SafetyPIN: Secure PIN Entry Through Eye Tracking

Mythreya Seetharama[1], Volker Paelke[2], and Carsten Röcker[3(✉)]

[1] Ostwesfalen-Lippe University of Applied Sciences, Lemgo, Germany
[2] Bremen University of Applied Sciences, Bremen, Germany
[3] Ostwestfalen-Lippe University of Applied Sciences
and Fraunhofer IOSB-INA, Lemgo, Germany
carsten.rocker@iosb-ina.fraunhofer.de

Abstract. When a user enters a personal identification number (PIN) into an automated teller machine or a point of sale terminal, there is a risk of some one watching from behind, trying to guess the PIN code. Such shoulder-surfing is a major security threat. In order to overcome this problem different PIN entry methods have been suggested. In this regard, gaze interaction methods are receiving attention in recent years, owing to the lowering cost of eye tracking technology. In this paper, we present SafetyPIN - an eye tracking based PIN entry system - which is aimed at making the PIN entry more secure with the help of an eye tracking device. We discuss the implementation and the initial evaluation of this system.

Keywords: PIN entry · Eye tracking · Security · Usability · Point of sale terminals

1 Introduction

The use of PINs (personal identification numbers) as passwords for authentication is ubiquitous nowadays. This is especially true for banking applications where the combination of a token (e.g. bank card) and the user's secret PIN is commonly used to authenticate transactions. In financial applications PINs are typically four-digit numbers, resulting in 10000 possible numbers. The security of the system relies on the fact that an attacker is unlikely to guess the correct PIN number and that the systems (e.g., Automated Teller Machines) limit the user to few attempts (e.g., 3) for entering the correct PIN. As most applications that use PINs for authentication operate in a public setting a common attack is to try to observe and record a user's PIN entry (shoulder-surfing). These security problems have been recognized for a long time and researchers have proposed a number of different schemes to minimize the risk of PIN entry observation. One such proposed alternate PIN entry method requires the user to input some information, which is derived from a combination of the actual PIN and some additional information displayed by the system, instead of the

T. Tryfonas and I. Askoxylakis (Eds.): HAS 2015, LNCS 9190, pp. 426–435, 2015.
DOI: 10.1007/978-3-319-20376-8_38

PIN itself [1]. Another approach proposes the use of an elaborate hardware to make PIN entry resilient to the observation attacks [2]. However, these methods have not been introduced into practical applications because the users would have to be retrained to use a completely different approach to PIN entry and the significant additional costs involved in the hardware setup.

In the SafetyPIN project our goal is to prevent observation attacks during PIN entry while retaining the same workflow that user's are already familiar with and with minimal additional hardware cost. Our setup can be easily integrated into existing designs of automatic teller machines (ATMs) and point of sale systems. To avoid shoulder-surfing attacks and enable users to enter their PIN without fear of being observed we have developed a system that employs an eye tracking device. With SafetyPIN, users select PIN numbers with their eyes by simply focusing on digits displayed on a screen. Since 0the physical key-press is not used for the PIN entry, no information about the entered digit is given away to the attacker through visual observation. Use of fake keypads are also rendered unnecessary.

The rest of the paper is structured as follows. In Sect. 2, some previous efforts related to preventing shoulder-surfing and use of eye interaction are mentioned. In Sect. 3, conceptual approach behind SafetyPIN is explained. Section 4 details the implementation. In Sect. 5, the initial evaluation and the results are discussed. Section 6 concludes the paper.

2 Related Work

Researchers have been evaluating the user gaze as an interaction method using eye tracking devices. In 1987, Ware et al. [3] evaluated two methods of interacting with the computers using eyes as input: dwell gaze, look and shoot. The dwell gaze method relied on the user looking at the region of interest on the screen for a certain amount of time. In the look and shoot method the user looked at the region of interest and then physically clicked a predefined button on the keypad to activate the region. The dwell gaze method needs more time for activation compared to the look and shoot method, as it needs the dwell time to ensure that spurious activations are avoided. On the other hand, the look and shoot method, though quicker, gives away more information for the potential shoulder-surfer via the button click feedback. Both these methods require calibration to be performed for the individual user.

Kumar et al. [4] evaluated the above two methods for ATM password entry to avoid shoulder-surfing. They used the Tobii 1750 eye tracker and a qwerty alpha numeric keypad for this purpose. The evaluation suggested that these methods are capable of deterring shoulder-surfers while taking comparable time for entering password as compared to conventional keypad. They also suggested that the calibration data for the user can be stored in the ATM card itself so that it need not be performed every time.

