face position of the user is obtained, the left and right eyes can be located in the face region according to the geometric relation between the face and eyes.

## 2. The eye-tracking system

In this section, we will describe the proposed video-based eye tracking techniques. We will first address some design issues considering the eye tracking for autostereoscopic displays. Next, we will depict the flowchart of the whole eye tracking process. Afterwards, we will describe in detail each component of the proposed eye-tracking techniques.

### A. Design Issues

#### A.1 Consideration of User Behavior

In our eye tracking system, we mount a video camera on top of the display for observing the user. When the user is watching the display, the frontal face of the user should appear in the acquired image. In an

autostereoscopic display system, it is not necessary to keep track of the user's eyes constantly. Instead, the system has to track the user's eyes when the user is *watching* the autostereoscopic display. This will simplify the tracking problem because correct tracking is required only for the frontal face. Moreover, we assume that the user intends to watch the autostereoscopic display in a comfortable way. That is, the user will not look askew at the display on purpose. The user can look at the display at different positions in the 3D space while keeping his/her face toward the display. Sitting in front of the display, the easiest way to change the horizontal viewing position is to rotate the body around the frontal axis of the user. Consequently, the user may also undergo rotation around the normal axis of the image plane in the image. Moreover, the size of the user's face in the image varies with the distance between the user and the camera. Thus, a scaling parameter which describes the size of the user's face in the image has to be updated whenever the user moves nearer, toward or farther away from the display. To sum up, there are four parameters to be estimated, which includes the translations in the X and Y-axes, the scaling, and the rotation around the normal axis of the image.

#### A.2 Tracking Using Template Matching

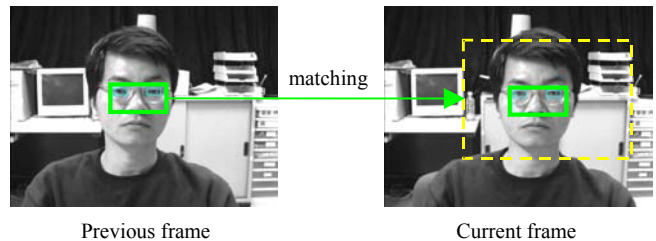


Figure 2: Illustration of face tracking by using template matching.

Template matching technique has been frequently used for visual tracking due to its simplicity and robustness. As shown in Fig.2, the face of the user can be tracked by using the face image in the previous frame as a template image block and matched within a search range in the current frame. However, there are two major disadvantages of the template matching technique. The first one is that its computational cost is so large that its applicability is restricted, particularly for real-time systems. The other disadvantage is that it can only track rigid object undergoing X/Y translation motion. If the object in the image is rotating or changing its scale, the template matching technique may fail to keep the object in track. In order to meet the real-time requirement, we

adopt a fast template matching algorithm, the winner-update algorithm [13], [14], for computational speedup. Moreover, a multilevel conjugate direction search technique (MCDS) is proposed to search and track the template image block, considering its scale and rotation motion parameters in addition to its X/Y translation parameters.

### A.3 Eye Tracking and Face Tracking

The eye positions can be tracked by matching the eye image (as a template image block) in consecutive image frames. However, this tracking method is not robust due to the following two problems. First, the images of the left and right eyes are similar. It is difficult to determine whether the tracked eye position belongs to the left eye or to the right eye. One simple way to solve this problem is to apply the geometric constraint of the left and right eye positions. The second problem is that the eye images are relatively small. Hence, the matching results are more ambiguous because of less information content. In order to overcome these two problems and to increase the accuracy of tracking results, we match the face image instead, which is a larger image block compared to the eye image. The face positions of the user are first tracked in consecutive image frames by using the template matching technique. Once the face region is located, the left and right eye positions can then be estimated in the upper-left and upper-right parts of the face region, respectively. Excluding the mouth, the face region comprises eyes and nose, which are salient features in the face image. Hence, the stability and accuracy can be greatly enhanced by tracking the face first.

