EYE BLINK DETECTION FOR SMART GLASSES

Hoang Le[⋆] *Thanh Dang*[⋆] *Feng Liu*[†]

[⋆] Washington State University, Vancouver
[†]Portland State University

ABSTRACT

Eye blink is a quick action of closing and opening of the eyelids. Eye blink detection has a wide range of applications in human computer interaction and human vision health care research. Existing approaches to eye blink detection often cannot suit well resource-limited eye blink detection platforms like Smart Glasses, which have limited energy supply and typically cannot afford strong imaging and computational capabilities. In this paper, we present an efficient and robust eye blink detection method for Smart Glasses. Our method first employs an eigen-eye approach to detect closing-eye in individual video frames. Our method then learns eye blink patterns based on the closing-eye detection results and detects eye blinks using a Gradient Boosting method. Our method further uses a non-maximum suppression algorithm to remove repeated detection of the same eye-blink action among consecutive video frames. Experiments with our prototyped smart glasses equipped with a low-power camera and an embedded processor show an accurate detection result (with more than 96% accuracy) on video frames of a small size of 16×12 at 96 fps, which enables a number of applications in health care, driving safety, and human computer interaction.

Index Terms— Eye blink detection, Smart glasses

1. INTRODUCTION

With recent advancements in micro-electromechanical systems, it is possible to integrate multi-modal sensors such as camera and accelerometer into embedded platforms and into user accessories like glasses. This paper focuses on Smart Glasses that combine both hardware and software technologies to monitor and collect human vision activities for human computer interaction and human vision health study applications. Smart Glasses have low computation power, limited energy, memory, and bandwidth. Thus, the computer vision algorithms for Smart Glasses must be efficient and be able to work robustly with low-quality videos.

This paper works on the problem of eye blink detection on Smart Glasses. Eye blink is a quick action of closing and opening of the eyelids [1]. Blink detection is an important enabling component in various domains such as human computer interaction, health care, and driving safety. For example, blink has been used as an input modality for people with disabilities to interact with computers [2, 3]. Blinking patterns (e.g., frequency and duration) are correlated to people's drowsiness and consciousness [4, 5]. Thus, blink detection has been used to detect consciousness degradation and concentration of drivers [4]. Blinking patterns are also used in modeling and animating eye blinks in film industry [6].

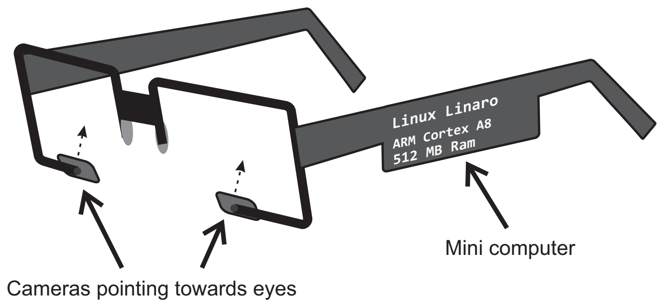

Fig. 1. Smart Glasses Camera Configuration.

Most previous blink detection systems use a camera to capture the face and then locate eye regions. Then, blinks can be detected using various methods. In [2, 7], correlation score with templates of opening eyes were used to detect blinks. In [8], blinks are detected when pupils cannot be detected in some successive frames. In [6, 9], a statistical active appearance model was used to track and detect blinks. While these methods achieve good blink detection results, they do not fit well with the design of our Smart Glasses. Smart Glasses are expected to last an extended period of time but have limited power supply, which prevents us from using powerful imaging, storage, communication, and computing units. In addition, Smart Glasses will be used under a wide range of illumination conditions as shown in Fig. 2, which also makes existing blink detection methods unsuitable for Smart Glasses.

In this paper, we present an efficient and robust approach to eye blink detection that suits low-power platform well, such as Smart Glasses. As closing-eye is an important signature of eye blink, our method first detects closing-eye in each video frame using an eigen-eye approach. Our method then uses a Gradient Boosting (GB) algorithm to learn the eye-blink patterns based on the closing-eye detection results. Note, while the learning steps in the eigen-eye approach and the GB-based blink detection method are slow, they can be performed offline. We only need to run the detection steps online. Our

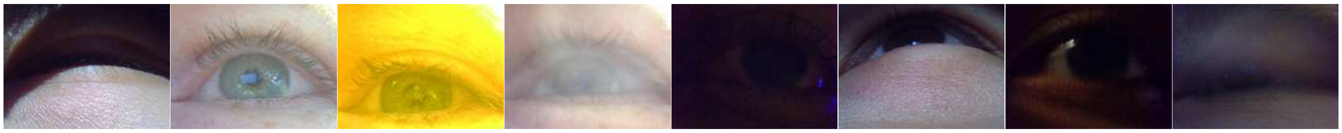

Fig. 2. Eye images captured by Smart Glasses in different lighting conditions. These images often have low quality and vary greatly from one another, clearly showing the need for a robust eye blink detection method.

method finally employs a non-maximum suppression algorithm to remove repeated detection of the same eye-blink action among consecutive video frames. Our experiments show that this eye blink detection algorithm can run in real time on Smart Glasses and work robustly with low-resolution videos.

