Exploiting Eye Tracking for Smartphone Authentication

Dachuan Liu[1,2], Bo Dong[2], Xing Gao[1,2], and Haining Wang[1]

[1] University of Delaware, Newark, DE, USA,
[2] College of William and Mary, Williamsburg, VA, USA,
{dliu,bdong,xinggao}@cs.wm.edu, hnw@udel.edu

Abstract. Traditional user authentication methods using passcode or finger movement on smartphones are vulnerable to shoulder surfing attack, smudge attack, and keylogger attack. These attacks are able to infer a passcode based on the information collection of user's finger movement or tapping input. As an alternative user authentication approach, eye tracking can reduce the risk of suffering those attacks effectively because no hand input is required. However, most existing eye tracking techniques are designed for large screen devices. Many of them depend on special hardware like high resolution eye tracker and special process like calibration, which are not readily available for smartphone users. In this paper, we propose a new eye tracking method for user authentication on a smartphone. It utilizes the smartphone's front camera to capture a user's eye movement trajectories which are used as the input of user authentication. No special hardware or calibration process is needed. We develop a prototype and evaluate its effectiveness on an Android smartphone. We recruit a group of volunteers to participate in the user study. Our evaluation results show that the proposed eye tracking technique achieves very high accuracy in user authentication.

Keywords: authentication, eye tracking, privacy protection, smartphone

1 Introduction

Two authentication methods, passcode-based and finger movement pattern-based, have been widely used by various smartphones for user authentication. However, previous research has revealed that both authentication methods are vulnerable to shoulder surfing attack [28], smudge attack [2], and keylogger attack [5, 26, 18, 19]. For shoulder surfing attacks, an attacker could steal a password just by peeking over a user's shoulder when the user is entering its password. Recently, some researchers found that it is possible to steal a password even when a user is behind some obstacles [25]. Smudge attacks exploit the oily residues left on the screen for inferring a password. Keylogger attacks are launched from the inside of device. The malicious program running on the smartphone utilizes smartphone's sensors to record the vibrations during the authentication. Then attackers could figure out the password based on those information. All these attacks exploit the information from user's hand typing or finger moving activities.

The authentication methods leveraging eye tracking do not need hand input; therefore, they are resistant to those attacks above. So far, there are already some works applying eye tracking techniques in user authentication. These works can be classified into biometric-based [3, 14, 12, 13] and pattern-based [15, 7, 9, 10]. The biometric-based methods authenticate a user based on the biometric information extracted from the user's eyes or eye movement characteristics. Differently, the pattern-based methods require a user to issue commands via their eye movements. The pattern-based authentication can be further divided into two types. The first type [15] tracks a user's gaze point on the screen as the input. A calibration process is required for predicting the gaze point accurately. And users have to keep their heads fixed after the calibration. The other type [7, 9, 10] recognizes a user's eye movement trajectory that represents a specific command, and does not need calibration process. Most of these eye tracking applications are proposed for the devices with large screen. Many of them require special hardware like high resolution eye trackers.

However, it is impractical for smartphone users to either carry a high resolution eye tracker or conduct the calibration process. In this paper, we propose a new eye tracking authentication method for smartphone users. We leverage the eye movement trajectories as the input, which reflect eye moving direction but not the exact gaze point on the screen. Neither extra eye tracker nor calibration process is needed.

In our proposed scheme, there are multiple moving objects on the screen, one of which is the target. A user just tracks the moving target with her eyes. The authentication passes when the user's eye movement trajectories match the target's movement trajectories. The routes of all moving objects are *randomly* generated every time. Therefore, an attacker cannot infer the password by observing the user's eye movement during authentication. Each object should also move very differently from the others, and thus the user's eye tracking trajectory can easily match the target's trajectory. We develop a prototype based on Android 4.2.2 and deploy it on Google Nexus 4 smartphone. Then we invite 21 volunteers to take the user study. The evaluation results show that average authentication accuracy is as high as 91.6%. The major research contributions of this work are summarized as follows:

- To the best of our knowledge, this is the first smartphone authentication method applying the eye tracking technique that does not require extra eye tracker and calibration process.
- We design a movement pattern for the authentication. The randomness within the movement pattern reduces the risks of leaking a password. Besides, The movement pattern just requires four corresponding eye movement actions, which are basic and straightforward for users to perform, achieving high detection rate.
- We introduce and compare six metrics used for matching the eye movement trajectory and target movement trajectory. We identify the most effective

metric based on our experiments. Four of them are not used in previous works, and two newly introduced metrics lead to higher detection rate than those used in the previous works.

– We implement a prototype on Android OS, and conduct a user study to evaluate the effectiveness of this proposed user authentication scheme.

The remainder of the paper is organized as follows. In Section 2, we introduce threat models to the popular user authentication methods on smartphones. We present the new authentication method in Section 3. Then we evaluate its effectiveness in Section 4. The limitations of our work are discussed in Section 5. We survey related work in Section 6. Finally, we conclude this work in Section 7.

2 Threat Models

In this section we present the threat models in the existing authentication methods on smartphone. Two kinds of authentication methods are popular among most smartphone users. One is the passcode-based and the other is pattern-based. As a classic authentication method, the passcode-based methods need a user to type its passcode. Pattern-based methods require a user to move fingers following some pre-set patterns. Both authentication methods are vulnerable to shoulder surfing attack, smudge attack, and keylogger attack.

To launch a shoulder surfing attack, an attacker just peeks from a user's shoulder when the user is entering the password. Then the attacker can infer the password based on the keyboard layout and the user's typing actions. A recent research work [25] reveals that attackers could steal a password even if the user is behind some obstacles. A smudge attack [2] exploits the oily residues, called smudge, left on the touch screen to infer a user's password. Attackers just hold camera at special angle to the orientation of the touch screen, and put the device under special lighting source and lighting angle. Under the certain conditions, the password pattern could be exposed. Some other attacks utilizing the acoustic of the tapping are introduced in [4, 29].

