Computer Input System Using Eye Glances

Shogo Matsuno(✉), Kota Akehi, Naoaki Itakura, Tota Mizuno, and Kazuyuki Mito

Graduate School of Informatics and Engineering, The University of Electro-Communications, 1-5-1 Chofugaoka, Chofu, Tokyo 182-8585, Japan
ml440004@edu.cc.uec.ac.jp

Abstract. We have developed a real-time Eye Glance input interface using a Web camera to capture eye gaze inputs. In previous studies, an eye control input interface was developed using an electro-oculograph (EOG) amplified by AC coupling. Our proposed Eye Gesture input interface used a combination of eye movements and did not require the restriction of head movement, unlike conventional eye gaze input methods. However, this method required an input start operation before capturing could commence. This led us to propose the Eye Glance input method that uses a combination of contradirectional eye movements as inputs and avoids the need for start operations. This method required the use of electrodes, which were uncomfortable to attach. The interface was therefore changed to a camera that used facial pictures to record eye movements to realize an improved noncontact and low-restraint interface. The Eye Glance input method measures the directional movement and time required by the eye to move a specified distance using optical flow with OpenCV from Intel. In this study, we analyzed the waveform obtained from eye movements using a purpose-built detection algorithm. In addition, we examined the reasons for detecting a waveform when eye movements failed.

Keywords: Eye glance · Eye input · Optical flow · Input interface

1 Introduction

Eye movements are the quickest among face movements. In this study, we developed a new method for automatically detecting eye glances by analyzing video images captured using a Web camera. We aimed to develop a multichannel computer system incorporating an eye movement input technique based on the eye-glance detection method. Improvements in computing performance have led to the downsizing and widespread use of wearable devices. However, these wearable devices are difficult for users to control because they are small, have many functions, and are complicated. To solve this problem, several methods involving eye movements have been attempted [1, 2], and eye-movement input has attracted attention as a hands-free input system. However, information devices need to become smaller and easier to control to develop an eye-movement input system.

A study of conventional eye-gaze input interfaces revealed that the most popular type of gaze input interface uses the gaze position to input characters. However, if the subject's head moves, it is not possible to identify the gaze position precisely by only

S. Yamamoto (Ed.): HIMI 2015, Part I, LNCS 9172, pp. 425–432, 2015.
DOI: 10.1007/978-3-319-20612-7_41

detecting the eye movement. Even if there is no eye movement, the gaze position changes unless the head is immobilized. Special equipment is required to identify the gaze position with high accuracy without fixing the head; therefore, the equipment becomes more expensive.

Thus, we devised an input method capable of accepting gaze inputs [3]. A previous study [4] of ours suggested the use of Eye Gesture inputs in combination with the directional movement of the eye instead of gaze inputs and realized a simple eye-gaze input method using the AC-EOG (Alternating Current-Electro-oculogram). Furthermore, we proposed that the only input that the Eye Glance input method should require should be the characteristic eye movement in combination with the direction of the eye movement as used by the Eye Gesture input. To summarize, we demonstrated the possibility of realizing a simple eye-gaze input that, unlike Eye Gesture inputs, does not require an input start operation. In the case of Eye Glance inputs, it is only necessary to detect the direction of the eye movement, which does not require a high degree of accuracy. The Eye Glance input is therefore a facile method for obtaining inputs. To date, Eye Glance input has been realized using the AC amplification EOG; however, its use of an electrode patch required effort to apply and caused discomfort.

In this study, we proposed a new method for automatically detecting eye glances by using a Web camera. We show the experimental results of this system using this method.

2 Eye Glance Input

The eye-gaze input method is popular among systems that use eye information [5, 6]. In a previous study [7], we reviewed existing input methods with the aim of reducing system costs and the restrictions placed on the user. The result was a novel Eye Gesture input interface that did not require a HMD. Instead, directional combinations for eye-gaze displacement were used as the selection method. We found that eye gaze displacements could be determined precisely using a derivative EOG signal amplified via AC coupling. A desktop display design created for use with the Eye Gesture input interface is shown in Fig. 1. In that study, it was assumed that Eye Gesture movements followed oblique patterns (upper left, lower left, upper right, lower right) and that each pattern consisted of a combination of two movements.