2. SMART GLASSES CAMERA CONFIGURATION

We have developed a prototype system for Smart Glasses using all low-budget Commercial Off-The-Shelf (COTS) components, as illustrated in Fig. 1. The frame of Smart Glasses is taken from a regular eyeglasses. We embed two cameras onto Smart Glasses, one for each eye. Each camera points towards the corresponding eye to capture eye activities. The camera is located below the eye center and close to the bottom eyelid in order not to block a user's eye sight. The cameras are firmly connected to the eyeglasses to avoid vibration when the user moves. For processing ability, we attach MK802, a mini computer, to Smart Glasses. MK802 contains CPU Allwinner A10 1.0GHz Cortex-A8 and has 512 MB ram. It supports Android Ice Cream Sandwich (ICS) 4.0 and Linux Linaro operating system. We use Linux Linaro in our current design.

3. BLINK DETECTION

In a video, we detect an eye blink if several consecutive frames capture the quick motion of closing and opening the eyelids. Detecting eye blink in real time from videos captured by Smart Glasses is challenging. The video quality is low and unstable. We show some sample frames in Fig. 2. Moreover, the computational capability is poor. As eyes are open for most of the time, closing-eye in a video frame is a strong indication of a possible eye blink. We therefore develop an eye blink detection approach that first detects closing-eye frames and then detects eye blink. We adopt data-driven approaches for both closing eye and blink detection. These approaches first train a detector and then use it to detect event of interest. While the training step is typically slow, it can be run off line. The detecting step can be designed to run more efficiently. In the following subsections, we first describe an eigen-eye approach for detecting closing-eye video frames and then describe a method for eye blink detection.

3.1. Close Eye Detection

Eigen analysis has been widely used in image and video analysis, such as face recognition [10]. We use eigen analysis for close-eye detection. We first collect a training dataset that contains images with open eyes and close eyes. We then apply Principal Component Analysis to this training dataset and obtain a set of basis images [11]. An eye image can then be reconstructed as a linear combination of these basis images. We use its coefficients as a feature vector to represent each eye image. We then use Gradient Boosting (GB) to train a close-eye/open-eye classifier using the training dataset [12]. For each frame, this classifier outputs the probability of this frame containing a close eye. We define this probability as close-eye score and use it as input to the next step of eye blink detection. Fig. 3 shows an example of close-eye detection.

The performance of this eigen-eye approach can be compromised by the misalignment of eye images. Different users can wear glasses differently, which makes the eye positions different for different users. We solve this problem by a pre-calibration step beforehand. Specifically, when a user starts to wear the glasses, Smart Glasses can automatically detect the eye position using the Haar Cascade classification method from OpenCV[1] and remind the user to adjust the wearing of the glasses until the eye is roughly in the video center.

3.2. Blink Detection

An eye blink action consists of a small number of consecutive frames that captures an open-close-open blink cycle. To save energy, Smart Glasses captures videos at 10 fps. Our discussion with a Human Vision Research scientist shows that 6 consecutive frames can well capture a full eye blink cycle. We therefore detect eye blink in a window of 6 frames.

We use a data-driven approach to capture the eye blink patterns from the closing eye score obtained in the above step. We collected a set of 6-frame video segments that contain eye blink as the positive training dataset and a set of 6-frame video segments that do not have eye blink as the negative training dataset. For each video segment, we use the close-eye scores for its frames as its feature vector. We then train an GB classifier to detect whether any given 6-frame segment contains an eye blink action or not.

This blink detection method will sometimes lead to duplicated detections of the same blink action when we slide the 6-frame window in the captured video. That is, two neighboring detection windows that overlap with each other significantly will often both be detected as containing a blink although they capture the same one. This makes the counting of eye blink difficult. We use a non-maximum suppression method

