Blink Detection for Real-Time Eye Tracking

Abstract

This work is motivated by our goal of providing non-contact head and eye based control of computer systems for people with motor difficulties. The system described here uses spatio-temporal filtering and variance maps to locate the head and find the eye-feature points respectively. These feature points are accurately tracked in the succeeding frames by using a modified version of the Lucas-Kanade tracking algorithm with pyramidal implementation. Accurate head and eye tracking results are obtained at a processing rate of more than 30 fps in more than 90% cases with a low false positive blink detection rate of 0.01%. This is achieved under varying lighting conditions for people of different ethnicity, with and without wearing glasses.

1 Introduction

For many people with physical disabilities, computers form an essential tool for communication, environmental control, education and entertainment. However access to the computer may be made more difficult by a person's disability. A number of users employ head-operated mice or joysticks in order to interact with a computer and to type with the aid of an on-screen keyboard. Head-operated mice can be expensive. In the UK, devices that require the users to wear no equipment on their heads, other than an infrared reflective dot, for example Origin Instrument's HeadMouse [16] and Prentke Romish's HeadMaster Plus [17], cost in excess of £1,000. Other devices are cheaper, notably Granada Learning's Head Mouse [4, 5, 6] (£200), and Penny and Gilles' device (£400). However, these systems require the user to wear a relatively complex piece of equipment on their head, an infrared transmitter and a set of mercury tilt switches respectively.

The goal of this research is to develop and evaluate a non-contact interface that is both low cost and also does not require the user to wear any equipment. The most obvious way of doing this is to use a camera, interfaced to a PC. The PC/camera system will track the movement of the user; it could translate head movements into cursor movements, or interpret changes in expression as button presses.

There have been many attempts at creating cursor control systems utilising head movements; these are reviewed below. The head monitoring system must also provide a mechanism for replacing the mouse buttons. We propose that this can be achieved by using the blinks of the eyes. Changes in expression could also be used as part of a separate device to provide a computer interface for more severely disabled persons.

A system such as the one proposed will confer significant benefits on its intended users. It will increase their independence by facilitating their communication with a computer and hence providing an avenue for communication and environmental control. The proposed interface may also have an application in the non-contact control of remotely controlled units for the able bodied; providing a perceptual control mechanism for such a unit will free the operator to concentrate on other demanding tasks.

The implementation of a system such as the proposed one presents several areas of difficulty:

1. Identifying and tracking the head location.
2. Identifying and tracking the location of the eyes.
3. Detecting blinks of the eyes.
4. Being able to process the information in real-time using a moderately priced processor that will be running other applications in the foreground (for example, Microsoft Word).

Most eye trackers require manual initialisation of the eye location before they can accurately track eye-features for real-time applications. A method for locating the eyes in static images was

developed by Kanade in 1973 and has been improved by other people over the years [2, 3, 8, 12, 14, 15]. Most of these researchers have based their methods on Yuille's deformable templates to locate and track eye-features [15]. This method looks for the valleys and peaks in the image intensity to search for the eyes. Once the location of the eyes is determined, its position information is used as a priori knowledge to track the eyes in succeeding frames. But it requires the eye template to be initialised manually at or below the eye otherwise it detects the eyebrow instead of the eye [12]. Hallinan [7] has tried to build an automated model for deformable templates to find the best candidates for the eye pair, but in order to make his method invariant to scaling, the template is initialised at different sizes at various places and the best candidates for the eyes are selected. Chow et al. [2] make use of the Hough transform in combination with the deformable templates to extract eye-feature points, but this approach is also time consuming as the Hough Transform for various scales had to be applied prior to detecting the iris, unless the approximate radius of the iris is known in advance. Deng and Lai [3] presented the concept of regional forces and regional torque to accurately locate and resize the template on the iris even when the iris is in an extreme position, and for the correct orientation of the template. But their method also requires hand initialisation to fix the position of the eye window before it can successfully locate and track the eyelids. All these methods track the eyes from frame to frame by readjusting the template of both the iris and the eye contour. Tian et al. [12] have shown that such an approach is prone to error if the eyes blink in the image sequence.

