SCREENID: Enhancing QRCode Security by Utilizing Screen Dimming Feature

Guangtao Xue, *Member, IEEE*, Yijie Li, *Student Member, IEEE*, Hao Pan, *Student Member, IEEE*, Lanqing Yang, Yi-Chao Chen, *Member, IEEE*, Xiaoyu Ji, *Member, IEEE*, and Jiadi Yu, *Senior Member, IEEE*

***Abstract*— Quick response (QR) codes have been widely used in mobile applications, especially mobile payments, such as Alipay, WeChat, PayPal, etc due to their convenience and the pervasive built-in cameras on smartphones. Recently, however, attacks against QR codes have been reported and attackers can capture a QR code of the victim and replay it to achieve a fraudulent transaction or intercept private information, just before the original QR code is scanned. In this study, we enhance the security of a QR code by identifying its authenticity. We propose SCREENID, which embeds a QR code with information of the screen which displays it, thereby the QR code can reveal whether it is reproduced by an adversary or not. In SCREENID, PWM frequency of screens is exploited as the unique screen fingerprint. To improve the estimation accuracy of PWM frequency, SCREENID incorporates a model for the interaction between the camera and screen in the temporal and spatial domains. Extensive experiments demonstrate that SCREENID can differentiate screens of different models, types, and manufacturers and thus improve the security of QR codes.**

***Index Terms*— Screen-camera communication, secure QR code.**

I. INTRODUCTION

QUICK response (QR) codes are barcodes comprising white and black blocks. In the past years, with the pervasive built-in cameras on smartphones, QR codes have been widely adopted in mobile applications such as communication, payment, etc. Especially, for mobile payment scenarios, the QR code system (hereafter we name a QR code system as QRCode) is almost a standard module for service providers such as AliPay, WeChat, PayPal, etc [1], [2]. Users just need to show their QR codes on smartphones for a quick transaction, which provides convenient and friendly experience.

Manuscript received 19 June 2021; revised 13 December 2021 and 8 July 2022; accepted 21 August 2022; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor K. Ren. This work was supported in part by the NSFC under Grant 62072306 and Grant 61936015 and in part by the Startup Fund for Youngman Research at Shanghai Jiao Tong University and Program of Shanghai Academic Research Leader under Grant 20XD1402100. *(Guangtao Xue and Yijie Li contributed equally to this work.) (Corresponding author: Guangtao Xue.)*

Guangtao Xue, Yijie Li, Hao Pan, Lanqing Yang, Yi-Chao Chen, and Jiadi Yu are with the Department of Computer Science and Engineering, School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: gt_xue@sjtu.edu.cn; yijieli@sjtu.edu.cn; panh09@sjtu.edu.cn; yanglanqing@sjtu.edu.cn; yichao@sjtu.edu.cn; jiadiyu@sjtu.edu.cn).

Xiaoyu Ji is with the College of Electrical Engineering, Zhejiang University, Hangzhou, Zhejiang 310027, China (e-mail: xji@zju.edu.cn).

Digital Object Identifier 10.1109/TNET.2022.3203044

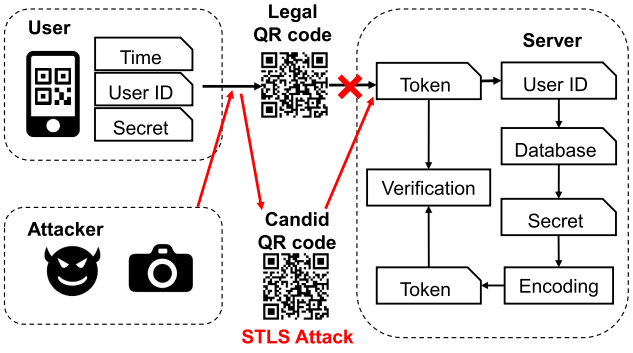


Fig. 1. A flowchart of a typical QRCode system for mobile transaction. Note that an adversary can capture the victim's QR code and then replay it for a fraudulent transaction.