Keylogger attacks compromise a user's password from the inside of device. They leverage various sensors like the accelerometer and the gyroscope equipped on a smartphone to extract the behavior features of each individual. These information could result in the leakage of a password. In [5], it is observed that tappings on the different position of a screen cause different vibrations. Attackers can infer a password based on the vibration features. Xu et al.[26] proposed to collect the information from more sensors like the accelerometer, gyroscope, and orientation sensors. Using the collected information, they constructed the user pattern to calculate the user's action and input. TapPrints [18] estimates the tapping location by using machine learning to analyze the motion data. In [19], the authors can conjecture the input sequences using the data extracted from the accelerometer sensor.

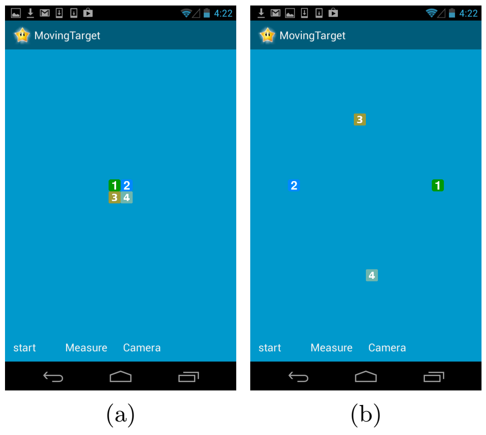


Fig. 1: (a) The layout of objects before they start to move (b) The four objects are moving in four different directions (up, down, left and right) in a round.

3 A New Authentication Approach

Applying eye tracking techniques in user authentication can significantly reduce the risks of suffering those attacks mentioned above. We design a new authentication approach based on eye movement pattern so that a smartphone user can just use the device's front camera and skip the calibration before each authentication. Compared to the user patterns like EyePassShape and Eye gesture [7] which require a user to draw some shape using eyes actively, tracking the moving object with eyes in a passive manner is much easier. Besides, users do not need to remember the complex shapes but just the target object as a password. Considering that humans' eyes move in fast and straight saccades and thus cannot perform any curves or other non-linear shapes [10], we make the objects move in straight lines for eye tracking. In the following, we first introduce the basic authentication process and the architecture of our eye tracking authentication system. Then we present how to measure the similarity between the eye movement trajectory and the target movement trajectory.

3.1 Authentication Process

The basic authentication process is described as follows. There are four objects in the center of the screen at the beginning. The layout is shown in Figure 1a. Each object is labeled with a number in the range of 1 to 4, and moves in a straight line smoothly for five rounds. In each round, the four objects move to different directions simultaneously. When the objects are moving, the user tracks the target object using eyes. Figure 1b shows a snapshot when the objects are moving. The target object represents the password in that round. When the objects start to move, the user eye-tracks the target and could extend the vision in that direction beyond the screen for providing a more clear eye movement trajectory. Once the movements stop at the end of each round, all objects return to the original positions. Meanwhile, there is a beep sound to notify the user to move eyes back to the center. Furthermore, before the next round starts, all the objects pause for one second to guarantee that the user moves eyes back to the center. The front camera of the smartphone captures the eye movements and delivers the frames to the analysis component, which extracts the eye points from each frames. Then a set of metrics will be calculated based on the eye points. These computed metrics are compared to those of the target's movement trajectory. If the metrics match, the authentication passes. Here is an example showing how our authentication scheme works.

- The user sets the password like 1-2-3-1-4. Each digit represents the target object in the corresponding movement round.
- When the user is ready, she just clicks the "start" button to initiate the authentication.
- All the objects are moving at the same time, the user uses her eyes to track the target object in that round.
- After the five rounds movement, the system outputs the match result.

3.2 System Architecture

The authentication system's architecture consists of two parts: the front-end and the back-end. The front-end includes pattern design, route generation, and moving control. The back-end mainly captures eye movement trajectory and matches it to the target movement trajectory. The architecture is illustrated in Figure 2.

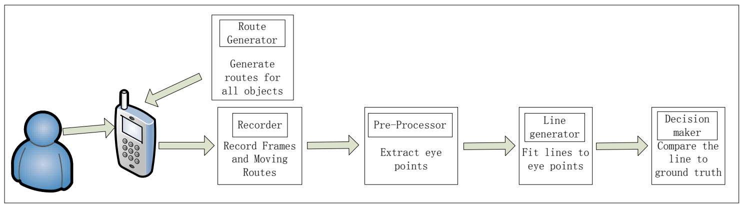


Fig. 2: Architecture of the system

Front-end We propose to secure the authentication by moving all the objects randomly each time. Since the authentication process does not need hand input, the smudge attack and keylogger attack cannot steal any information from the authentication process. For the shoulder surfing attack, even if attackers record the eye movement and figure out the eye movement trajectories, they cannot pass the authentication by replaying the same set of trajectories. This is because the target's routes for moving are random in each time. Moreover, attackers have to deploy a camera close enough to the user's eyes to capture the eye movement trajectories, which makes it a challenging task without alerting the user.

With respect to the layout, all the objects are in the center of the screen at the beginning of each round. To make the user locate the target easily, the start positions of all objects should not be changed in each round. Pursuits [24] shows that the detection rate decreases when there are many objects on the screen. We also find that users may look towards some other moving objects when they are tracking the target. We call it distraction problem. It becomes serious when two moving objects are close to each other. If we leave the objects at one side of the screen or the corners of the screen at the beginning, the objects could move across one another. In addition, setting the start positions of the objects at different corners may exposure the password. In such a scenario, the user will look to a corner at the beginning of each round. Then, the attacker could figure out the start position of the target by only observing the gaze direction of the user. Thus, in our design, the objects move far away from each other while they are clustered together in the center of the screen at the beginning of a round.

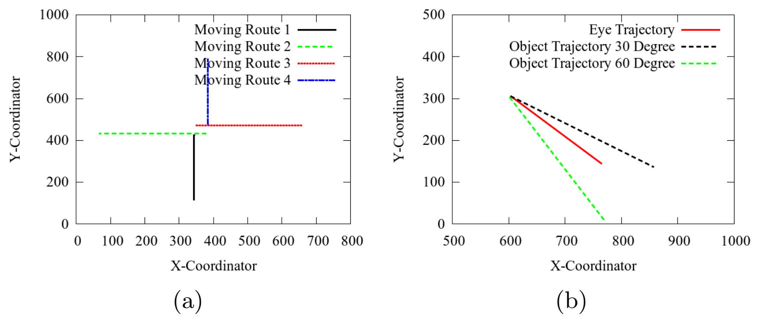


Fig. 3: (a) The trajectories of four objects in a round (b) It is hard to tell which object the user tracks with eyes when the trajectories of two objects are close to each other.