This paper describes a method that can be employed for automatic initialisation of the position of the eyes, which can then be used for any eye-tracking application. It is based on Turk's method [13] for the detection of the eyes in an image sequence, which uses simple spatio-temporal filtering to locate the position of the head of a person in an image sequence. The located position of the head is continuously tracked and variance maps are used simultaneously to identify any possible eye-blink pair candidates, within the region of the detected head. Once, an eye-blink pair is shown to exist at a known location in the frame of an image sequence, the contours around it are adjusted and four feature points for each eye are recorded (i.e. the outer and inner corners and the centre of the upper and lower eyelids). These feature points are then accurately tracked in the remaining frames of the sequence by employing the modified version of the Lucas-Kanade tracking algorithm [9], which involves its pyramidal implementation as described by Tian et al [12].

The primary contribution of this paper is a novel method of automatically initialising the eyes' locations in an image sequence for real time eye tracking applications. It also calculates the size of the eyes, which then allows the possibility of the computation of other anthropometric measurements for the approximate location of other facial features. A vision system is described that dynamically updates the new location of the head and eye-feature points for other vision applications such as driver blink rate monitoring, lie detection by measuring blink detection patterns, eye-gaze input for human computer interaction, etc. In this evaluation system we have assumed that a single user is present and they are presenting a head and shoulders view to the camera.

This paper is organised as follows. First, a method for locating the approximate position of the head is summarised in Section 2. In Section 3, a method for the automatic initialisation and adjustment of the eye-feature points is described. Section 4 presents the eye tracking procedure. In Section 5, sample results are presented, in which the eyes are automatically detected and tracked. The paper concludes with a critical evaluation of the work and suggestions for improvements in section 6.

2 Face Tracking

In this section a face tracking method is described whose goal is to obtain the approximate bounding box around the face. This bounding box serves to restrict the region of the image that is searched for the eyes. The method is based on the simple spatio-temporal filtering algorithm presented by Turk [13]. A user does not sit motionless in front of a computer for a long duration, therefore, it is feasible to detect motion over a limited spatial and temporal region and identify this motion with the user's head.

Frame differencing is primarily used for head segmentation. In an image containing a single user, where there is no background disturbance (such as variation in the lighting conditions or movement of other people or objects) and only the head and shoulders of a user are visible, this method can locate the head position quite accurately. This is because the lateral movement of shoulders is much smaller than the movement of the head. Also, users constantly adjust their gaze direction and unwittingly move their heads, which is sufficient for motion-based head detection.

The steps for motion-based head segmentation are as follows. (Grey scale images are used; colour information would add to the volume of data to be processed, but would not add any knowledge.) First a reference image, at index i, from an image sequence is obtained. The next N images, from (i + 1) to (i + N), are subtracted from the reference image on a pixel-by-pixel basis, so that N difference images are obtained. These difference images contain the absolute difference in pixel intensity values from the reference image. The difference images are all added together to obtain the accumulated difference frame ADF, this is a grey scale image that is restricted to 8 bits. The accumulated difference frame is thresholded at a suitable value to remove noise and the differences due to insignificant movements. For this work a threshold, τ, of 255 is used, which is the maximum absolute value for an 8-bit pixel. The values of pixels in ADF range from 0 to 255. This is described in Equation (1)

$$ADFT(x,y) = \begin{cases} 255 & if \;\; ADF(x,y) \geq 255 \\ 0 & otherwise \end{cases} \tag{1}$$

Equation 1 describes the process of thresholding on a pixel at coordinates (x, y). The number of pixels in the ADFT image with the value of 255 is counted. If the ratio of this number to the total number of pixels in the image is greater than a threshold, θ, then motion analysis is performed otherwise a new reference image at index (i+N+1) is obtained. For this work a value of $\theta = 0.015$ is used.