De Luca et al. [5], in addition to the methods above, introduced a gaze-gesture method of password entry and compared it with the other two methods. In the

gaze-gesture method the user is required to remember a graphical pattern and then input that pattern via gesturing through his/her eyes [6,7]. The advantage of this method is that it requires no calibration as it depends on the relative position of the eye, not the absolute position. However, it suffers on the usability front as the users need many retries to get the pattern right.

Other such efforts can be seen in [8,9]. In SafetyPIN, we have implemented a new activation method called Blinking, along with the other two methods. In this method, unlike the look and shoot, the user looks at the region he/she wants to activate and then instead of pressing a key, blinks his/her eyes to activate the region. This is more secure than the look and shoot, since the feedback given to the shoulder-surfer via the physical pressing of the button is completely avoided. This method is also less error prone compared to the gaze method since the spurious activations are less likely.

3 Conceptual Approach

Our initial prototype runs on a standard Windows PC and uses the Tobii EyeX low-cost eye tracker. The eye tracker consists of a small bar that can be attached to a display screen and could be incorporated into an ATM at a later stage. The sensor bar contains micro-projectors that project distinct patterns of infrared light at the user's eyes. The reflections of these patterns are then recorded by infrared cameras in the sensor bar. Through image processing the user's eyes are detected and the eye movements tracked, which is then used to determine the user's gaze

Fig. 1. 9-digit visual key-pad displayed on the screen, Tobii EyeX eye tracker mounted below the screen

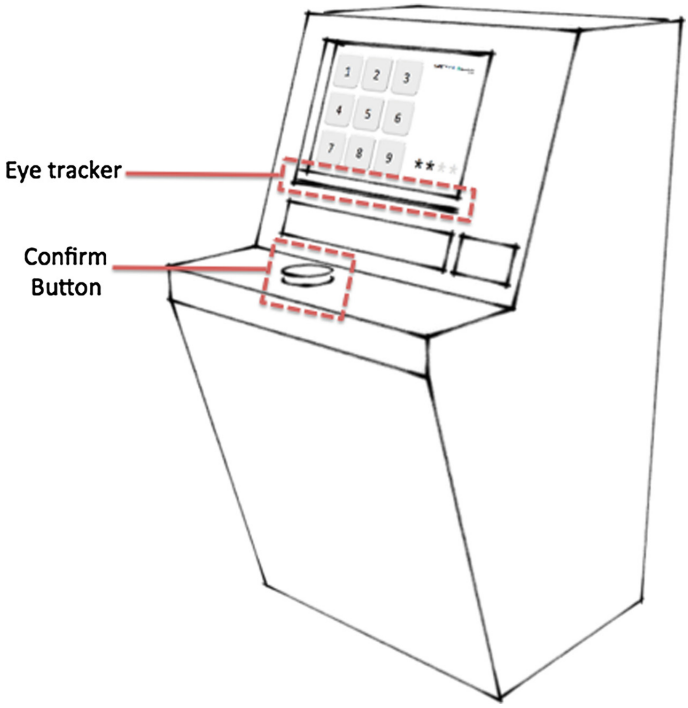

Fig. 2. SafetyPIN hardware components for retrofitting into existing point of sale terminals

point: the point on the screen that the user's view currently focuses on. For the PIN input we display the possible digits on the screen (see Figs. 1 and 2).

From the practical perspective a key advantage of the SafetyPIN approach is that only minimal additional hardware is required which fits easily into existing terminals. Because the sensor bar is small and placed directly below the screen it could be integrated into the screen housing of existing terminals, simplifying the development of new versions with integrated SafetyPIN entry and proving the opportunity to retrofit existing terminals.