The four objects move to four different directions: up, down, left, and right, which is shown in Figure 3a. There are four reasons for such a design. (1) Since the screen of a smartphone is much smaller than a regular screen device, the number of the moving objects on the screen should be small to avoid the distraction problem. (2) For the purpose of not exposing the target, all the objects' movement directions are evenly distributed on the screen. Furthermore, the fewer objects, the larger the angle between two objects' movement directions. Consequently, it is easier to match the eye movement trajectory to the target movement trajectory. In other words, it is easier to distinguish the eye movement trajectory from the other objects' movement trajectories. (3) Although users can look at any direction theoretically, it is difficult for them to control eyes to move in an exact angle. Looking up, down, left, and right are the four most basic and simplest eye movement actions for users. (4) Since the eye movement just roughly follows the target movement, the problem appears when the eye movement trajectory is close to two different objects' movement trajectories. In such a case, it is hard to tell which object the user eye-tracked. As shown in Figure 3b, it is unclear which object the user is eye-tracking. In our design, the four directions are distinguishable from each other and help alleviate such problems.

Another disadvantage of a small screen is that the user's eye movement could be negligible if the user only looks inside of the screen boundary. Some users could be able to look any positions on the screen without obvious eye movements. In such a case, it is hard to tell the user's eye movement trajectory. To make the eye movement more clear to be detected, we allow the user to look beyond the screen area following the target's movement direction, and provide a beep sound to remind the user look back when the movement ends.

In our current design, we just set five movement rounds in the prototype and the corresponding key space is $4^5 = 1024$. The key space can be enlarged simply by allowing a user to choose different number of movement rounds for authentication. Specifically, a password could consist of arbitrary number of digits. The system first asks the user to input the number of movement rounds, then it provides corresponding object movements for authentication. More movement rounds make the authentication safer. Note that the authentication method can be applied in different scenarios, such as unlocking a phone and accessing an important file.

Back-end We leverage the front camera to capture the eye movements. The record starts when the target begins to move. It ends when the target finishes its movement within one round. The eye tracking component will extract eye points from these continuous frames. After the 5th round eye tracking finishes, the Decision Maker starts to match the eye movement trajectory to the target's movement trajectory. Note that the Decision Maker only informs the user of the final match result after five rounds, and does not inform the user about the match result for each round. A mismatch notice could benefit the legitimate user because the user can start a new authentication early if the current eye-tracking round fails. However, it is insecure, because it also informs the attacker whether the guessed number in the current round is correct. Then, the attacker just needs to try at most four times to identify the target object in each round and 20 times to uncover the whole password.

There are two sets of eye points (left eye and right eye) whose corresponding trajectories could be different. It could be that both trajectories match the target's trajectory or only one of them matches that of the target. When the user is eye-tracking the target object, it is possible she peeks to another moving object because of the distraction problem. The problem could make one eye's movement trajectory deviate from the target's trajectory. However, it is very hard for the user to intentionally eye-track two different objects at the same time. So, we regard that the user eye-tracks the target when there is at least one eye's movement trajectory matching the target's movement trajectory.

We introduce six metrics to measure the similarity between the eye movement trajectory and an object's movement trajectory. If the eye movement trajectory is most similar to the target's movement trajectory, we regard it as a match. On the other hand, if the eye movement trajectory is most similar to a non-target object's movement trajectory, the authentication fails.

Measure the Similarity After the pre-processor extracts eye points, the crucial task is how to effectively measure the similarity between the eye movement trajectory and the target's movement trajectory. Assume that the screen is a rectangular coordinate system, we can refine the problem as how to measure the similarity between two lines with directions. We expect that the user's eye movement trajectory should be similar to that of the moving target. The similarity is represented as that the two trajectories' direction should be close to each other.

In previous works [22, 24], correlation is used for matching. Principle Component Analysis (PCA) [3] is also used to estimate the direction. In this paper, we propose to fit a straight line into the eye points and compare the angle difference between this line and the target trajectory. We adopt RANdom SAmple Consensus (RANSAC) algorithm and introduce three error functions for line fitting. Simple Linear Regression (SLR) is another potential option for line fitting. We compare and evaluate them with the previous methods in the evaluation part.

In the following, we present the metrics used to measure the similarity. Correlation can measure the linear association between two variables X_a and X_b in statistics. It is defined as the covariance of the two variables divided by the product of the two variables' standard deviations. The formula is

$$\rho_{X_a,X_b} = \frac{E[(X_a - \mu_{X_a})(X_b - \mu_{X_b})]}{\sigma_{X_a}\sigma_{X_b}}$$

The coefficient is between $+1$ and -1, where $+1$ represents the total positive correlation, 0 means no correlation, and -1 stands for the total negative correlation. The formula can calculate the correlation between two variables. However, each eye point contains two variables X and Y coordinates. In such a case, the correlation between eye movement and object movement has to be calculated separately: one is for X and the other is for Y. In the previous works, the authors claimed that if X and Y of the eye movements change with those of the object movements, the user's eyes move following the objects. A threshold is set for determining whether the two trajectories match.

In this work, we propose to fit a straight line into the eye points whose angle should be close to the target's trajectory. Four methods are used to fit a line into the eye points. The first three are based on the RANSAC algorithm. RANSAC is designed for removing the noise and identifying the inliers in a set. As an iterative method, RANSAC cannot test all data points for the mathematic model exhaustively for a large set of data. However, the number of eye points is limited. Thus, we can try all possible combinations in a short time. The algorithm is described in Algorithm 1.