### B. Flowchart of the Eye-Tracking System

Fig.3 shows the flowchart of the proposed eye tracking techniques. Initially, we repeat the process of acquiring an image and detecting the face region in the image until the detection succeeds. Once a face region is found, we save the image of this face region as a long-term face template, which is called a face representative. Next, the left and right eye positions can be located within their corresponding areas in the face region obtained either through face detection or through face tracking. Also, the image of the face region obtained in both detection and tracking cases is saved as a short-term face template. (In the following, we refer to the short-term face template as the face template for simplicity.) To track the face position in the following

stage, we use the face template for matching in the coming image. After a new image is acquired, the winner-update algorithm [13], [14] is adopted to match the face template, very efficiently, in a search range of the newly acquired image. Then, the candidate face position having the minimum matching error is verified by using the face representative (long-term face template) to determine whether the image region located at this candidate position actually contains a face. Meanwhile, we also use the MCDS technique to refine the face position.

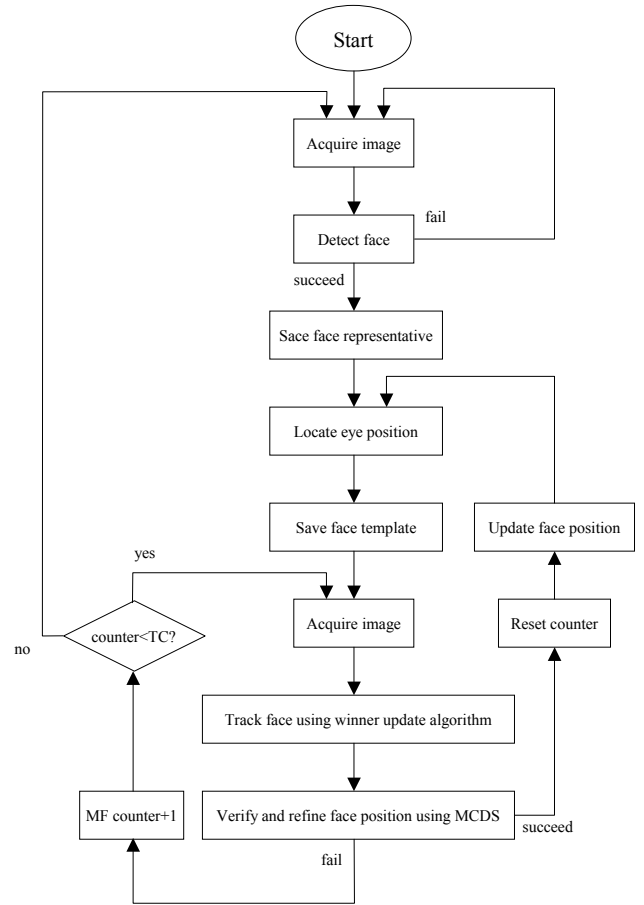


Figure 3: Flowchart of the eye-tracking system.

When the verification succeeds, the eye positions are estimated in the corresponding area of face region. Also, the face template is updated by using the newly obtained image of the face region. This iteration is repeated to track the face and eye positions in consecutive image frames. On the other hand, when the verification fails, we want to recover the tracking process as soon as possible. Hence, we keep the face template without modification and match the face template in a few consecutive image frames. If this recovery also fails, we reset the tracking process by proceeding face

detection and use a new face representative after succeeding in the detection.

### C. Components of the Eye-Tracking System

This section will present in detail each component of the proposed eye tracking technique.