4 Implementation

The prototype is aimed at mocking up a typical ATM PIN entry screen and allowing the user to enter the PIN using three different interaction methods: look and shoot, gaze activation and blink activation. The GUI is a 1680×720 pixel window with buttons labeled 0–9 along with ',' and '.'. The GUI buttons are 160×100 pixels in dimension with a spacing of 50 pixels between them. The user is supposed to enter a predefined sequence of numbers as his PIN. The software then checks for the accuracy and speed of the entered PIN for each of the three entry methods mentioned above. This software has been developed in VC++.

```
initialization;
while Wait for data/events from EyeX do
    if coordinates receieved correspond to a GUI button then
        if the activation button is pressed then
            store the PIN corresponding to the button activated;
        end
    end
end
```

Algorithm 1. Algorithm for the Look and Shoot Activation Method

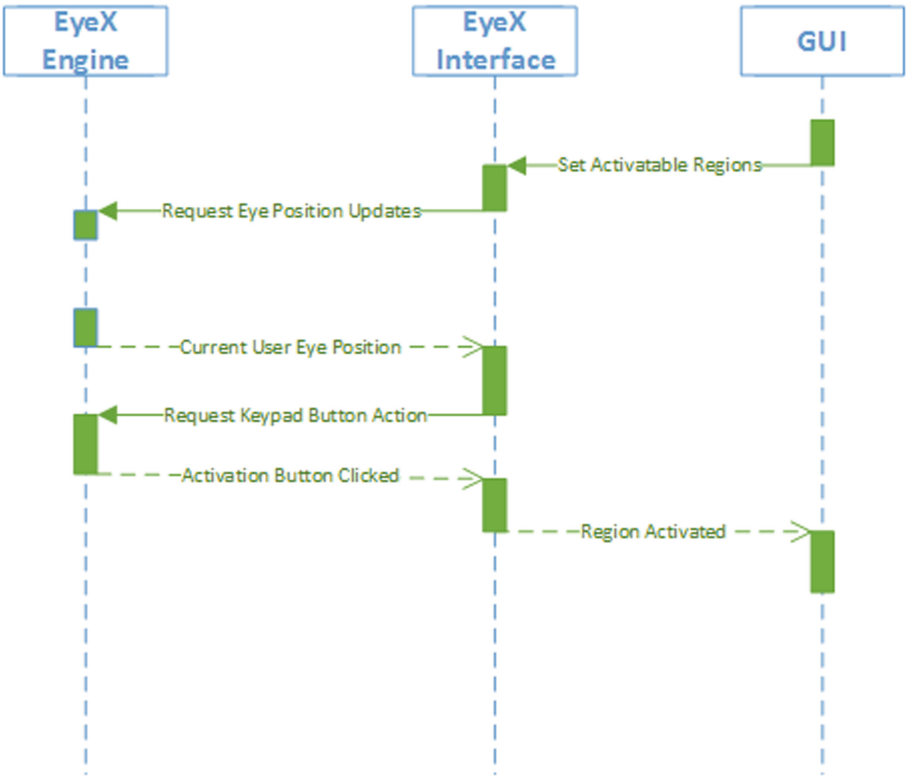

Fig. 3. Sequence diagram for the look and shoot activation method

The Tobii SDK [10] provides drivers for the eye tracking device along with C/C++ library engine which gives API for interfacing with the device. These APIs provide functionalities higher than the raw eye position data from the device. The engine provides two kinds of high level operations. On the one hand the application program can inform the engine about the boundaries of the regions that it wants to get activated on. The engine will then intimate the application program about when the user looks at one the regions specified.

```
initialization;
while Wait for data/events from EyeX do
    if coordinates receieved correspond to a GUI button then
        if gaze event notification is received for the current GUI button then
            store the PIN corresponding to the button activated;
        end
    end
end
```