We leverage RANSAC's idea and introduce three error functions (Err1, Err2, Err3) in the algorithm. Err1 measures the number of points whose distance to the line is less than a threshold. Based on the observed data, we set the threshold to 3 pixels, implying that the line containing most points under this distance

Algorithm 1 RANSAC algorithm with error functions

```
1: Data: Eye movement points
2: Result: A line matches the points
3: BestMode1, BestMode2, BestMode3
4: BestMode1Score= 0
5: BestMode2Score = BestMode3Score = Infinity
6: Err1(line L): return the number of points whose distance to the line is smaller than a threshold
7: Err2(line L): return the sum of distance to the line of all points
8: Err3(line L): return the sum of squared distance of all points
9: for point p1 in the set do
10:    for another point p2 in the same set do
11:        generate a line L based on the two points p1 and p2
12:        if Err1(L) > BestMode1Score then
13:            BestMode1 = L
14:        end if
15:        if Err2(L) < BestMode2Score then
16:            BestMode2 = L
17:        end if
18:        if Err3(L) < BestMode3Score then
19:            BestMode3 = L
20:        end if
21:    end for
22: end for
23: Return BestMode1, BestMode2, BestMode3
```

bound is chosen as the best fitting. Err2 measures the sum of all points' distance to the line. Err2 chooses the line, which has the smallest sum, as the best fit. Err3 measures the sum of squared distance. The error functions 2 and 3 are similar, but their results could be different.

All of the three error functions choose the line that is calculated from two points in the eye point set. It is possible that a better-fit line would not pass any two points. Therefore, we introduce another function SLR to generate the line. The function SLR is used to fit a straight line through a set of points so that the sum of the squared residual of the mode is as small as possible. Suppose there are n eye points $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$. SLR will fit a straight line $y = \alpha x + \beta$, which provides the minimum sum of squared residues (the vertical distance from a point to the line).

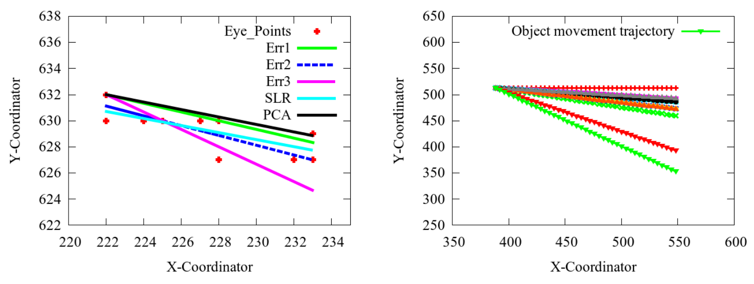
$$Find \min_{\alpha,\beta} Q(\alpha, \beta)$$

$$For Q(\alpha, \beta) = \sum_{i=1}^{n} \xi^2 = \sum_{i=1}^{n} (y_i - \alpha x_i - \beta)^2$$

The values of α and β that result in minimum Q can be computed by either using the calculus and the geometry of inner product spaces, or expanding to get quadratic in α and β:

$$\alpha = \frac{COV[x,y]}{Var[x]} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})}, \beta = y - \alpha x.$$

PCA is a statistical procedure which employs the orthogonal transformation to convert a set of observed possibly correlated data into a group of linear uncorrelated variables called principle components. In our case, PCA is used to



(a) (b)

Fig. 4: (a) The lines generated by the five functions based on one set of eye points (b) The lines generated by one function (Err2) based on 10 sets of eye points

estimate the direction of the set of eye points. Assume there are n eye points $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, the steps for PCA calculation are listed as follows:

– Calculate x' and y' as: $x'_i = x_i - \bar{x}, y'_i = y_i - \bar{y}$.
– Construct covariance matrix M

$$\begin{pmatrix} COV[x', x'] & COV[x', y'] \\ COV[y', x'] & COV[y', y'] \end{pmatrix}$$

– Calculate the eigenvalues and eigenvectors of the matrix. The eigenvector of the highest eigenvalue is the principle component of the data set.
– Assume the eigenvector is $\begin{pmatrix} x' \\ y' \end{pmatrix}$. The straight line's slope is the value of $\frac{y'}{x'}$.

To provide a detailed view of these metrices, we conduct some preliminary experiments to measure and compare them. We deploy a preliminary eye tracking prototype on Google Nexus 4 running Android 4.2.2. There is only one object moving on the screen. The object's moving distance is set as 300 pixels on the screen. The moving speed is 200 pixels per 1000 ms. The object moves on the screen with 45 degree. A volunteer eye-tracks the moving object for 10 times. The gaze point is used for eye tracking in the previous work. Considering the low resolution of front camera and the hand tremble during eye tracking, the gaze points could be unreliable for smartphone authentication. Therefore, we utilize eye points to identify the eye movement.

There are 10 movements corresponding to 10 sets of eye points. The average range of eye points' x coordinate is 19.4 ± 10.26 pixels; that of y coordinate is 7.9 ± 3.78 pixels. Figure 4a shows the straight lines which fit the eye points of one movement. The x-range is 11 and y-range is 5. It is clear that the lines generated by the five different methods can reflect the eye movement trajectory. Figure 4b shows the object movement trajectory and the lines generated by RANSAC Err2 using the 10 sets of eye points.

From this figure, we can see that the user's eye movement basically follows the object movement. In other words, the eye movement trajectory is similar to

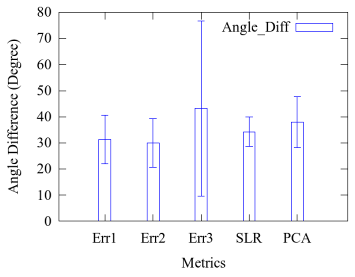


Fig. 5: The average angle difference between the trajectory of eye movement and that of object movement

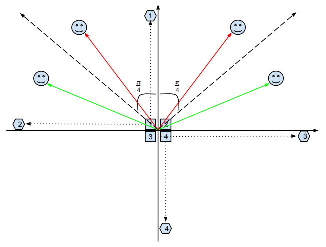


Fig. 6: When there are any four successful matches, we relax the match constraint for the remainder match.

the object movement trajectory. However, we also observe that the user's eye movement cannot strictly follow the object movement. There are three possible reasons. First, the eye tracking technique cannot guarantee 100% accuracy; second, distraction causes the user to move her eyes to a different direction; and third, the head and hand trembles impact the eye tracking. The average correlation of the 10 set data is 0.74 ± 0.1. The value is the sum of x-coordinate's and y-coordinate's correlation. Figure 5 depicts the angle difference between the eye movement and the object movement. The smaller the angle difference, the eye movement is more similar to the object movement.

