FACE AND EYE RECTIFICATION IN VIDEO CONFERENCE USING ARTIFICIAL NEURAL NETWORK

Ben Yip

The National ICT Australia Limited, Australian Technology Park, Sydney, NSW, Australia
The school of Information Technologies, University of Sydney, NSW, Australia
benyip@it.usyd.edu.au

ABSTRACT

The lack of eye contact in video conference degrades the user's experience. This problem has been known and studied for many years. There are hardware-based solutions to the eye gazing problem. However, these specialized systems are not generally accessible. This paper suggests a software approach that rectifies the face and the eyes in video conference, only utilizing one camera. In the setup phase, the view point, the head poses are calculated using 100 frames each from the front view and the above view, with the aids of face detection and eye detection algorithm. The weights of the artificial neural networks (ANN) are then trained. Once the setup is done, for each frame in real time, we apply ANN on the features found with the aid of face detection. The ANN outputs are feed to an image warping algorithm to rectify the face. Rectification of the eyes is done using image warping, based on an eye model. This is done in near real time (13 frames per second, for the resolution of 320x240). The result is not genuine but is better. However, the rectified face jerks when the user's head moves or turns rapidly. The author shall need to investigate more on this issue.

1. INTRODUCTION

In video conference, because the camera that captures the user's face does not align with the conference image the user looks at, it is not possible to have eye contact. If the camera is mounted above the monitor, the face appears to be looking downward. On the other hand, the face would look upward if the camera were mounted below the monitor. The problem of deviated eye gaze has been known since 1969 [11]. At that time, the video device was called "PicturePhone".

The magnitude of the eye gaze deviation depends on the angle between the camera, the human eyes, and the conference image from the monitor. This vertical deviated angle is approximately 15˚ to 20˚ in a usual laptop or desktop setting. Chen [1] showed that the sensitivity to eye contact is asymmetric. We are less sensitive to eye contact when people look below our eyes than when they look to the left, right, or above our eyes. Hence, the camera should be placed on top of the monitor to minimize the loss of eye contact.

Advanced hardware has been invented based on the idea of semi-reflective material to make the image and the camera aligned. The product VPD-32 from Digital Video Enterprises is an example available on the market [3]. Okada et al. [8] suggests projecting the conference image onto a semi-transparent screen and capturing the user's face behind the screen. The hardware-based approaches solved the eye gaze deviated problem with no additional software algorithm. However, these specialized systems are not generally accessible.

1.1. Software Approach

Zitnick et al.[17] and Gemmell et al.[4] proposed a solution with a monocular camera. They manipulated the eye gaze by synthesizing the eyeballs and re-orienting the head. A planar eyeball model was used to draw and map onto the face. Image warping was used to re-orient the head, which is successful for small angles (<5˚) of rotation. For larger angles, a 3D-head model mapped with a face image was used, which minimized the error in the correspondence maps resulted in distortions. However, the texture mapping approach using a crude predefined model reduced the reality. Also, tracking the head orientation was inaccurate and led to face distortions.

For researches using a pair of stereo cameras, Yang and Zhang [15] proposed to mount one camera on top, and the other at the bottom of the computer monitor. First, it tracked the head pose in 3D by a personalized face model. Secondly, objects that are out of the head

0-7803-9332-5/05/$20.00 ©2005 IEEE

model, such as hands and shoulders are matched. Finally, a virtual view is combined and generated.

Liu et al, [6] used 3 cameras to solve the eye gazing problem, placing them on the right, the left and on the top of the computer monitor. The disparity analysis processes images from the 3 cameras and matches for feature points. The feature points are then rendered through a virtual camera. The textures are obtained from the left and right cameras. In the case of occlusion, the texture data will come from three cameras. Talmi and Liu later proposed another approach with 3 cameras. [12] Two cameras on the left and right of the computer monitor are used to track the head and face, and the third one underneath mainly used for tracking one of the eyes. The information from all three cameras is combined to perform 3D rendering, and auto-stereoscopic display.

1.2. Our approach

Our approach requires only one camera. In the setup phase, the view point and the head poses are calculated using 100 frames each from the front view and the above view, with the aids of face detection and eye detection algorithm. The weights of the artificial neural networks (ANN) are then trained. Once the setup is done, for each frame in real time, we apply ANN on the features found with the aid of face detection. The ANN outputs are feed to an image warping algorithm to rectify the face. Rectification of the eyes is done using image warping based on an eye model.