$$\frac{\text{Number of Thresholded motion Pixels}}{\text{Total number of Pixels}} > \theta \tag{2}$$

In a head and shoulders image, such as is processed in this application, most of the motion shown in the ADFT is due to head movement, but there may also be some small regions which are due to non-head movements or to having incorrect values for the number of difference frames N or the threshold θ. Assuming that these parameters are approximately correct, the goal of the first phase of motion analysis is to remove extraneous motion regions. Since head movement is the dominant motion, small regions (of size less than 16 x 16 pixels) are assumed to be due to other movements and are deleted. The remaining region(s) is due to head movement. The second phase of motion analysis will identify a region of the image within which the eyes are probably located. The centre of mass of the remaining region in the ADFT will correspond to the centroid of the head. A square bounding box is drawn of length 120 pixels and centred on the centre of mass. It has been found in 95% of the cases processed that this box includes the subject's eyes. The bounding box is therefore used to define a search region of the image. The centroid is reevaluated every N frames.

(a) (b) (c)

Figure 1. Motion-Based Head Segmentation a) Accumulated difference frame b) Thresholded accumulated difference frame c) Bounding box drawn around centre of the mass of the threshold image.

3 Blink Detection

The approach employed in this paper is based on the blink detection algorithm presented by Turk [13]. This process is used to initialise the location of the eyes in the image and it provides the position of the feature points for eye tracking. A user will blink involuntarily (with both eyes) at intervals. Providing that head movement is restricted, the blinks may be detected via a variance map. The steps are as follows. Only the region of the image defined by the motion analysis is processed. This yields a significant improvement in throughput.

The first frame of the sequence, of M×N pixels, is obtained (colour images are captured, but they are processed as grey scale data). The variance map, σ^2_1, of M×N pixels, is initialised to zero, it consists of 8-bit pixels. Thus:

$$\sigma_1^2(x,y) = 0 \tag{3}$$

The mean image, μ, of size M×N pixels and 24 bit pixel depth is initialised by assigning the pixels of the first image to its corresponding pixels, so that

$$\mu_1(x,y) = I_1(x,y) \tag{4}$$

The next frame, I_2, is obtained and used to update the mean image μ and variance map σ^2 according to the recursive formulae given in Equation (5). Thus the mean image and variance map reflect the current knowledge of data.

$$\begin{aligned} \mu_{j+1}(x,y) &= \frac{j\mu_j(x,y) + I_{j+1}(x,y)}{j+1} \\ \sigma^2_{j+1}(x,y) &= \left(1 - \frac{1}{j}\right)\sigma_j^2(x,y) + (j+1)\left(\mu_{j+1} - \mu_j\right)^2 \end{aligned} \tag{5}$$

After updating it with the new frame, a thresholded version of the variance map is computed. The threshold value used in the case of this work is 255. The number of pixels within the bounding box that remain after the thresholding operation is counted. If the number is large in comparison to the size of the bounding box, then we assume that a blink has been discovered. This decision is made by comparing the ratio of the number of thresholded pixels to the number of pixels within the bounding box:

$$\frac{\text{Number of thresholded variance pixels}}{\text{Total number of pixels in the bounding box}} > \vartheta \tag{6}$$

Simple statistical operations are performed on each connected component, i.e. the area, the centroid, and the minimum and maximum dimensions are determined. An eye-blink region is detected by the elimination of the connected components that are too large (more than 200 pixels) or too small (fewer than 50 pixels) or too far from circular (with a ratio of horizontal to vertical dimension greater than 2.0 or less than 0.5). To be considered an eye-blink pair, the two regions are required to be at approximately the same horizontal level, and their horizontal separation must be within a certain range.