The QRCode system works in a straightforward way. As shown in the flowchart in Fig. 1, payment transactions begin with the generation of a legal QR code, which includes the ID of the user in the application, e.g., the AliPay account, a time-stamp, and the secret transaction information in the form of an encrypted token. The user then presents the QR code to the cashier to have it scanned into the transaction system, together with other transaction information such as total amount and currency type. On the server side, the merchant extracts the token from QR code, obtains the user ID from the token and then accesses the database maintained by the company to retrieve the stored secret of the payer. This secret is then converted into a new token for verification. As long as the token is deemed valid, the transaction proceeds.

However, the above-mentioned transaction process is far from secure. Recently, researchers have reported that a QRCode system is susceptible to the Synchronized Token Lifting and Spending (STLS) attack and other similar attacks [3]. In this attack, the adversary first acquires an image of the QR code displayed on the victim's device [4], when the victim is showing the QR code to the cashier, for example. Then the adversary replays the stolen QR code for another transaction, which we call fraudulent transaction. To ensure the success of such an attack, the adversary should also finish the transaction before the legitimate scanning process from the cashier.

To against the STLS attack, researchers have proposed a lot of solutions. Most existing protection schemes aim at

safeguarding the QRCode system by inserting codewords [5], [6], [7] or concealing the original QR codes using visually cryptographic techniques [8], [9], [10], [11], [12], [13]. Unfortunately, attackers can still succeed in replaying the stolen QR code as they do not need to decrypt the message embedded in the QR code.

In this paper, we sought to enhance the security of the QRCode system by identifying the authenticity of a QR code. We propose SCREENID, which embeds a QR code with information of the screen that is displaying it, thereby the generated QR code can reveal whether the screen is corresponding to it. In SCREENID, we utilize the pulse width modulation (PWM) frequency of screens as the unique screen fingerprint. From our experiments, it can hardly find any two smartphones with the same PWM frequency. PWM frequency makes a good candidate for screen fingerprint for the reason that it is adjusted to different value by screen manufacturers. Even for the same manufacture, the PWM frequency shows variances due to the variations in the manufacturing processes. In Section IV, we show that PWM frequencies are different even for phones of the same model. Although the results cannot fully guarantee the security of QRCode system, the probability that any two mobile phones have the same frequency is negligible, therefore SCREENID can successfully enhance the security of QRCode system.

On the receiver side, e.g., the camera in a cashier, the PWM frequency is measured by the rolling shutter effect observed on the cashier's camera. The QR code is legal and the transaction proceeds only if the PWM frequency embedded in the QR code and the one measured from the screen are closely match.

In realizing the SCREENID, we overcome several challenges. First, the interaction between screen and camera involves many influencing factors, such as PWM dimming and screen refresh rate, etc. Obtaining an accurate frequency estimation requires that the interactions among these factors be modeled in the temporal as well as the spatial domains (Sec. VI). Second, it was necessary to compensate for variations of distance and angle between the display device and the scanning device, e.g., a camera (Sec. VII-F). Third, we found that the differences in PWM frequency among smartphones were really small (e.g., $0.1Hz$), particularly when dealing with smartphones of the same model. Thus, we need to increase the frequency resolution, while minimizing the number of captured images (Sec. VII-G).

We verified the efficacy of a SCREENID prototype using 50 screens under a variety of camera settings and environmental conditions. The usability of the proposed system was also verified in experiments involving ten participants of various ages. We summarize the contribution of this paper as follows:

- We demonstrated the feasibility of using PWM frequency of screens for fingerprint, and the fingerprint can be embedded in to a QR code to check its authenticity. From our dataset, 99.3% pairwise frequency differences of screens are larger than $0.1Hz$.
- We proposed SCREENID, which incurs no additional hardware for the QRCode system and has no requirements for user behavior. SCREENID can be implemented via a simple software update.
- We proposed to model the interaction between the camera and the screen in both temporal and spatial domains and achieved high estimation accuracy of PWM frequency, i.e., within $0.03Hz$.
- We conducted exhaustive experiments and demonstrated the efficacy of the SCREENID system under a variety of operating conditions. The evaluation shows the identifiable success rate (i.e. True Positive Rate) is above 94.3% and the wrongly accepted rate (i.e. False Positive Rate) against attack attempts is below 0.8%.