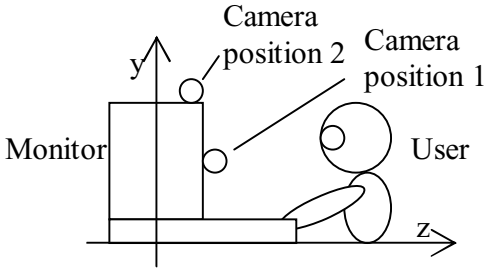

Figure 1. In the setup phase, the camera is first placed in front of the monitor, and then at the usual position.

2. SETUP PHASE

Our approach needs a simple setup. First, we ask the user to put the video camera in front of the monitor, and look through to the monitor as if the video camera wasn't there. 100 frames are captured from this view. Then we ask the user to put the camera at the usual position (we recommend on top and centered of the monitor, and assume so for the rest of this paper). In the second position, the user looks at the monitor, and 100 frames are captured. The setup phase is depicted in Figure 1.

Three important calculations are done in the setup phase: (1) face and eye detection, (2) viewpoint determination, and (3) train the artificial neural network with different head poses.

2.1. Face detection and eye detection

Face detection algorithm is applied on both the front view, and the above view image sequences. The aim is to found the smallest rectangle that bounds the entire head and hence find the top, left, bottom, right of the head.

Face detection has gained much attention in the recent years. The idea of face detection using neural network, by Rowley et al, in 1996, was a significant work at that time.[10] Since then, there are many other approaches. Interested reader may read the comprehensive survey paper by Yang, et al [14]. Viola and Jones [13] uses boosted cascade of simple features for object detection. This is one of the most well known algorithms in face detection. Barreto, et la [1] refines the algorithm by Viola and Jones. In our approach, face detection is not the main goal; we use the face detection library in OpenCV library [9] for convenience, which implements the haar-like algorithm by Lienhart. [6]

The result from face detection gives a rough rectangular area of the head. To obtain the location of the top, left, bottom, right of the head up to the precision of a pixel, advance techniques are needed. If the background is also captured in the setup phase, background substitution would be helpful to find the precise locations, however, the chin area may still be challenging.

In our approach, we use a simple color tracking algorithm to find the top of the head, assuming the color of the background is distinguishable from the color of the hair. And, for the right, left and the chin area of the head, we take what is given by the face detection algorithm. This is one source of inaccuracy that we need to improve on in the future.

With a given face, eye detection is easier, and simple ways based on the color cue, could accomplish the task. [5] However, in our approach, we use the same face detection algorithm, but trained with the eyes pairs. A simple color tracking algorithm is then used to find the precise locations of the eyeballs.

2.2. View point determination

The aim of view point determination is to find out the view angle of the camera (the above camera), relative to the front view. This information is vital as we need to calculate the magnitude of the eye rectification in later stage. From each image frame, we could obtain 8

parameters, the top, left, bottom, right of the head, and the xy coordinate of both eyes. Then, we calculate the vector from the centre of the head, to the middle of the eyes. The viewpoint of the usual location can be determined by comparing the vector of the front view to the vector of the above view. The details can be found in [16].

2.3. Train the artificial neural networks

In each frame of the above video, we calculated the head pose – the pitch angle (α) and the yaw angle (β). The details of head pose determination could also be found in [16]. In each frame of both video streams, our approach finds the location of the four feature points. They are, the top, left, bottom, right of the head. In the front view image sequence, because the camera is closer to the user, the head may appear larger. The coordinates are normalized based on the eye separation distance, and take (0, 0) at the middle of the eyes on both views.

An artificial neural network is created for each of the feature. Each ANN has five inputs: sin(α), cos(α), sin(β), cos(β), and the distance of the feature point obtained from the above video to (0, 0). The desire output of the ANN is the distance of the feature point obtained from the front video to (0, 0). The assumption behind this ANN setting is: the feature point, say the top of the head, of the front view could be calculated if we know the head pose and the location of this feature point in the above view. After 100 frames, 100 input and output sets are collected. Then, each of the ANN is trained with 1000 iterations, using backward propagation.

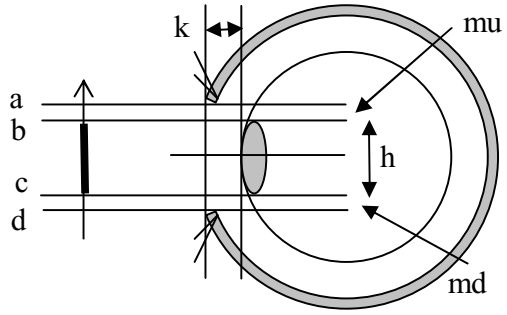

Figure 2. The eye model used to calculate the magnitude needed of eye rectification.