Finally, if more than two regions qualify as eye-blink pairs, then the best two are selected on the basis of their similarities. The inner and outer corners of the eye-blink pair are recorded and adjusted so that the size of both the eyes (i.e. the width and the height) becomes the same. It is assumed that a straight line joins the four corners of the two eyes. Therefore, these four points are brought in line by measuring the slope of the line joining the two outer points and adjusting the positions of the inner points as shown in Figure 2.

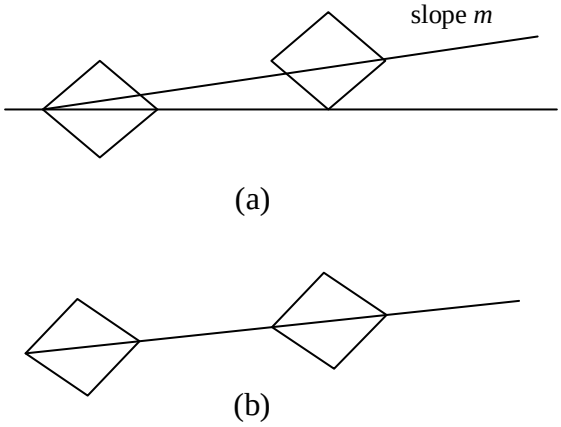


Figure 2. a) Detected corners in the eye-blink pair
b) Corners adjusted according to slope

The adjusted corners are passed ahead for eye tracking as the feature-points. A total of eight points are passed, which include four from each eye. These four points are the outer corner, the inner corner, the topmost point and the bottom most point. The process is shown in Figure 3.

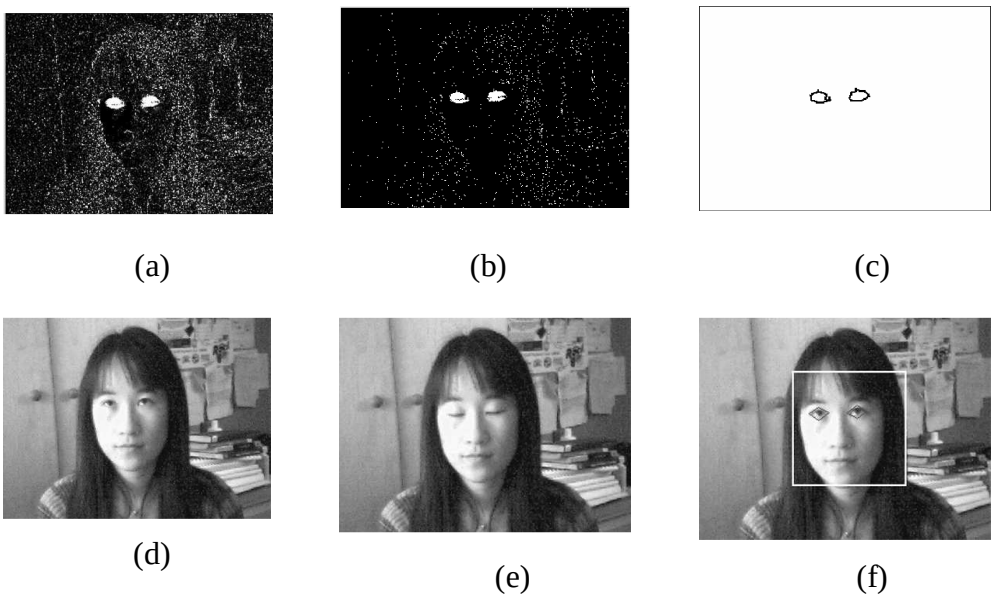


Figure 3. Blink Detection a) Variance map b) Thresholded variance map c) Contour of the eye-blink pair d) Selected frame from a sequence of 10 images showing eyes open e) frame from the same sequence showing eyes closed f) Detected position of the head and the eyes