#### C.1 Image Preprocessing

For each acquired image frame, we discard the odd field scanlines and use only the even field scanlines. Also, we perform averaging sub-sampling operation (4:1) for each scanline to smooth the image and reduce the noise. As a result, each 640 x 480 image frame acquired from the camera becomes a 160 x 240 miniature. This downsampling operation can reduce the effect of motion blur and noise. Also, because of the smaller image size, the search region for the template matching is also reduced and this will speed up the tracking process. Besides, we only track the upper part of the face including the eyes and nose because the upper part is relatively more rigid than the lower part of the face, which contains the mouth that may be speaking, laughing, or eating during the tracking process. By using this averaging sub-sampling operation, the rectangular upper face image is reduced to a square block, which can facilitate the winner-update algorithm [13], [14] for fast template matching. In general, the winner-update algorithm does not require square image block. However, it is beneficial to use the square image block due to easier implementation and computation of the winner-update algorithm, which will be briefly described later in Section II-C.3.

#### C.2 Face Detection

Automatic face detection has to be performed when a user appears in the image or when the tracking process has to be restarted because the user has been out of track for a certain period of time. In this work, we adopt a framework of multi-resolution and focus-of-attention. The face detection module is divided into two stages: the candidate selection stage and the candidate verification stage. In the candidate selection stage, we first select some face candidates which are more likely to be face regions in lower resolution using eigenface [10]. Then these face candidates are verified in higher resolution using eigencomponents, such as eigeneye and eigennose, in the candidate verification stage.

A general eigenspace-based method for face detection works as follows: first a set of training face

images are collected; these training images can form a matrix with each row representing the pixel values of a training face image. The eigenspace of smaller dimension can then be obtained by using principal component analysis. For the image block (with the same size as the training face images) at each position in the acquired image, its distance to the eigenspace is calculated to determine whether this image block contains a face image. In this work, if the distance from the image block under examination to the eigenspace is small enough, we can conclude that this image block may contain a face; this image block is labeled as a face candidate and subjected to further verification.

One of the major difficulties of the eigenface method is the lighting variation problem. If the image under examination is acquired in a lighting condition that is very different from the lighting condition in which the training images of the eigenspace are acquired, the distance from the image block under examination to the eigenspace will be very large, even if it actually contains a face. Thus we adopt a lighting compensation method containing plane fitting and histogram equalization techniques to reduce the affection by lighting variation. For each image block, before being subjected to principal component analysis, we can find a linear plane which best fit the image block; we subtract the linear plane from the image block and then perform histogram equalization on the residual image block. This lighting compensation preprocessing is performed on each image block which is then subjected to principal component analysis, during both the training stage and the detecting stage.

As mentioned above, some face candidates which are more likely to be face regions are selected in the candidate selection stage. After that stage, there may be more than one face candidates with different scales, and located at various positions. Before the verification stage, face candidates of the same scale will be merged if they are close enough, and the centroid will be calculated as the new position of the new block. Each block after merging will then be subjected to verification and assigned an evidence score. If blocks are overlapping, the one with the lower score will be discarded.

As shown in Fig.4 (b), there may be more than one face candidate after the merging operation. We perform a component-wise verification on each merged face candidate.

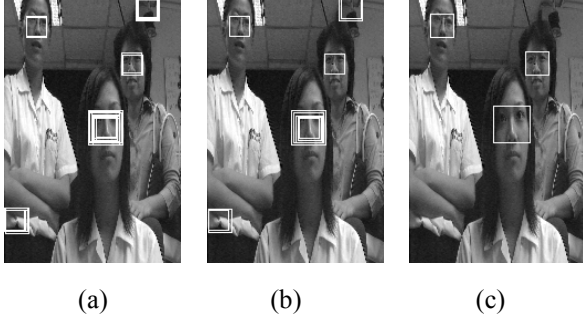


Figure 4: Illustration of candidate merging. (a) All face candidates without merging. (b) Face candidate after merging. (c) The final result verified as faces.

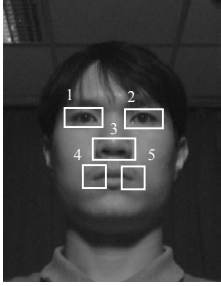


Figure 5: Illustration of the facial components. The number is the index of the component.