3.3 Majority Vote

We regard that the eye movement trajectory matches the target movement trajectory if the angle difference between them is less than 45 degree. However, the user's eye movement trajectory could deviate more than 45 degree from the target movement trajectory in practice. The reasons could be eye tracking's error or the distraction problem. Moreover, a user cannot control her eyes to move in an exact straight line, which is just like that a user cannot draw an exact straight line.

To tolerate these errors, we introduce the majority vote to improve detection accuracy. The majority vote mechanism works as follows: as long as there are any 4 successful matches within 5 rounds, we relax the matching condition (i.e., the angle difference) from 45 degree to 90 degree for the deviated eye movement trajectory.

For example, assume that the object "1" in Figure 6 is the target, the eye movement trajectories in red are regarded as successful matches, since their angle difference from the target movement trajectory is less than 45 degree; however, the eye movement trajectory in green, whose angle difference is larger than 45 degree but less than 90 degree, is still classified as a successful match under the relaxed matching condition. While the matching relaxation reduces the number of false rejections, it also increases the chance of false acceptance.

Using the example above, if an attacker guesses "2" as the password and her eye movement trajectory happens to fall into the north-west quadrant with the probability of 50%, it will be classified as a match. The similar situation exists when the attacker guesses "3" as the password and her eye movement trajectory falls into the north-east quadrant.

However, in the design of the majority vote, the matching relaxation happens only if the attacker has already made four successful matches. Thus, the probability that the attacker could pass the authentication by simply guessing a password is only $C_4^1 \cdot C_4^1 \cdot C_4^1 \cdot C_4^1 \cdot C_4^{(1+1/2+1/2)} = \frac{1+1/2+1/2}{4\times4\times4\times4\times4} = 0.2\%$.

4 Evaluation

We implement a prototype as an app based on Android 4.2.2. The prototype can be integrated as an option in Android's authentication setting. Currently, we use the beep sound to notify a user to look back to the screen center. In a noisy environment, we could replace the beep with vibration. We leverage the Snapdragon SDK from Qualcomm [1] to track the user's eye movement. The snapdragon can be deployed on many existing smartphones. It can extract the eye points in real time. To better evaluate and analyze the results, we record eye points and object routes into files. For future real world deployment, these functions can be easily integrated together and the data can be analyzed in real time without writing them into files.

To evaluate the effectiveness of the proposed authentication method, 21 volunteers are invited to participate our user study with age range from 24 to 33. Among them, 14 wear glasses. In the following, we first measure detection accuracy. Then we compare the performance of matching trajectories using different metrics. Finally, we assess the security of our scheme.

4.1 Experimental Setup

Our experiments consist of three parts: indoor, outdoor, and mimic attacks. The indoor experiments are conducted in a normal office environment with enough lights. It is common that people use smartphones indoors. Unlike outdoor lights, indoor lights remain stable as time goes on. So, indoors is the ideal environment for accuracy evaluation. All volunteers are involved with the indoor experiments, and each of them applies the correct password for 30 times. Users hold the smartphone in the front of their faces, and stay in a comfortable posture (either sitting or standing). They take a short break (at least 5 seconds) between two sets of experiments. After the indoor experiments are completed, we select five users to do the outdoor experiments. They perform the same operations as the indoor experiments. Two users do the outdoor experiments on a cloudy day. The other three use the smartphone under the tree shade on a sunny day. We do outdoor experiments under the tree shade because users feel uncomfortable when they look at the screen in the sun. It also results in inaccurate eye movement detection. Finally, five users are involved in the mimic attacks for security evaluation.

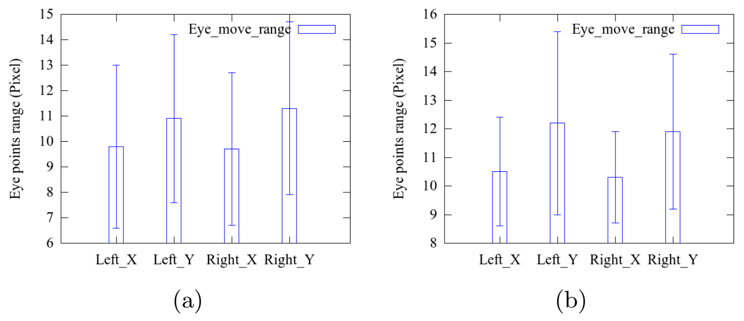


Fig. 7: The average range of eye points in indoor experiments (a) and that in outdoor experiments (b)

4.2 Detection Accuracy

Detection accuracy is the key performance indicator of an authentication method. A user could be unsatisfied if the authentication fails even when the correct password is applied. Our detection accuracy (i.e., true positive rate) is listed in Table 1.

Table 1: Accuracy of the authentication method

Users	**Trials**	**Environment**	**Left eye**	**Right eye**	**Detection accuracy**	**Detection accuracy Majority Vote**
21	630(30×21)	Indoor	1584	1566	77.1%	91.6%
5	150(30×5)	Outdoor	336	414	79.3%	97.3%

While using the RANSAC Err2 metric for matching, the detection accuracy of indoor experiments is 77.1% (486/630) and that of outdoor experiments is 79.3% (119/150). We regard that such results are reasonable, considering that neither extra eye tracker nor calibration process is required. In the previous work [22] that utilizes the front camera for eye gesture detection, five users were enrolled in the user study with the smartphone fixed on the table. Its recognition rate is just about 60%. In our evaluation, we further observe that many authentication failures only have one digit mismatch. After applying the majority vote, the detection accuracy of indoor experiments increases to 91.6% (577/630) and that of outdoor experiments increases to 97.3% (146/150).

Since we track the eye movement for authentication, users do not need to keep their heads fixed during the authentication. They can take a comfortable posture to conduct eye movements. Different postures like standing or sitting have little impact on detection accuracy. Our method can tolerate the slight head and hand tremble, because the eye point range is large enough for reflecting the eye movement trajectory. The eye point ranges are shown in Figures 7a and 7b.

As stated before, a user's eye movement trajectory of left eye could be different from that of right eye. We choose the one which is closer to the target movement trajectory for matching. There are 1584 left eye movement trajectories and 1566 right eye movement trajectories being used in the evaluation. Note that we use different eye movement data just for matching with higher accuracy. When left eye movement data is selected, it does not mean that the right eye movement data mismatches.