4 Eye Tracking

This method is based on the Lucas-Kanade feature tracker, which tracks feature points to sub-pixel accuracy by optical flow [9], with a pyramidal image representation as given by Bouguet [1]. Tian et al. [11] have observed that the inner corner of the eyes are almost invariant to lighting changes and therefore provide the best results in feature tracking. They have successfully employed a similar algorithm for eye-feature point tracking as given by Poelman [10]. This is done by minimising a residual function $\varepsilon(x, y)$ based on the image intensity, $I(x, y)$, at a point (x, y).

$$\varepsilon = \left(I(x,y) - I(x+dx, y+dy)\right)^2 \tag{7}$$

Because of the aperture problem a window larger than a single pixel has to be employed to determine the motion of a pixel from one frame to the next. This larger window is the neighbourhood region R(**x)** of the point **x**. This becomes the problem of minimisation of the residual function ε for the entire region R. The residual is expressed as:

$$\varepsilon(d\mathbf{x}) = \int_{\mathbf{x}\in R(\mathbf{x})} (I(\mathbf{x}) - (I(\mathbf{x}+d\mathbf{x}))^2 \tag{8}$$

In discrete terms, when the width of the integration window is ($2w_x + 1$) and its height is ($2w_y + 1$) as shown in Figure 4, the expression for the calculation of the residual function ε is:

$$\varepsilon = \sum_{x'=x-w_x}^{x+w_x} \sum_{y'=y-w_y}^{y+w_y} (I(x',y') - I(x'+dx, y'+dy))^2 \tag{9}$$

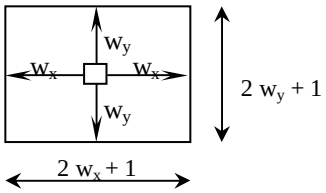


Figure 4. 2D Neighbourhood of pixel (x , y) used to minimise residual function ε

The best match between the features in successive pairs of images is determined by finding the values of dx and dy that minimize the residual. This is achieved using the published methods.

5 Performance Evaluation

In order to measure the validity of the design qualitatively and quantitatively, a prototype was implemented on a Pentium III 450 MHz Intel Microprocessor with 128 MB RAM running under the Windows 98 operating system. A Logitech Viewcam USB was employed to capture sequences of colour images at the rate of 30 frames/s. The resolution of each captured image was 320×240 pixels with a pixel depth of 24 bits (eight bits in each red, green and blue channel). The data set obtained for evaluation consisted of sequences from five people of different ethnic origin: one European, two Chinese and two South Asians. The subjects were asked to move as they would normally when using a personal computer. Two different types of light sources were used in the data set: sunlight from a window and lamplight from a tungsten filament bulb. The contrast between the brighter and darker side of the face was regulated by using a fill-in light of varying intensity for the less illuminated part to ensure fair representation of different possibilities.

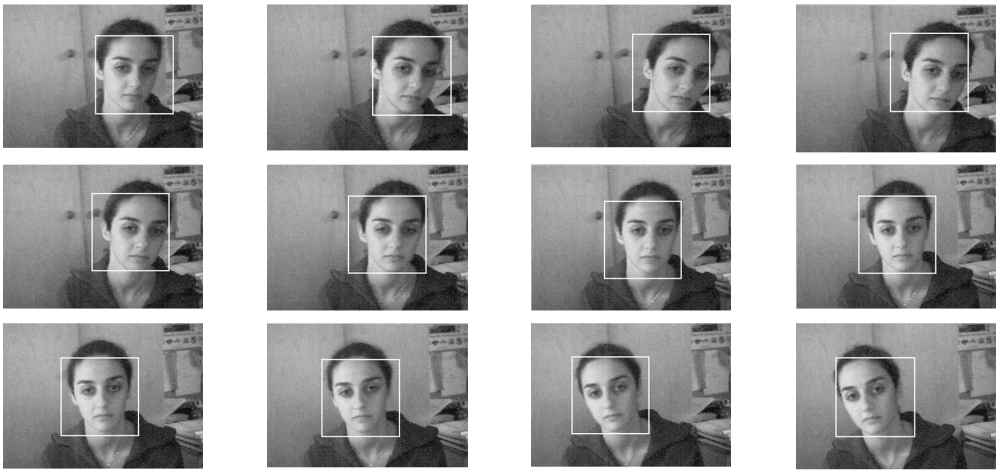


Figure 5. Motion-Based Head Tracking. Selected frames from the image sequence after the interval of 2/3 of a second are shown. The frame number increases from left to right and top to bottom.