[1] http://opencv.org/

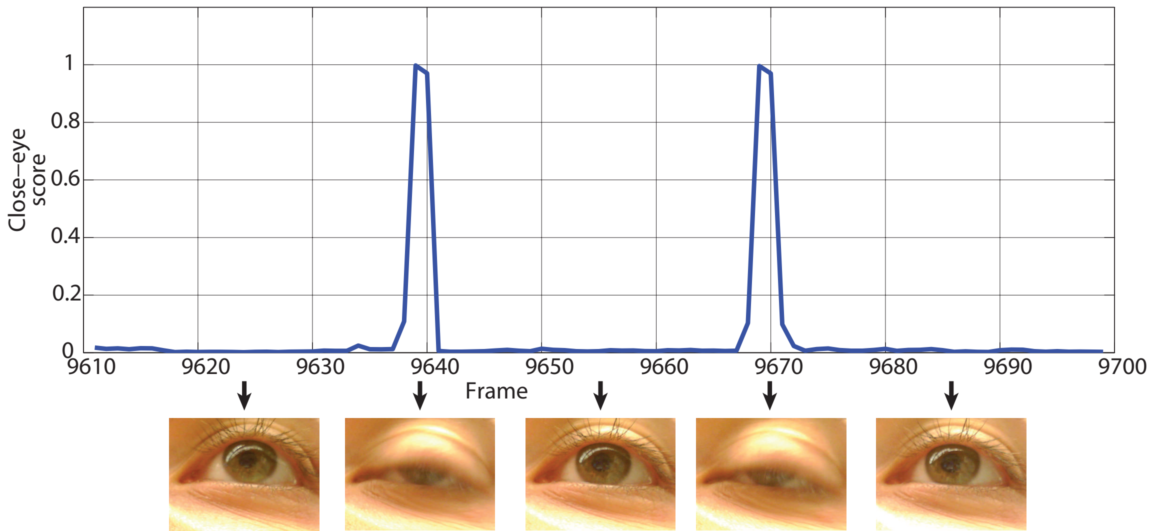

Fig. 3. Close-eye detection.

to solve this problem. Specifically, our method looks ahead a few frames and examines the probabilities of the consecutive detection windows containing an eye blink. Our method picks the detection window with the maximal probability and nullifies the results of its neighboring detection windows.

4. EXPERIMENTS

We experimented with our blink detection algorithm on videos captured by our Smart Glasses prototype system. While Smart Glasses can capture videos with resolution 640×480, it is preferable that our blink detection method can work well with smaller sizes due to the limited resource on Smart Glasses. For our experiments, we captured 640×480 videos and down-sampled them to three smaller video sizes, 64×48, 32×24 and 16×12. These videos were captured from 4 participants covering 3 different races in various lighting environment when participants were using computer, walking indoors and outdoors. Fig. 2 shows sample frames.

Our method uses PCA to extract features and uses Gradient Boosting for close eye and blink detection. It is interesting to test how these two technical components compare to other alternatives. Therefore, we tested PCA and Independent Component Analysis (ICA) [13] for the feature extraction step and tested Support Vector Machine (SVM) [14], Gradient Boosting (GB) and Gaussian Naive Bayes (GNB) for the detection steps. So we have 6 different algorithm combinations in total.

We collected an eye-image set that contains 3861 images with a close-eye as positive examples and 3861 images with an open-eye as negative examples. We also collected a set of 6-frame video segments. 1419 of these video segments contain an eye blink action and the other 1419 segments do not. We then randomly allocated 75% of the dataset into a training set and the rest 25% into a testing set to evaluate different combination of feature extraction and detection methods.

Table 1, 2, and 3 show the blink detection accuracy of the 6 different methods on thee three different frame sizes mentioned above. For each feature selection method, we also vary the number of features. The speeds of these algorithms, running on Smart Glasses, are reported in Table 4, 5, and 6.

Our experiment shows the performance of these methods is pretty consistent with different image sizes. The speed of each algorithm decreases when the image size or feature number increases. Overall, at frame size 16×12, our methods, which use either PCA or ICA to extract less than 30 features and use the Gradient Boosting as the detection method, obtain a good trade-off between accuracy and speed.

5. CONCLUSION

In this paper, we described a novel approach to eye blink detection on resource-limited Smart Glasses, which have weak power supply and cannot afford powerful imaging and computing capacities. Our method uses data-driven approaches and can work robustly with low-quality videos captured by Smart Glasses. Our method can also run in real time on Smart Glasses. Our method provides robust and real-time eye blink detection for a range of applications of Smart Glasses in human computer interaction and human vision health care.

Acknowledgment

This research was sponsored in part by the Portland State University Faculty Enhancement Grant and NSF CNS-1218589.

6. REFERENCES

[1] W. P. Blount, "Studies of the movements of the eyelids of animals: Blinking," *Experimental Physiology*, vol. 18, no. 2, pp. 111–125, 1927.

Table 1. Detection accuracy (%) of frame size 16×12.