The accuracy of outdoor experiments during the daytime is close to that of indoor experiments. No matter it is sunny or cloudy, the accuracy does not change much. Our authentication method does not work well in weak light or dark. If there is adequate light, the number of captured eye points should be about 89. It means that we extract eye points from 16 frames in a second. The eye point number in weak light could be as low as 28, which corresponds to that 5 frames are handled in a second. It is clear that the eye point number in weak light is much less than that in normal light. This will negatively impact the line generation and match precision. Be aware that different people have different understanding of the weak light. Thus, we provide an approximation view on the connection between eye point number and light strength. Figure 8 shows the eye point number extracted at different time of a day. We can see that in most time when a user needs authentication, the light should be strong enough. We do not suggest to use this authentication method in weak light, which could cause an authentication failure. In such a case, the user could choose an alternative option, for example the pattern-based authentication.

We further classify the failures into one time failure, two consecutive failures, three consecutive failures, four and more consecutive failures. When a legitimate user suffers a failure, she will expect to pass the authentication in the next trial. The consecutive failures will frustrate the users. Table 2 demonstrates the failure statistics. There are 84 one time failures, 18 two consecutive failures, 7 three consecutive failures, and only 1 four consecutive failures. The three and more consecutive failures happen in a low probability. When majority vote is applied, there are 42 one time failures, 5 two consecutive failures, and only 1 three consecutive failures.

Table 2: Consecutive failures statistic

No majority vote	**One time failure**	**Two failures**	**Three failures**	**Four and more failures**
Number	83	36	21	4
Rate	13.2%	5.7%	3.3%	0.6%
Majority vote	**One time failure**	**Two failures**	**Three failures**	**Four and more failures**
Number	40	10	3	0
Rate	6.3%	1.6%	0.5%	0%

4.3 Effectiveness Comparison

Correlation and PCA are used to estimate the eye movement direction in previous works. Besides these two metrics, we further consider four additional methods to fit a line into the eye points and compare the angle difference between the fitted line and the target movement trajectory. Thus, in total we use six metrics to measure the similarity between the eye movement trajectory and the tar-

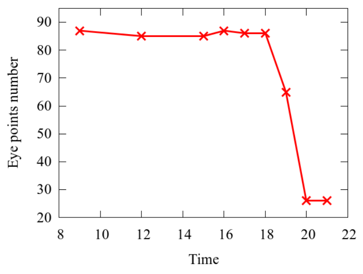


Fig. 8: The number of eye points captured at different time

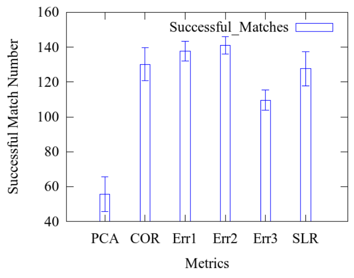


Fig. 9: Numbers of successful matches belonging to the 6 metrics

get movement trajectory. Figure 9 shows the average successful match number among the 21 users.

There are 150 (5×30) comparisons in 30 sets of experiments for one user. The correlation's average successful match number is 129.7, which corresponds to the match rate of 86.5%. RANSAC with error function 2 makes the largest number of successful matches (141), and the match rate is as high as 94%. Among the 21 users, 20 user's RANSAC ERR2 successful match number exceeds that of other metrics, and only 1 user's SLR successful match number is higher than that of RANSAC ERR2. Table 3 lists the overall successful match number and the corresponding match rate for six metrics without applying the majority vote. We note that oscillation could happen during a user's eye movement. It means that users' eyes may move back, left, right, and then forward. Correlation is calculated by the eye point sequence. Such oscillation will impact the final correlation result. However, it brings little impact on fitting a line, since all these points are still distributed along the line. It could be a reason why fitting a line achieves higher match rate than correlation. Through the comparison, we identify that RANSAC ERR2 is the most effective and reliable metric for matching among the six metrics.

Table 3: Total match of all metrics

	PCA	**Cor**	**Err1**	**Err2**	**Err3**	**SLR**
Number	1200	2724	2890	2961	2304	2661
Rate	38.1%	86.5%	91.7%	94%	73.1%	84.5%

4.4 Security Evaluation

Since our authentication method requires no hand input, it is resistant to the smudge and keylogger attacks. We try to compromise it by mounting a shoulder surfing attack. When a user is authenticating, an "attacker" peeks the process from different angles around the user. However, the "attacker" cannot figure out the password no matter standing in the front of the user or facing the user's back. In such cases, the "attacker" can only see either the objects' movements on the screen or eye movement. We observe that the password could be stolen only if the “attacker” stands at a special position—the “attacker” stands very close to the user (less in a meter) and face to the user’s one side so that the “attacker” can make slight turns to monitor both the user’s eye movement and the objects’ movements. But the user will notice the peek easily in such a special scenario. Thus, our authentication method can significantly reduce the vulnerability to shoulder surfing attacks.

To further evaluate the security of our proposed scheme, we ask 5 users to authenticate using incorrect passwords. Each user tries 15 incorrect passwords. These passwords are divided into 5 groups, one of which contains 3 passwords. Each password in the first group contains 0 correct digit. For example, if the correct password is “1-4-3-1-2”, the incorrect passwords could be “2-3-1-3-1”, “3-1-1-2-4”, or “4-1-2-3-3”. Each password in the second group contains 1 correct digit, each password in the third group contains 2 correct digits, and so forth. All incorrect passwords are generated randomly. In this set of experiments, all trails (75) fail as expected without matching relaxation. After matching relaxation is applied, there is still no false acceptance if the number of correct digits in a password is smaller than 4. The false acceptance could occur if an incorrect password contains 4 correct digits. However, as we discussed in Section 3.3, the false acceptance rate is merely 0.2% given that matching relaxation is active.

5 Discussion

To provide a comprehensive view of this work, we discuss the limitations of this work and the potential future work in this section.

Like other applications leveraging face recognition and eye tracking techniques, our work depends on adequate light. It cannot accurately track a user’s eyes in weak light or dark. We try to capture eye movements by leveraging the screen illumination in dark; however, the eye movements cannot be recorded. One possible solution is to use the infrared detector to capture the eye movements. Unfortunately, many current smartphones have not yet equipped with the infrared detector. We plan to explore this problem in the future.