Algorithm 2. Algorithm for the Gaze Activation Method

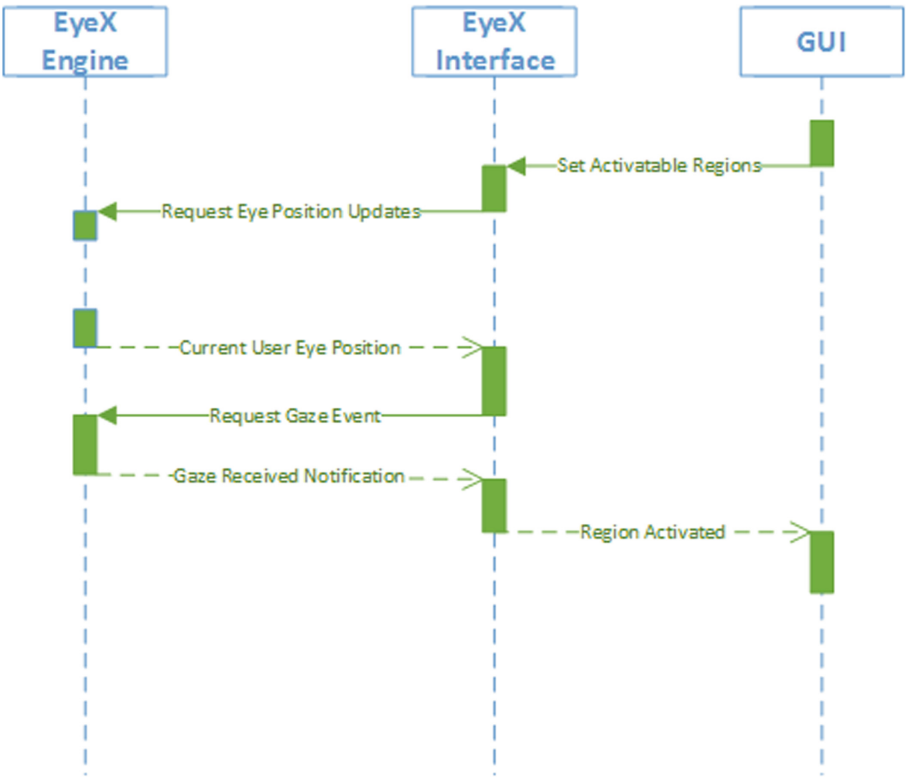

Fig. 4. Sequence diagram for the gaze activation

This scheme relieves the application program from having to poll the incoming raw position data from the device. On the other hand, the application program can register for one of the many events for which it would like to get notifications on. For example, when the user looks at a region for more than half a second a gaze event can be notified and when the device does not see the user's eyes for more than a second an absence event can be notified to the application program depending on whether the application program has registered for the event or not. These notifications are used in our GUI application program.

The three different methods of activation are described below with the help of pseudo-code and sequence diagrams.

4.1 Look and Shoot

The sequence diagram in Fig. 3 shows the three major modules of the software and gives an overview of their interactions. The GUI interacts with the EyeX Engine using an EyeX Interface module. The GUI initializes the windows and button components and sends the coordinates of the buttons to the EyeX Interface, which in turn requests the EyeX Engine for the periodic eye position updates. The EyeX Engine is directly communicating with the controller device. Upon receiving the position coordinates from the EyeX Engine, the EyeX Interface checks whether the position where the user is currently looking falls within the bounds of any buttons or not. If so, it requests the EyeX Engine to intimate it when the predefined activation button, right control key in this case, is pressed. Once this activation event is received, the EyeX Interface sends the message to the GUI with the button number to be activated. The GUI, upon receiving this message stores the activated PIN. Algorithm 1 depicts this process in a pseudo-code fashion.

4.2 Gaze Activation

Figure 4 shows the interaction for the gaze activation method. The major difference in this case is that upon receiving the position coordinates from the EyeX Engine, the EyeX Interface requests for the gaze event notification after performing the bounds check. EyeX Engine produces this notification if the user stares at the same region for more than half a second. But this time was found to be too small and lead to spurious activations. Therefore, the gaze time was increased to more than a second by validating it in the EyeX Interface. After this, the message is passed onto the GUI from the EyeX Interface module.

4.3 Blink Activation

Figure 5 shows the interaction for the blink activation method. The major difference in this case is that upon receiving the position coordinates from the EyeX

```
initialization;
while Wait for data/events from EyeX do
    if coordinates receieved correspond to a GUI button then
        temporarily remember this GUI button;
        if user absence event notification is received then
            if user presence event notification received then
                store the PIN corresponding to the button remembered;
            end
        end
    end
end
```