Sample head tracking results are shown in Figure 5. The performance with respect to speed is good as a face is tracked from a stored image sequence with a frame rate of 50 frames/second. This allows the possibility of using it as a pre-processing step for a real-time blink detection system.

Figure 6. Blink-Detection. (The head is steady and eyes are visible clearly through the glasses). Selected frames from an image sequence at the interval of 1/15 of a second are shown. The appearance of the red border around the eyes in the last frame indicates the detection of the blink of the eye-pair. The frame numbers increase from left to right.

The results for the image sequence, where a user's head remains steady while blinking are shown in Figure 6. It was observed that this method worked equally well for subjects of all racial origins. The computational cost for generating the variance map and finding a proper eye blink pair was approximately 15 frames/s, (i.e. a processing rate of 35 fps was achieved) which was acceptable in this blink detection process.

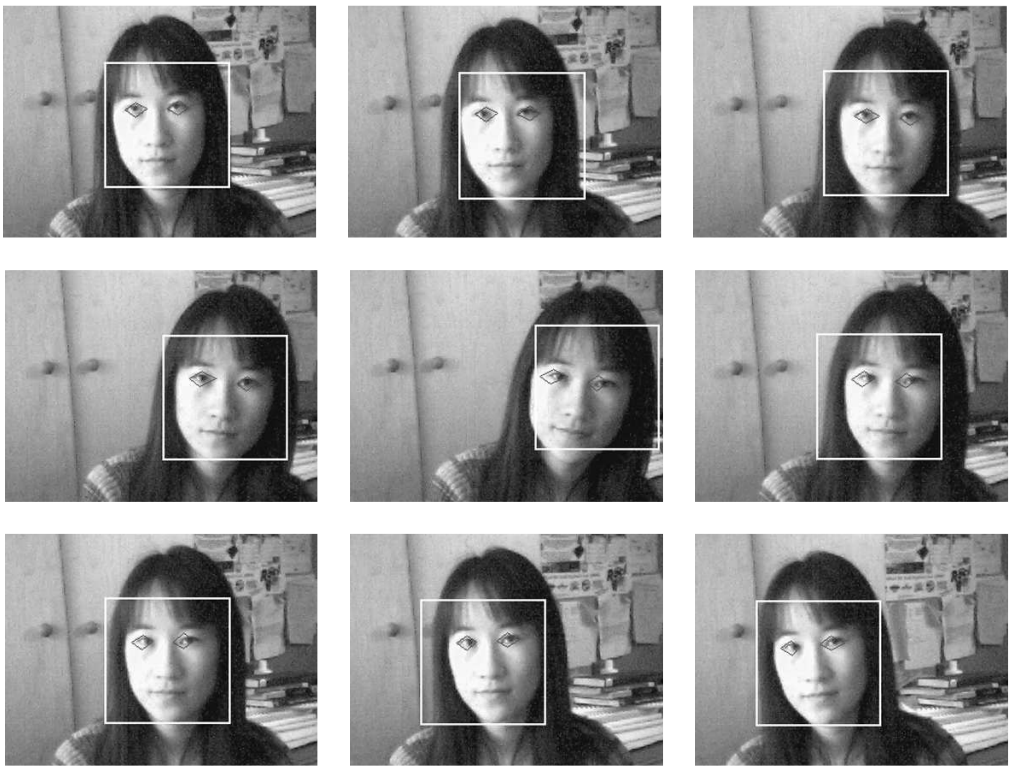
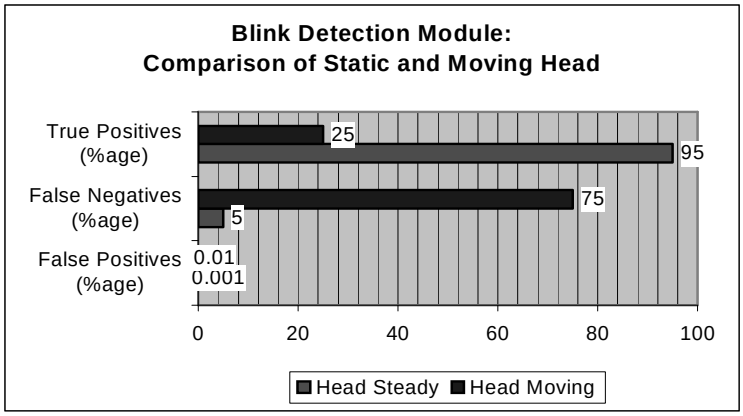