Another limitation is that our method will cost longer time than passcode-based and pattern-based authentication. However, eye tracking authentication methods offer stronger security to resist those attacks discussed before. In addition, the time of conducting our authentication method for one time is about 9.6 seconds. It is shorter than the average time of existing works EyePIN (48.6 s) and EyePassShape (12.5 s) [7], which work with the assistance of desktop display.

When the eye points are clustered together, it is hard to tell the eye movement direction. A simple solution to this problem is that we just fail the authentication when either the x-range or y-range of eye points is less than a threshold. The threshold could vary from user to user.

This work is a first step towards applying the eye tracking technique in smartphone authentication. It may not be able to satisfy all because of individual difference. But it provides smartphone users a new authentication option for lowering the risks. With the development of hardware equipped on smartphone, e.g., higher resolution front camera, front infrared detector or front flash light, we believe that this authentication method can achieve higher accuracy within a shorter period.

6 Related work

Biometric information, such as fingerprint, has been used in authentication. However, researchers have shown that a fingerprint-based authentication system could be defeated [17]. Moreover, an attacker may bypass the fingerprint-based authentication system using a carefully printed fingerprint image. Human behavioral biometrics have also been used for user authentication. Keystroke dynamics have been studied as a second-factor for user identification. Each individual shows unique rhythm during keypad tapping. Zahid et al. [27] extracted six features from a user's keystrokes for individual identification. In [6], neural network classifier is utilized to distinguish impostors from legitimate users when someone dials a phone number or types text.

An abundance of sensors equipped on a smartphone can provide much valuable information on a user's tapping behaviors. The sense data from multi-touch screen, accelerometer, orientation and compass is translated to a user's gesture in work [16]. The data is used to train a classifier which can decide whether a user is legitimate. Another work [8] collects touch pressure, size, X and Y coordinates, and time as the raw data, then uses Dynamic Time Warping (DTW) algorithm to decide whether the input data matches the legitimate user's pattern. GEAT [21] authenticates a user based on behavioral features, including finger velocity, device acceleration, and stroke time extracted from users' hand input. Zheng et al. [28] proposed an authentication method based on a four-feature combination (acceleration, pressure, size, and time). Their study indicates that the four-feature combination can effectively distinguish impostors from legitimate users. Different from other works, they used one-class classifier for user verification, which only needs the legitimate user's data in training. However, the work [20] reveals that it is feasible to highly increase the equal error rate of the classifiers, which could penetrate the second level authentication methods by utilizing the data from a general population of operation statistics.

Authentication based on eye tracking can be classified into two categories. The first authenticates a user using the biometric features of the user's eyes or eye movements. The second authenticates a user based on eye movement patterns.

The biometric authentication methods in the first category extract biometric features, e.g., the distance between two eyes, light reflection, and so on, to identify a user. These features belong to physical biometrics like fingerprint. Special hardware such as the eye tracker is needed for catching a user's biometric features. Usually, a calibration process will be launched before authentication, during which the user keeps head fixed in the front of the eye tracker and stays a certain distance from the device. Bednarik et al. [3] made the first step towards using eye movements as biometric identification. They found that the distance between eyes turns out to be the most discriminative and stable indicator. However, this feature does not truly reflect the behavioral properties of eyes. The best dynamic feature is the delta pupil size, which brings 60% identification rate in this work. CUE [14] incorporates the individual and aggregated characteristics belonging to a scanpath. Using the combination of Oculomotor Plant Characteristics (OPC) and Complex Eye Movement (CEM) patterns, it can reduce the authentication error by 30% comparing to using one of them. It can also achieve the highest False Rejection Rate (FRR) 18% and False Acceptance Rate (FAR) 20% at the same time. Holland et al. [12] evaluated the effects of stimulus types and eye tracking specifications on the accuracy of biometric verification based on CEM. The work [13] presents an objective evaluation of utilizing patterns identifiable in human eye movement to distinguish individuals. The authors hypothesized that the distribution of primitive features inherent in basic eye movements could be exploited to uniquely identify a given individual. However, these works are not applicable for portable devices because it is infeasible for a user to carry an eye tracker and conduct calibration in public places.

The methods in the second category leverages gaze points as the input. To use the authentication system proposed in [15], users input password by staring at corresponding buttons on the display. Researchers also proposed to use the trajectory of eye movement as password. Since recognizing a trajectory is much easier than identifying the gaze points, these methods do not need calibration process and high resolution eye tracker. De Luca et al. [9] evaluated three different eye gaze interaction methods. They also investigated an approach on gaze gestures and compared it to the well known gaze-interaction methods. The authors of work [10] introduced three types of password patterns-ShapePass, Eye Gesture and EyePass. ShapePass allows users to easily remember complex shapes, which consist of arbitrary combinations of eight basic strokes (eight directions). Eye gesture is constructed by different gaze tracks that represent different digits. EyePass is a combination of ShapePass and EyeGesture. They mentioned that the stroke perfectly fits human eye's biometric constraint because eyes move in fast and straight saccade, and thus cannot perform any curves or other non-linear shapes. EyePassShape [7] combines EyePin and PassShape. It requires a user to remember some shape and draw the shape via eye movement actively. Unfortunately, none of these works is applicable for smartphone users.

Some recent works reveal the feasibility of exploiting the eye tracking techniques for smartphone authentication. Drewes et al. [11] evaluated eye gaze interaction as a new input method on mobile phones with the assistance of eye tracker. They compared a dwell time based gaze interaction to the gaze gesture, and found that both methods are feasible on mobile phones. The work [22] presents the first prototype of eye gesture recognition system for portable devices. The system does not need any additional hardware. It incorporates techniques of image processing, computer vision, and pattern recognition to detect eye gestures in a video recorded by the device's front camera. Normalized correlation coefficient is used as the metric which brings about 60% accuracy. Although eye gesture makes authentication robust, users cannot easily remember the complex eye gestures in practice. The work [23] introduces a novel set of shape features, which capture the characteristic shape of smooth pursuit movement over time. Each feature individually represents incomplete information about the smooth pursuit, but they can reflect the pursuit once combined. Pursuit [24] is proposed to recognize a user's eye movement when the user tracks a moving target on a big screen through eyes. It provides a general design guidance for pursuit applications.