Fig.5 shows the facial components that we use. For each components, we collect some train images and calculate the eigenspace with the standard PCA mentioned before. The obtained eigenvectors are called eigencomponents. We use these eigencomponents to detect the facial features respectively in the

corresponding area of the face candidate; a verification score  $SCR_x$  is given to the face candidate  $x$  according to the following equation:

$$SCR_x = \sum_{i=1}^5 (W_i \times C_i)$$

where  $W_i$  is the weighting of the  $i^{th}$  component.

$$W_i = \begin{cases} 12 & i < 3 \\ 10 & i = 3 \\ 5 & i > 3 \end{cases}$$

Let  $b_i$  be the detected block of the  $i^{th}$  component and  $MDIS(b_i)$  be the Mahalanobis distance of  $b_i$ . The confidence ratio  $C_i$  of block  $b_i$  is defined according to the error ratio  $E_i$ .

$$C_i = \begin{cases} 0 & \text{if } E_j > 1 \\ 1 & \text{if } E_j < 0.1 \\ 1 + \frac{E_i - 0.1}{0.1 - 1.25} & \text{otherwise} \end{cases}$$

$$E_i = \frac{MDIS(b_i)}{Threshold_{MDIS}^i}$$


Due to the existence of different expressions and structural components such as facial hair, the corners of mouth are relatively less robust than other facial features. We treat them as bonus to help verifying those candidates in which an eye or the nose is failed to be found.

For face candidate block  $x$ , if the verification score  $SCR_x$  is smaller than the threshold  $Threshold_{SCR}$ , we claim that  $x$  is not human face;  $Threshold_{SCR}$  is defined as follows:

$$Threshold_{SCR} = \left( \sum_{i=1}^3 MS_i \right) \times 0.6$$

For those blocks with  $SCR$  larger than  $Threshold_{SCR}$ , there are two more criteria to satisfy: first, the arrangement of facial components must be symmetric enough; second, the intensity of eyes must be dark enough. These two criteria is formulated as follows:

Let  $D_{ij}$  be the distance from the center of the  $i^{th}$  component to the center of the  $j^{th}$  component in candidate block  $x$ . We say that  $x$  is symmetric enough if the following stands:



$$\begin{cases} \frac{\min(D_{1,3}, D_{2,3})}{\max(D_{1,3}, D_{2,3})} \geq 0.8 \\ \frac{\min(D_{4,3}, D_{5,3})}{\max(D_{4,3}, D_{5,3})} \geq 0.8 \end{cases}$$

Let  $AI\_upper_x$  be the average intensity of the upper part of face candidate block  $x$  (namely, the average intensity of upper face) and  $AI_{x,i}$  be the average intensity of the  $i^{th}$  component; we say that the eyes are dark enough if the following stands:

$$\begin{cases} AI\_upper_x \geq AI_{x,1} \\ AI\_upper_x \geq AI_{x,2} \end{cases}$$

Candidate blocks meet all criteria described above will be claimed as human face in the long run. However, the eye-tracking system require only one face, thus it only tracks the face with the maximum verification score.

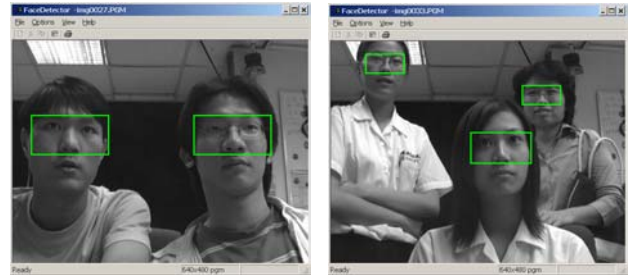


Figure 7: Examples of face detection results.