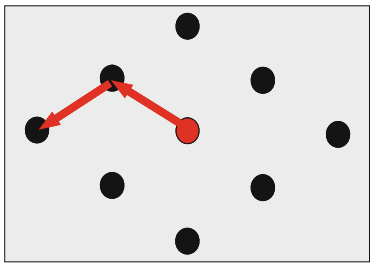

Fig. 1. Eye gesture input model

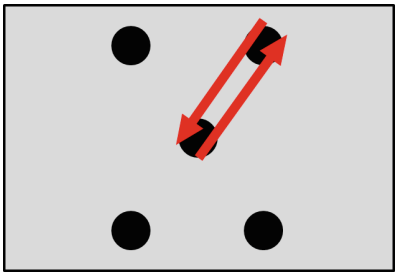

Fig. 2. Eye glance input model

However, in this study, we used eye glances (eye movements) because they are relatively fast and easy to input. Using an eye glance has two advantages over the eye-gaze input method: the wait time of the input is short, and the eye movement is a very quick and unique motion that leads to relatively light processing load. As the first step, we examined a four-channel input method that can be applied to the operation of various devices such as smartphones [3]. Figure 2 shows an example of the eye-glance model. The input trigger is the coming-and-going movement of the eyes in the slant direction.

Eye Glance input is a technique that uses eye movements that occur when someone only looks at something for a moment as inputs. Previous studies have used the AC EOG method, in which an electrode patch must be placed on the face. New methods use video cameras instead.

3 Eye Glance Detection Method

Incorporating eye movement detection equipment in commonly used devices such as tablets would necessitate the use of inexpensive equipment such as the device's built-in camera or a USB camera, rather than using expensive equipment such as a high-speed camera. Therefore, we used an inexpensive USB Web camera in this experiment. The camera has a resolution of 1.3 million pixels and can obtain images at 30 fps. Figure 3 shows the measurement range, the size of which is 200 × 60 pixels.

Fig. 3. Measurement range

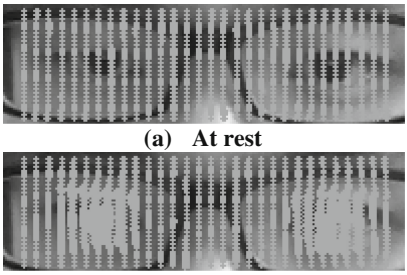

Fig. 4. Measurement of optical flow

This method uses the optical flow to detect eye movements. The optical flow was measured using a library based on that of Gunnar Farneback in OpenCV. To calculate the vector, 200 points are considered in the measurement range. Figure 4 shows facial images that were obtained by measuring the optical flow. Any eye movement has the result of changing the vector of every point around the eye. This value is used to calculate the optical flow between the two images. By repeating this from the first frame to the last frame, the amount of movement of the eye in the area is recorded as the amount of change in the optical flow. The frame rate of the camera was 30 fps.

Figure 5 shows the waveform obtained during eye glance movements (upper right). If we can automatically detect ocular movements in each direction, we can use a multichannel input system that has different commands for each direction. An input is recorded when the waveform exceeds the threshold set for the horizontal direction and when this occurs continuously within the stationary time. The stationary time is the time between two peaks of a round-trip waveform. We calculated the optical flow and determined the threshold required to detect eye glances automatically. The threshold of the speed vector was assumed to be 70 % of the average of the data obtained by each