7 Conclusion

In this paper, we propose an eye tracking authentication method for smartphone users. Unlike conventional user authentication on a smartphone, our scheme only needs a user to track a moving target on the screen through eyes. Thus, it is resistant to shoulder surfing attack, smudge attack, and many other attacks that infer a user's password based on the hand input information. In our design, the moving pattern consists of four basic strokes to reduce distraction as much as possible. Meanwhile, the object movement route is randomly changed to lower the risk of password leakage. We introduce six different metrics to measure the similarity between the eye movement trajectory and the target movement trajectory, and identify the most effective metric for development. To validate the efficacy of the proposed authentication approach, we implement a prototype on Android and conduct a user study with the help of 21 volunteers. The evaluation results show that our authentication method is able to achieve high accuracy.

References

1. https://developer.qualcomm.com/mobile-development/add-advanced-features/snapdragon-sdk-android.
2. A. J. Aviv, K. Gibson, E. Mossop, M. Blaze, and J. M. Smith. Smudge attacks on smartphone touch screens. WOOT'10, pages 1–7. USENIX Association, 2010.
3. Bednarik, K. T., M. A., and F. P. Eye-movements as a biometric. In *Proceedings of the 14th Scandinavian Conference on Image Analysis*, pages 780–789, 2005.
4. Y. Berger, A. Wool, and A. Yeredor. Dictionary attacks using keyboard acoustic emanations. In ACM CCS '06, pages 245–254. ACM, 2006.
5. L. Cai and H. Chen. Touchlogger: Inferring keystrokes on touch screen from smartphone motion. In *Hot Topics in Security HotSec*, 2011.
6. N. L. Clarke and S. M. Furnell. Authenticating mobile phone users using keystroke analysis. *Int. J. Inf. Secur.*, 6(1):1–14, Dec. 2006.
7. A. De Luca, M. Denzel, and H. Hussmann. Look into my eyes!: Can you guess my password? SOUPS '09, pages 7:1–7:12. ACM, 2009.
8. A. De Luca, A. Hang, F. Brudy, C. Lindner, and H. Hussmann. Touch me once and i know it's you!: Implicit authentication based on touch screen patterns. CHI '12, pages 987–996. ACM, 2012.
9. A. De Luca, R. Weiss, and H. Drewes. Evaluation of eye-gaze interaction methods for security enhanced pin-entry. OZCHI '07, pages 199–202. ACM, 2007.

10. R. De Luca, A.and Weiss, H. Humann, and X. An. Eyepass-eye-stroke authentication for public terminals. In M. Czerwinski, A. M. Lund, and D. S. Tan, editors, *CHI Extended Abstracts*, pages 3003–3008. ACM, 2008.
11. H. Drewes, A. De Luca, and A. Schmidt. Eye-gaze interaction for mobile phones. Mobility '07, pages 364–371. ACM, 2007.
12. C. Holland and O. Komogortsev. Biometric verification via complex eye movements: The effects of environment and stimulus. In *Biometrics: Theory, Applications and Systems (BTAS)*, pages 39–46, Sept 2012.
13. C. D. Holland and O. V. Komogortsev. Complex eye movement pattern biometrics: Analyzing fixations and saccades. Proc.IAPR ICB, 2013.
14. O. V. Komogortsev, A. Karpov, and C. D. Holland. Cue: counterfeit-resistant usable eye movement-based authentication via oculomotor plant characteristics and complex eye movement patterns. volume 8371, pages 83711X–83711X–9, 2012.
15. M. Kumar, T. Garfinkel, D. Boneh, and T. Winograd. Reducing shoulder-surfing by using gaze-based password entry. SOUPS '07, pages 13–19. ACM, 2007.
16. L. Li, X. Zhao, and G. Xue. Unobservable re-authentication for smartphones. In *NDSS*. The Internet Society, 2013.
17. T. Matsumoto, H. Matsumoto, K. Yamada, and S. Hoshino. Impact of artificial "gummy" fingers on fingerprint systems. volume 4677, pages 275–289, 2002.
18. E. Miluzzo, A. Varshavsky, S. Balakrishnan, and R. R. Choudhury. Tapprints: Your finger taps have fingerprints. MobiSys '12, pages 323–336. ACM, 2012.
19. E. Owusu, J. Han, S. Das, A. Perrig, and J. Zhang. Accessory: Password inference using accelerometers on smartphones. HotMobile '12, pages 9:1–9:6. ACM, 2012.
20. A. Serwadda and V. V. Phoha. When kids' toys breach mobile phone security. In ACM CCS '13, pages 599–610. ACM, 2013.
21. M. Shahzad, A. X. Liu, and A. Samuel. Secure unlocking of mobile touch screen devices by simple gestures: You can see it but you can not do it. MobiCom '13, pages 39–50. ACM, 2013.
22. V. Vaitukaitis and A. Bulling. Eye gesture recognition on portable devices. UbiComp '12, pages 711–714. ACM, 2012.
23. M. Vidal, A. Bulling, and H. Gellersen. Detection of smooth pursuits using eye movement shape features. In *ETRA*, pages 177–180. ACM, 2012.
24. M. Vidal, A. Bulling, and H. Gellersen. Pursuits: Spontaneous interaction with displays based on smooth pursuit eye movement and moving targets. UbiComp '13, pages 439–448. ACM, 2013.
25. Y. Xu, J. Heinly, A. M. White, F. Monrose, and J. Frahm. Seeing double: reconstructing obscured typed input from repeated compromising reflections. In ACM CCS '13, pages 1063–1074. ACM, 2013.
26. Z. Xu, K. Bai, and S. Zhu. Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors. WISEC '12, pages 113–124. ACM, 2012.
27. S. Zahid, M. Shahzad, S. A. Khayam, and M. Farooq. Keystroke-based user identification on smart phones. RAID '09, pages 224–243. Springer-Verlag, 2009.
28. N. Zheng, K. Bai, H. Huang, and H. Wang. You are how you touch: User verification on smartphones via tapping behaviors. IEEE ICNP, 2014.
29. L. Zhuang, F. Zhou, and J. D. Tygar. Keyboard acoustic emanations revisited. *ACM Trans. Inf. Syst. Secur.*, 13(1):3:1–3:26, Nov. 2009.