### C.3 Face Tracking

For each image frame, the face image block that is successfully tracked is stored as the face template. This face template will be used for template matching within a search range in the consecutive image. The matching error criterion we used is the Sum of Absolute Difference (SAD); the matching error between the template and each block  $I_t(x+u, y+v)$ , within the search range in current frame  $I_t$ , is calculated, and the one with the minimum matching error will be the new template and used to match in  $I_{t+1}$ .

$$\text{SAD}_{(x,y)}(u, v) = \sum_{i=0}^{B-1} \sum_{j=0}^{B-1} |I_{t-1}(x+i, y+j) - I_t(x+u+i, y+v+j)|$$

$$(\tilde{u}, \tilde{v}) \equiv \arg \min_{(u,v) \in \{(u,v) | -R \leq u, v \leq R\}} \text{SAD}_{(x,y)}(u, v)$$

In order to achieve high accuracy, we adopt the winner-update algorithm [11],[12] which guarantee that the global minimum of the template matching can be found efficiently. This algorithm first constructs a block sum pyramid (BSP) [13] for the template image block and for each image block in the search region. The SAD calculated by using the upper levels of the corresponding pyramid pairs can be proved [13] to be the lower bound of the desired SAD calculated by using the bottom levels, which contain the original images of the template image block and the image block in the search region. For a search position, once one of its lower bound of SAD calculated at some upper level is already larger than the global minimum SAD, this search position can be skipped for further consideration. The computational cost of the SAD lower bound is smaller than that of the SAD itself because the lower bound is calculated at upper level of the pyramid. Moreover, we reduce as many possible SAD lower bounds actually calculated by using the winner-update search strategy [11], [12]. This search strategy initially calculates the first SAD lower bounds for all the search positions by using their top levels of pyramids. The one having the minimum lower bound is selected as the temporary winner and its next lower bound is calculated by using its lower level. Then, the temporary winner is selected again among all the search positions. This process is repeated until the minimum SAD lower bound of the newly selected temporary winner is calculated at the bottom level, that is, the SAD lower bound is the desired SAD. In [13], [14], we also proposed an efficient method, in terms of computation and storage, of constructing pyramids for all the search positions in the search region. It is beneficial to use

square image block due to the simplicity of pyramid construction and SAD lower bound calculation.

### C.4 Face Verification and Position Refinement

For the candidate face position yielding the minimum matching error, the goal of face verification is to determine whether the image block at this position actually contains a face. Two criteria can be used for making this decision. The first one is that the image block at this candidate face position should “looks” similar to the face template. The second one is that this image block should “looks” similar to the detected face representatives. The latter criterion is necessary because the tracking error may accumulate during the tracking process. For example, if A is similar to B and B is similar to C, A may not be similar to C. That is, the face position tracked may have drifted away after a period of time. Consequently, we make use of the face representative, which is a long-term face template, to perform the verification of the second criterion while refining the face position. Because the user is allowed to rotate his/her head and to move nearer or farther away from the display, the rotation and the scaling of the candidate face position may be different from those of the face representative, which is recorded when the face is detected. To deal with the scaling and rotation of the face image block, we compute various combinations of scaling and rotation for the face representative in advance. Multilevel conjugate direction search technique is then used to search for the combination such that the corresponding face representative is similar to the candidate image block.

When the face region is detected, not only the original face image is stored as the face representative but also the face images which are obtained by rotating and scaling up and down the original face image block. Fig.8 illustrates an example comprising of 25 face representatives of five different rotations and five different scales. A block sum pyramid structure is then constructed for each face representative.

During the matching process, two dimensions of the scaling and rotation can be searched by using the one-at-a-time search, which is a simplified conjugate direction search [16], on each level of image pyramids. On the top level, we find the combination of scaling and rotation yielding the local minimum of SAD by using the following process. Started from the initial combination of scaling and rotation, we move along the scaling direction



Figure 8: Face representatives comprising various combinations of rotation (horizontal direction) and scaling (vertical direction).

first and find the local minimum of SAD by using the gradient descent search. Then, starting from this local minimum, we move along the rotation direction to find the new local minimum of SAD by using the gradient descent search again. The combination of scaling and rotation yielding the new local minimum is used as the initial combination for the next level. By repeating such one-at-a-time search from the top to the bottom level, we can obtain the combination of scaling and rotation yielding the local minimum of SAD at the bottom level. The reasons that we utilize the pyramid structures are included in the following: (1) it is more efficient; and (2) we have more chance to bypass the local minimum to achieve the global minimum due to the smoother error surface on the upper levels, which contain smoother images.