3. REAL TIME PHASE

The trained artificial neural networks in the setup phase are then used in real time, taking the input from the above camera. Face detection and eye detection are run and we can obtain the locations of the 4 feature points, the positions of the eyes, and also the head pose. The feature points are then normalized and feed into the ANNs along with the head pose. The outputs from the ANNs are the rectified locations of the features. Image warping is used to distort the image so that the rectified feature points are in the correct place. This process rectifies the shape of the face, and gives the feeling that the video is capturing from the front of the user. Although the eyes are very small compared to the entire image; humans are very sensitive to the shape of eyes. Hence, we need an additional algorithm to rectify the eyes.

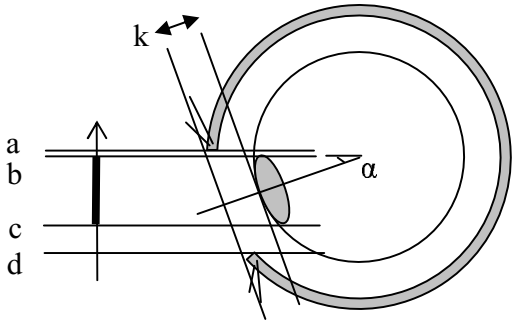

Figure 3. The eye model changes when the eyes are looked from an angle, α.

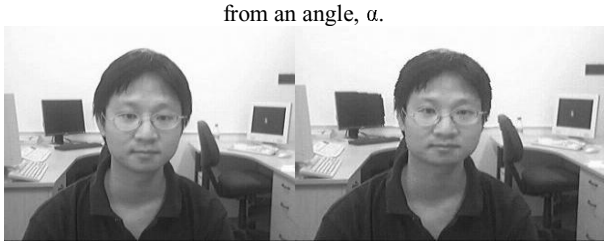

Figure 4. Video (left) is rectified with our approach (right).

3.1. Eye rectification

Figure 2 depicts the model of the eyes. **k** is the thickness of the eyelid. **h** is the diameter of the iris. **mu** is the distance from the top of the iris to the eyelid. **md** is the distance from the bottom of the iris to the eyelid. Points **a**, **b**, **c**, and **d** depict the projection of the eyes on a plane. Between point **a** and point **d** is the visible part of the eye, and between point **c** and point **d** is the iris. When the eye is viewed from an angle, the locations of points **a**, **b**, **c**, and **d** will change. This is depicted in Figure 3. The magnitudes of the changes are:

$$\Delta a = (\frac{h}{2} + mu)(\cos\alpha - 1) - k\sin\alpha\,,\ \Delta b = \frac{h}{2}(\cos\alpha - 1)$$

$$\Delta c = \frac{h}{2}(1 - \cos\alpha)\,,\ \Delta d = (\frac{h}{2} + mu)(1 - \cos\alpha) - k\sin\alpha$$

The viewpoint angle α is already calculated in the setup phase. In real time phase, after the face rectification with the affine transformation, we apply the eye detection algorithm mentioned in section 2.1. We then detect the

points **a**, **b**, **c** and **d**, and change their locations with the calculated magnitude by image warping. The value of **h** can be found from the image, the value of **k**, **mu**, and **md** are estimated as h/15, -h/10 and -h/10 respectively.

4. RESULT AND PERFORMANCE

Figure 4 shows the result of the rectification of a frame from the video stream. In evaluation, we used two video cameras, one from above, one in the front, simultaneously capture a user talking. For each frame, and each feature in both views, we normalize it, and then calculate the coordinate difference between the above view and the front view. The average difference of all the features in each frame is considered as the 'disparity' of the two views at the given frame. This disparity is plotted in gray line in Figure 5. Similarly, the disparity between the rectified view and the front view is plotted in Figure 5 with black line. The Y axis is in the units of pixels, and the X axis is the frames (200 frames). We can see that the rectified view is performing better. The average disparities for the rectified view and the above view are 6.42 pixels and 9.29 pixels respectively.