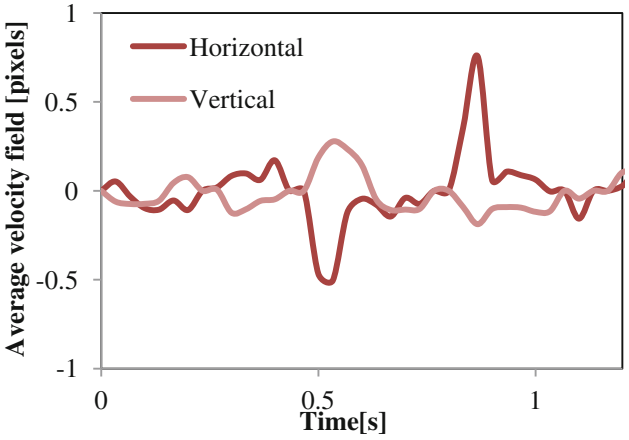

Fig. 5. Waveform of eye glance moving to upper right

subject. Further, thresholds were determined using a separate waveform for the positive and the negative directions.

4 Experiment

We investigated the process to determine the threshold to detect eye movements automatically by experimenting with five subjects. Figure 6 shows an overview of the experimental system. The subjects were shown video images of the eye captured using a Web camera. First, the subjects performed a calibration to determine the threshold to remove artifacts. They watched the index at the center of the screen for 5 s so that they would not move their face; we measured the optical flow and determined the threshold. We then displayed the experimental image that set the photographed image of the eye at the center and the indexes at the four corners. The view angle between the middle indicator and the corner indicators in the experimental image was 15 × 7. The subjects gazed at the indexes once each in the following order: center, one of the four corners, and center clockwise from the upper right. The obtained data were treated as one set. The subjects performed 10 trials and took a break between these trials. They adjusted the face position by looking at the image photographed by the camera and displayed before each set.

4.1 Non-Real-Time Measurement Experiment (Experiment 1)

Table 1 presents the experimental results. The pre-evaluation (non-real-time measurement) experiments indicated that the average detection rate of the proposed method

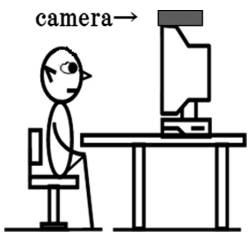

Fig. 6. Overview of experimental system

Table 1. Results of experiment 1

Subjects	Upper right [%]	Lower right [%]	Lower left [%]	Upper left [%]	Average [%]
A	80.0	80.0	90.0	80.0	82.5
B	70.0	60.0	50.0	60.0	60.0
C	100.0	60.0	100.0	60.0	77.5
D	90.0	80.0	70.0	60.0	75.0
E	90.0	60.0	90.0	70.0	75.0
Average	86.0	68.0	80.0	66.0	74.0

was 74 %. In this experiment, 900 points are considered in the measurement range to calculate the vector.

4.2 Real-Time Measurement Experiment (Experiment 2)

In this experiment, the system, method, and environment are the same as those in experiment 1. However, four different subjects are used. These subjects performed 30 trials and took a break between these trials. In addition, this system sends feedback signals to the subject about when to input an eye glance. Figure 7 shows the experimental screen. The feedback signal is shown after 0.8 s (Fig. 8).

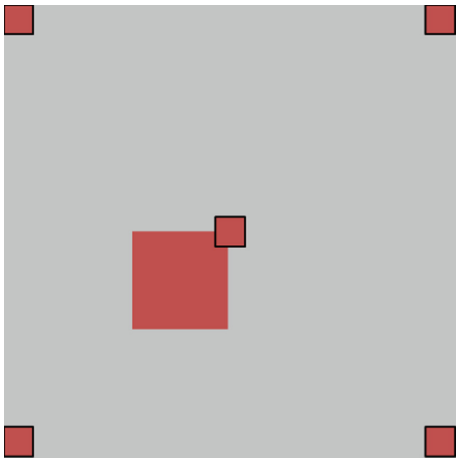

Fig. 7. Experiment screen (movement order toward lower left)

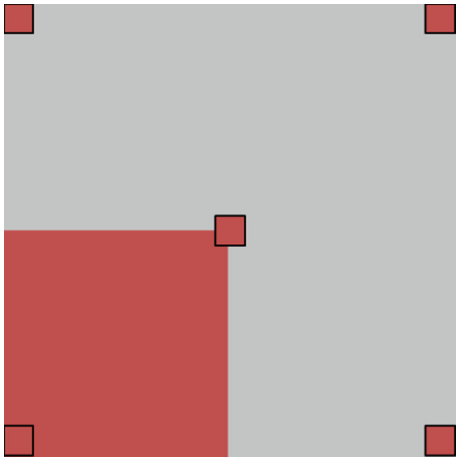

Fig. 8. Feedback signal (to input eye glance)