Because of the accumulation error, the matched face position may have drifted away from the correct face position. To improve the accuracy, we utilize the similar conjugate direction search technique to refine the face position along the gradient directions in the X and Y dimensions, one at a time. Along the search path, the matching error using the face representatives are calculated. The position yielding the minimum matching error is reported to be the refined face position. According to our experimental results, this method can effectively overcome the drifting problem.

### C.5 Eye Location

Once the face position is determined, we can locate the left and right eye positions within the face region. Instead of feature-based methods [14], we adopted a convolution-based method for simplicity and efficiency. After obtaining the position of the user's face, we divide the face image block into four parts and perform eye detection in both upper-left and upper-right sub-blocks by using the convolution operation with an 8x8 mask, as shown in Fig.9. The position with the smallest convolution result, which means the darkest region, is considered as the eye position.

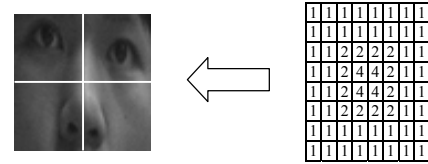


Figure 9: Illustration of eye detection.

### C.6 Temporary Out-of-Track

When the verification of the candidate face position fails, the face position is out of track and the eye-tracking system has to recover from the failure. This situation occurs because of temporary occlusion. For example, the user waves his hand across his face. One way of recovery is to detect the face position all over again. However, this process is more time-consuming and should not be performed frequently. Instead of performing face detection, one can try to recover face tracking by simply acquiring the next image and using the face template to match again for a pre-specified period of time. This will be helpful when the user temporarily turns his/her face away or his/her face is temporarily occluded. The eye tracking system can recover from tracking failure faster in this way.

## 3. Experiments

The proposed eye-tracking system is deployed on a PC with Pentium® IV 1.2 Ghz CPU running Microsoft Windows® 2000. The video frame rate of the video acquiring equipments is 30 Hz, which will limit the maximum frame rate and the minimum latency time of the eye-tracking system. In our implementation, the tracking process and video acquisition proceed concurrently. The tracking process for each frame can be finished in less than 1/30 second; consequently, the overall frame rate of our eye tracking system is 30 Hz, that is, the video frame rate, and the latency time ranges from 1/30 to 2/30 second. Besides the eye-tracking system using a single camera, we also build a stereo eye-tracking system which can provide the position of user's eyes in 3D space. With good calibration, the error of 3D position is within 3% and the frame rate can still achieve 25 fps.

## 4. Conclusions

We have presented video-based eye tracking techniques that can accurately, robustly, and efficiently track the user's face and eye positions in video sequence. When combining with an autostereoscopic system, the resulting look-around system can provide the user stereo video without the requirement of wearing any special



glasses or sensors.

The system first detects the existence of the frontal face in the image and then tracks the face image block in real-time with the winner-update algorithm. Besides using a single camera, we have also deployed a stereo eye-tracking system which can provide precise 3D positions of the user's eyes.



Figure 10: The stereo eye-tracking system.

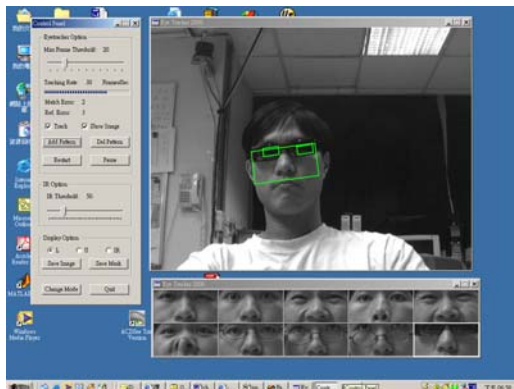


Figure 11: Result of detecting and tracking faces with various expressions.