Algorithm 3. Algorithm for the Blink Activation

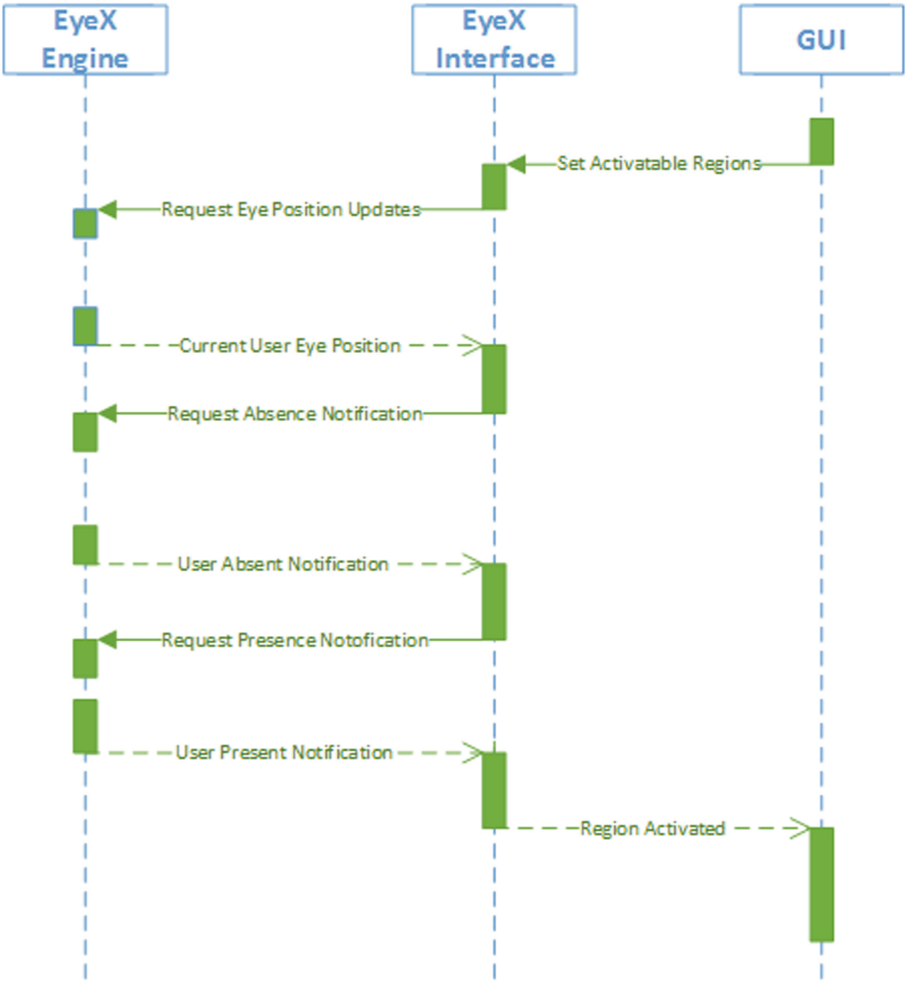

Fig. 5. Sequence diagram for the blink activation

Engine, the EyeX Interface requests for the 'user absence' notification after performing the bounds check and temporarily saving the button number. EyeX Engine produces this notification if the device fails to see the eyes of the user for more than a second. Therefore, if the user closes his eyes for a second, he effectively becomes absent for the device, producing the required notification. Upon receiving this notification, EyeX Interface requests for the 'user presence' notification, effectively waiting for the user to open his eyes again, thus performing a blink operation. Once the user opens his eyes the EyeX Engine sends the required notification and the EyeX Interface sends the component number of the saved button to the GUI.

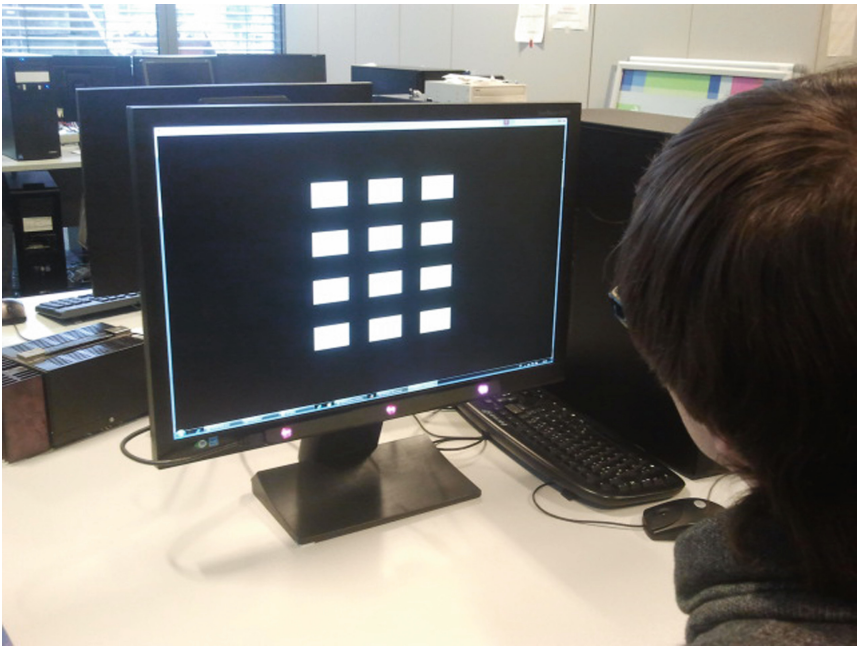

Fig. 6. Usability test with the user in front of the prototype system

5 Evaluation

In the initial user tests, we have examined both the technological aspects like the impact of calibration errors, the impact of glasses and other eyeware, practical usability aspects like error rates and user satisfaction, as well as the user's perception of safety and security aspects.