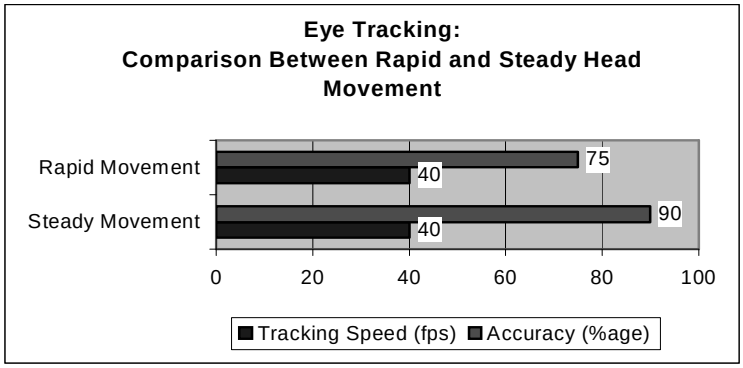
Figure 7. Eye-tracking and Motion-Based Head Tracking. Eyes are tracked after the detection of the eye-pair blink. (The head is moving steadily, tilting sideways and no forward or backward head movement). Selected frames from an image sequence at the interval of 1 second are shown. The frame numbers increase from left to right and from top to bottom. Every feature point of each eye (top, bottom, left and right) is tracked independent of the other three points of that eye and head is tracked independent of the eye tracking module.

Figure 7 shows the results obtained when processing an image sequence where the user's head gradually moves sideways. Eyes are tracked quite accurately in almost every frame of the sequence. Similar results are obtained for other image sequences, where the blink is detected properly and the eye-feature points are extracted. The computational cost of this method is quite low as only a 25×25 window is used for computing the optical flow of these feature points. The processing speed exceeds 30 frames/second for tracking 8 eye-feature points. A decline in eye-tracking accuracy was observed when the capturing frame rate of the camera was reduced below 30 frames/sec. The system developed for this paper was tested for slight change in orientations of the user's head and works well as long as both the eyes are approximately horizontal and visible in the image, these are not unreasonable requirements for computer users.

The bar graphs in Figure 8 indicate the performance of the head tracking, blink detection and eye-tracking modules we have developed.



(a)



(b)

Figure 8. Comparison Charts a) Blink Detection Module b) Eye Tracking Module

6 Conclusion

The success of the system in the constrained conditions as mentioned above shows that blink detection is a viable technique for automatically initialising the eyes' locations. We have demonstrated that eye-feature points obtained from the blink detection process can be successfully tracked in the succeeding frames of an image sequence. The system has worked in real-time and is robust with respect to variations in scaling and lighting conditions, different orientations of the head and presence of distracting objects on the face (such as glasses etc.), and for subjects of different racial origin.