Table 2. Results of experiment 2

Subjects	Upper right [%]	Lower right [%]	Lower left [%]	Upper left [%]	Average [%]
A	83.3	70.0	86.7	83.3	80.8
B	80.0	66.7	63.3	73.3	70.8
C	93.3	90.0	83.3	93.3	90.0
D	93.3	86.7	66.7	83.3	82.5
Average	87.5	78.4	75.0	83.3	81.0

Table 2 presents the experimental results. The evaluation experiments indicated that the average detection rate of the proposed method was 81 %.

5 Discussion

Overall, the real-time measurement (experiment 2) showed a higher successful detection rate than the non-real-time measurement (experiment 1) despite lesser vector points being calculated in experiment 2 than in experiment 1. We attribute this to the presence of feedback signals in experiment 2. The average input success rate will increase if the feedback signal contributes to successful input and habituation of the subjects because subjects use this system for the first time during this experiment.

By using our proposed method, the average rate of successful eye glance input is $\sim$80 % for the experimental sample of four subjects (Table 2). We think this rate is sufficient for device control if the system can tolerate some missing inputs. However, subject B's average rate was comparatively low because of the noise of waiting time. Some subjects move their eye frequently and do not perform eye glances in order. We attribute this to unconscious movements by the subjects. This movement destabilizes the vector of the near-eye area. We consider changing the measurement area of the vector to the black eye area from the near eye area to solve this problem.

6 Conclusion

We proposed a real-time eye glance input method that detects eye glances using a Web camera. This method realizes easy noncontact inputs to a computer to detect eye movements using image analysis. We experimentally tested the system implemented using this method. In our experiments with four different subjects, the average successful input rate was 81 %.

In the future, we plan to conduct additional experiment with more subjects. We also plan to improve this method to increase the detection accuracy. In addition, we will investigate the incorporation of this method into mobile devices.

References

1. Miluzzo, E., Wang, T., Campbell, A.T.: Eye gesture recognition: eyephone: activating mobile phones with your eyes. In: Proceedings of the Second ACM SIGCOMM Workshop on Networking, Systems and Applications on mobile handhelds, pp. 15–20 (2010)
2. Mayberry, A., Hu, P., Marlin, B., Salthouse, C., Ganesan, D.: iShadow: design of a wearable, real-time mobile gaze tracker. In: Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services, pp. 82–94. ACM (2014)
3. Naoaki, I., Takumi, O., Kazutaka, S.: Investigation for calculation method of eye-gaze shift from electro-oculograph amplified by ac coupling with using eye-gaze input interface. IEICE Trans. Inf. Syst. **J90-D**(10), 2903–2913 (2007)
4. Dekun, G., Naoaki, I., Tota, M., Kazuyuki, M.: Improvement of eye gesture interface system. In: Proceedings of the 16th Asia Pacific Symposium of Intelligent and Evolutionary Systems, No. 3, pp. 1–3 (2012)
5. Abe, K., Ohi, S., Ohyama, M.: An eye-gaze input system using information on eye movement history. In: Stephanidis, C. (ed.) UAHCI 2007 (Part II). LNCS, vol. 4555, pp. 721–729. Springer, Heidelberg (2007)
6. Majaranta, P., Raiha, K.-J.: Twenty years of eye typing: systems and design issues. In: Proceedings of the Symposium on ETRA 2002, pp. 15–22. ACM (2002)
7. Gao, D., Itakura, N., Mizuno, T., Mito, K.: Study of eye-glance input interface. In: Kurosu, M. (ed.) HCII/HCI 2013, Part IV. LNCS, vol. 8007, pp. 225–234. Springer, Heidelberg (2013)