		Number of Features					
Feature	Detection	15	20	30	40	60	80
PCA	SVM	96	97	97	97	98	97
	GB	**96**	**96**	**97**	**97**	95	96
	GNB	85	86	88	86	74	71
ICA	SVM	93	94	96	97	97	97
	GB	**96**	**97**	**96**	**97**	93	94
	GNB	86	86	88	89	89	87

Table 2. Detection accuracy (%) of frame size 32×24.

		Number of Features					
Feature	Detection	15	20	30	40	60	80
PCA	SVM	97	97	97	98	98	98
	GB	96	97	97	97	97	98
	GNB	85	86	88	86	78	71
ICA	SVM	93	94	96	97	98	97
	GB	96	97	98	97	96	96
	GNB	85	85	88	90	89	90

Table 3. Detection accuracy (%) of frame size 64×48.

		Number of Features					
Feature	Detection	15	20	30	40	60	80
PCA	SVM	97	97	98	98	98	98
	GB	96	97	97	97	97	97
	GNB	84	86	87	86	78	73
ICA	SVM	94	94	97	97	98	98
	GB	96	97	97	97	96	96
	GNB	84	85	88	91	90	91

Table 4. Detection speed (FPS) of frame size 16×12.

		Number of Features					
Feature	Detection	15	20	30	40	60	80
PCA	SVM	90	78	85	82	69	66
	GB	**95**	**95**	**95**	93	93	92
	GNB	88	89	88	87	86	84
ICA	SVM	78	77	75	73	64	60
	GB	**95**	**95**	**95**	94	92	91
	GNB	90	88	88	87	86	85

Table 5. Detection speed (FPS) of frame size 32×24.

		Number of Features					
Feature	Detection	15	20	30	40	60	80
PCA	SVM	82	75	80	77	66	54
	GB	91	90	88	87	82	79
	GNB	86	84	82	80	77	74
ICA	SVM	75	74	72	68	65	62
	GB	92	90	87	86	84	80
	GNB	86	85	83	81	77	74

Table 6. Detection speed (FPS) of frame size 64×48.

		Number of Features					
Feature	Detection	15	20	30	40	60	80
PCA	SVM	77	65	52	61	52	41
	GB	80	76	71	66	59	53
	GNB	76	72	68	63	56	50
ICA	SVM	68	65	61	56	50	45
	GB	81	78	72	68	60	54
	GNB	76	74	69	64	57	51

[2] K. Grauman, M. Betke, J. Gips, and G. Bradski, "Communication via eye blinks - detection and duration analysis in real time," in *IEEE CVPR*, 2001, pp. 1010–1017.

[3] K. Grauman, M. Betke, J. Lombardi, J. Gips, and G. Bradski, "Communication via eye blinks and eyebrow raises: Video-based human-computer interfaces," *Universal Access in the Information Society*, vol. 2, no. 4, pp. 359–373, 2003.

[4] N. Kojima, K. Kozuka, T. Nakano, and S. Yamamoto, "Detection of consciousness degradation and concentration of a driver for friendly information service," in *IEEE International Vehicle Electronics Conference*, 2001, pp. 31 –36.

[5] P. Smith, M. Shah, and N. da Vitoria Lobo, "Monitoring head/eye motion for driver alertness with one camera," in *IEEE ICPR*, 2000, pp. 636–642.

[6] Laura C. Trutoiu, Elizabeth J. Carter, Iain Matthews, and Jessica K. Hodgins, "Modeling and animating eye blinks," *ACM Trans. Appl. Percept.*, vol. 8, no. 3, 2011.

[7] M. Chau and M. Betke, "Real time eye tracking and blink detection with usb cameras," Tech. Rep., Boston University, 2005.

[8] D. Kim, S. Choi, J. Choi, H. Shin, and K. Sohn, "Visual fatigue monitoring system based on eye-movement and eye-blink detection," in *Stereoscopic Displays and Applications*, 2011, pp. 786303:1–8.

[9] I. Bacivarov, M. Ionita, and P. Corcoran, "Statistical models of appearance for eye tracking and eye-blink detection and measurement," *IEEE Transactions on Consumer Electronics*, vol. 54, no. 3, pp. 1312 –1320, 2008.

[10] M.A. Turk and A.P. Pentland, "Face recognition using eigenfaces," in *IEEE CVPR*, 1991, pp. 586–591.

[11] I. Jolliffe, *Principal component analysis*, Wiley Online Library, 2005.

[12] Jerome H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, vol. 29, pp. 1189–1232, 2000.

[13] A. Hyvärinen and E. Oja, "Independent component analysis: algorithms and applications," *Neural Netw.*, vol. 13, no. 4-5, pp. 411–430, 2000.

[14] Chih-Chung Chang and Chih-Jen Lin, "Libsvm: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 27:1–27, 2011.