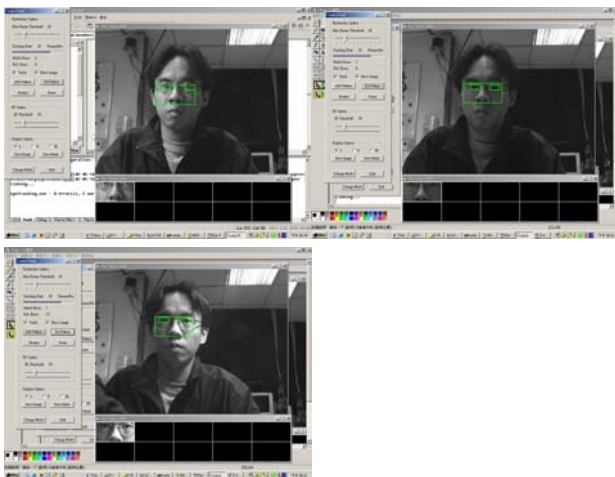


Figure 12: Result of detecting and tracking faces under different lighting conditions.

## Acknowledgements

This work is partially supported by Opto-Electronics & Systems Laboratories, Industrial Technology Research Institute, Hsinchu, Taiwan.

## References

- [1] G. Woodgate, D. Ezra, J. Harrold, N. Holliman, G. Jones, and R. Moseley, "Autostereoscopic 3D display systems with observer tracking," *Signal Processing: Image Communication*, 1998.
- [2] S. Pastoor, J. Liu, and S. Renault, "An experimental multimedia system allowing 3-D visualization and eye-controlled interaction without user-worn devices," *IEEE Transactions on Multimedia*, 1999.
- [3] A. Azarbayejani, T. Starner, B. Horowitz, and A. Pentland, "Visually controlled graphics," *IEEE Transactions on PAMI*, 1993.
- [4] P. Fieguth and D. Terzopoulos, "Color-based tracking of heads and other mobile objects at video frame rates," in *Proceedings of the IEEE CVPR*, Puerto Rico, 1997.
- [5] N. Herodotou, K. N. Plataniotis, and A.N. Venetsanopoulos, "Automatic location and tracking of the facial region in color video sequences," *Signal Processing: Image Communication*, 1999.
- [6] T. Darrell, G. Gordon, M. Harville, and J. Woodfill, "Integrated person tracking using stereo, color, and pattern detection," *IJCV*, 2000.
- [7] Y. Shirai, "Estimation of 3-D pose and shape from a monocular image sequence and realtime human tracking," in *Proceedings of the International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, 1997.
- [8] C. Stauffer and W. Eric L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Transactions on PAMI*, 2000.
- [9] K. Talmi and J. Liu, "Eye and gaze tracking for visually controlled interactive stereoscopic displays," *Signal Processing: Image Communication*, 1999.
- [10] B. Moghaddam and A. Pentland, "Probabilistic visual learning for object representation," *IEEE Transactions on PAMI*, 1997.
- [11] Yong-Sheng Chen, Yi-Ping Hung, and Chiou-Shann Fuh, "A fast block matching algorithm based on the winner-update strategy," in *Proceedings of the ACCV*, Taipei, 2000.
- [12] Yong-Sheng Chen, Yi-Ping Hung, and Chiou-Shann Fuh, "Fast block matching algorithm based on the winner-update strategy," *IEEE Transactions on Image Processing*, Accepted at Mar. 15, 2001.
- [13] C.-H. Lee and L.-H. Chen, "A fast motion estimation algorithm based on the block sum pyramid," *IEEE Transactions on IP*, 1997.
- [14] K.-M. Lam and H. Yan, "Locating and extracting the eye in human face images," *Pattern Recognition*, 1996.