The test was performed for 9 users, where each user was given a command sequence of 12 digit PIN to be entered using the eye tracker (see Fig. 6). This test was performed for all three activation methods, four times per activation method. The PIN entered by the user was stored and then compared with the command PIN to find out how many errors occurred. Time taken by the users for the test was also recorded. After the tests were completed, the users were given a questionnaire to collect the feedback from the users regarding how usable and useful did they feel the system was. Their feedback on the safety was also recorded.

Results of these tests have been very encouraging in all the three activation methods. The average error rate was around 5 % and the average time taken was around 1.6 s per a digit entry or around 6.7 s for a typical 4 digit PIN entry. Most of the errors were committed by the users when they did not remember the correct PIN and therefore entered the wrong PIN, rather than activating an unintentional PIN. Users felt that the system was easy to use and that this was a safer way to enter their pin compared to the traditional pin entry method. To

draw statistically significant conclusions, we need to perform further tests with larger sample set.

6 Conclusions

In order to protect the users from shoulder-surfing in ATMs while entering the PIN, new methods of entering the PIN are being evaluated. With the eye tracking technology becoming cheaper, eye interaction for PIN entry is emerging as a practical solution. In this paper, we have discussed SafetyPIN, which proposes retrofitting the ATMs with an eye tracking device, so that users can enter their PIN without using the keypad for pin entry. In our prototype, we have implemented and evaluated the system for a PC. In addition to the 'look and shoot' and gaze activation methods, we have introduced a new activation method called blink activation. Initial user evaluations have yielded encouraging results, prompting further work.

References

1. Roth, V., Richter, K., Freidinger, R.: A PIN-entry method resilient against shoulder surfing. In: Proceedings of the ACM conference on Computer and communications security (CCS 2004), New York pp. 236–245 (2004)
2. Sasamoto, H., Christin, N., Hayashi, E.: Undercover: authentication usable in front of prying eyes. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2008), Florence, pp. 183–192 (2008)
3. Ware, C., Mikaelian, H.: An evaluation of an eye tracker as a device for computer input. In: Proceedings of CHI 1987, Toronto (1987)
4. Kumar, M., Garfinkel, T., Boneh, D., Winograd, T.: Reducing shoulder-surfing by using gaze-based password entry. In: Proceedings of the 3rd Symposium on Usable Privacy and Security, pp. 13–19. ACM (2007)
5. De Luca, A., Weiss, R., Drewes, H.: Evaluation of eye-gaze interaction methods for security enhanced PIN-entry. In: Proceedings of the 19th Australasian Conference on Computer-human Interaction: Entertaining User Interfaces, pp. 199–202. ACM (2007)
6. Drewes, H., Schmidt, A.: Interacting with the computer using gaze gestures. In: Baranauskas, C., Abascal, J., Barbosa, S.D.J. (eds.) INTERACT 2007. LNCS, vol. 4663, pp. 475–488. Springer, Heidelberg (2007)
7. Drewes, H., De Luca, A., Schmidt, A.: Eye-gaze interaction for mobile phones. In: Proceedings of the 4th International Conference on Mobile Technology, Applications, and Systems and the 1st International Symposium on Computer Human Interaction in Mobile Technology, pp. 364–371. ACM (2007)
8. Forget, A., Chiasson, S., Biddle, R.: Shoulder-surfing resistance with eye-gaze entry in cued-recall graphical passwords. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1107–1110. ACM (2010)
9. Dunphy, P., Fitch, A., Olivier, P.: Gaze-contingent passwords at the ATM. In: 4th Conference on Communication by Gaze Interaction (COGAIN), pp. 59–62 (2008)
10. Tobii EyeX SDK for C/C++, Developer's Guide. Tobii Technology (2014)