However, the system has limitations, primarily because it is intended to track the eyes in near-frontal head postures. If the head is rotated more than 45^0 to either side, about a vertical or horizontal axis, the systems will lose track of one eye, but it will continue to track some feature. This will cause tracking of false feature points when the head pose once again becomes more upright. A process is required which can identify the loss to tracking of an eye caused by the head rotation and estimate the missing eye location correctly until it becomes visible again. This is the type of movement that will occur, for example, when a user is momentarily distracted from the on-screen activity.

The system also assumes a single user is present in the frame of an image sequence. This limits the possibilities of its application. Future work will need to cope with the presence of multiple faces and multiple users, which can allow multi-user gaze input for applications such as multi-user computer games etc.

References

[1] Jean-Yves Bouguet, “Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm”, Intel Corporation, Microprocessor Research Labs, OpenCV Documents, 1999.

[2] G. Chow and X. Li, “Towards a system for automatic facial feature detection”, Pattern Recognition, Vol. 26, pp. 1739-1755, 1993.

[3] Jyh-Yuan Deng, Feipei Lai, “Region-Based Template Deformation and Masking for Eye-Feature Extraction and Description”, Pattern Recogntion, Vol. 30, pp. 403-419, 1997.

[4] R Drew, S Pettitt, P Blenkhorn, D G Evans, “A Head Operated 'Joystick' Using Infrared”, Computers and Assistive Technology ICCHP '98, Proc XV IFIP World Computer Congress, Ed. A. D N Edwards, A Arato , W. L. Zagler, Österreichische Computer Gesellschaft, 1998

[5] D. G. Evans, R. Drew, P. Blenkhorn, "Controlling Mouse Pointer Position Using an Infrared Head Operated Joystick", IEEE Trans. Rehabilitation Engineering, Vol. 8, pp 107-117, 2000.

[6] D. G. Evans, P. Blenkhorn, “A Head Operated Joystick - Experience with Use”, Fourteenth Annual International Conference on Technology and Persons with Disabilities, Los Angeles, March 1999

[7] Peter Hallinan, “Recognising human eyes”, In SPIE Proc. Geometric Methods in Computer Vision, Vol. 1570, 214-226, San Diego, 1991.

[8] T. Kanade. “Picture Processing System by Computer Complex and Recognition of Human Faces”, PhD Thesis, Dept. Of Information Science, Kyoto University, 1973.

[9] B. Lucas, T. Kanade, “An iterative image registration technique with an application in stereo-vision”, 7th International Joint Conference on Artificial Intelligence, pp. 674-679, 1981.

[10] C. Poelman, “The paraperspective and projective factorization method for recovering shape and motion”, Technical report CMU-CS-95-173, Carnegie Mellon University, 1995.

[11] Y. Tian, T. Kanade, and J. Cohn, “Multi-State Based Facial Feature Tracking and Detection”, Tech. Report CMU-RI-TR-99-18, Robotics Institute, Carnegie Mellon University, August, 1999.

[12] Y-l. Tian, T. Kanade, J. Cohn, “Dual-State Parametric Eye tracking”, Fourth IEEE International Conference on Automatic Face and Gesture Recognition 2000, 26 - 30 March, 2000, Grenoble, France.

[13] Matthew Alan Turk, “Interactive-Time Vision: Face Recognition as a Visual Behaviour”, PhD Thesis, MIT, 1991.

[14] X. Xie, R. Sudhakar, H. Zhuang, “On Improving Eye Feature Extraction Using Deformable Templates”, Pattern Recognition, Vol.27, pp. 791-799, 1994.

[15] A. L. Yuille, D.S. Cohen, P. W. Hallinan, “Feature Extraction from faces using deformable templates”, Proc. Computer Vision and Pattern Recognition, pp. 104-109, 1989.

[16] http://www.orin.com/access/ (current at 15 July 2001).

[17] http://store.prentrom.com/ (current at 15 July 2001).