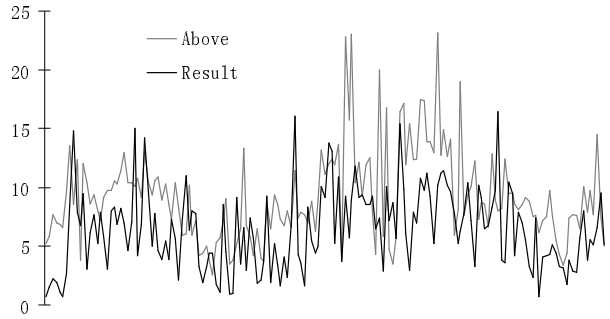

Figure 5. The disparity of the rectified view to the front view (black) is compared with the disparity of the above (gray).

5. CONCLUSION

There are many published works and proposed methods for eye gaze correction in video conferencing. Our approach utilizes artificial neural networks and a novel eye model to rectify the face and the eyes. Because image warping is used, the foreground objects (e.g. glasses, hand gestures) and the background objects are preserved however may be distorted. The setup phase is simple. With a Pentium4 2.5 GHz machine, we could achieve 13 frames per second with a 320x240 video stream in the real runtime phase. The result is not perfect, but still provides significant improvement. One drawback is the rectified face jerks when the user's head moves or turns rapidly. The imprecise locations of the head features are the main source of error. Hence, our future work will be focused on improving our method's performance through better head feature localization.

6. REFERENCES

[1] Barreto, J., Menezes, P., Dias J.; Human- robot interaction based on Haar-like features and eigenfaces, Proc. ICRA, session Th-A2, 2004.

[2] Chen, M.; Leveraging the asymmetric sensitivity of eye contact for videoconference. SIGCHI, ACM Press, pp.49 – 56, 2002

[3] Digital Video Enterprises. http://www.picturephone.com - Eye-Contact VPD-32 Videoconferencing Display System, last accessed 2003.

[4] Gemmell, J., Zitnick, C.L., Kang, T., Toyama, K., and Seitz, S.; Gaze-awareness for videoconferencing: a software approach. IEEE Multimedia, 7(4):26–35, 2000.

[5] Kumar, T.R., Raja, K.S., and Ramakrishnan, A.G. Eye detection using color cues and projection functions. In Proc. ICIP, pp Vol.3 337-340, 2002.

[6] Lienhart R., Maydt, J. An Extended Set of Haar-like Features for Rapid Object Detection. Proc. ICIP 2002, Vol. 1, pp. 900-903, Sep. 2002

[7] Liu, J., Beldi, I-P., and Wöpking, M.; A computational approach to establish eye-contact in videocommunication. Proceedings of International Workshop on Stereoscopic and Three Dimensional Imaging, IWS3DI'95, 1995.

[8] Okada, K.I., Maeda, F., Ichikawa, Y., and Matsushita, Y.; Multiparty videoconferencing at virtual social distance: MAJIC design. Proc. CSCW '94, pp.385-395, 1994

[9] Open source Computer Vision library, Intel Corp. http://www.intel.com/research/mrl/research/opencv/, last accessed 2005.

[10] Rowley, H.A., Baluja, S. and Kanade, T. Neural Network-Based Face Detection, IEEE Transactions on Pattern Analysis and Machine Intelligence, 20 (1) pp 23-38, 1998

[11] Stokes, R., Human Factors and Appearance Design Considerations of the Mod II PicturePhone Station Set, IEEE Trans. Communication Tech., 17(2):318-323, 1969

[12] Talmi, K., and Liu, J.; Eye and gaze tracking for visually controlled interactive stereoscopic displays. Signal Processing: Image Communication 14:799-810, 1999

[13] Viola, P., Jones, M.; Rapid object detection using a boosted cascade of simple features. In proc CVPR, 2001

[14] Yang, M-H., Kriegman, D., and Ahuja, N.; Detecting Faces in Images: A Survey, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), vol. 24, no. 1, pp. 34-58, 2002.

[15] Yang, R., and Zhang, Z.; Eye gaze correction with stereovision for video-teleconferencing. Microsoft Research, Technical Report, MSR-TR-2001-119, 2001

[16] Yip, B; Jin, J.S.; Viewpoint determination and pose determination of human head in video conferencing based on head movement. Proc. 10th MMM, pp 130-135, 2004

[17] Zitnick, C.L., Gemmell, J., and Toyama, J.; Manipulation of video eye gaze and head orientation for video teleconferencing. Microsoft Research, Technical Report, MSR-TR-99-46, 1999.