II. RELATED WORK

A. Visible Light Communication

Many works exploited the interaction between the light and camera. The visual microphone [14] used the rolling shutter effect to measure the vibration of objects to reconstruct the speech. Danakis *et al.* [15] exploited the rolling shutter effect to increase data rates in light-to-camera communications. LiTell [16], iLAMP [17], and Pulsar [18] have applied visible light communication to indoor localization based on frequencies specific to individual light sources. LiShield [19] uses the flickering of smart LEDs to protect visual privacy. ChromaCode [20] achieves unobtrusive, high-rate, screen-camera communication by considering perceptually uniform color space and design a novel adaptive embedding scheme accounts for pixel brightness and texture. Rainbar+ [21] optimized the layout for color barcodes to improve encoding capacity of each barcode, and realized a robust high-goodput visual communication system using color barcodes with real time feedback.

Our work also relies on the rolling shutter effect to capture the light frequency. Different from previous works which dealt with a single light source (e.g., a light bulb or a LCD screen), one of the challenges we faced is that many light sources (i.e., many pixels on OLED screen) flickering asynchronously at a given frequency. Obtaining accurate frequency estimation requires modeling interactions in temporal and spatial domains.

B. Visual Cryptographic Security

Most previous studies have focused on the encryption of authentication codes [5], [6], [7], [22] directly within barcodes. Chen [7] embedded authentication data including codewords and signatures within QR code. Nonetheless, even those researchers concede that their systems are vulnerable to Replay and STLS attacks [3] without hardware fingerprint. To avoid STLS attacks, POSAUTH [3] requires double scanning which changes users' habits. Zhou *et al.* [23] utilize various brightness pixels as hardware fingerprint, but is restrict with completely dark environment and pixel aging problem.

Visual cryptography (VC) techniques [24] have also been developed to ensure that messages remain concealed. Essentially, a secret image is embedded within shared images aimed at camouflaging the hidden data, which can be revealed only through the stacking of multiple shared images or capturing with a specific approach [8], [9], [10], [11], [12], [13]. mQRcode [12] utilizes moire patterns to encrypt the original

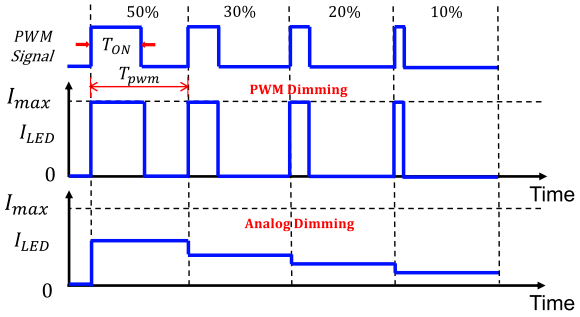


Fig. 2. An illustration of PWM dimming and analog dimming.

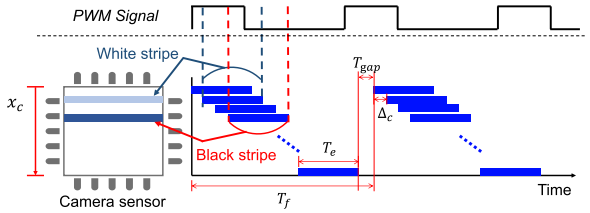


Fig. 3. Sampling performed by camera sensors. Some of the rows sample during the PWM on-time, and others sample during the off-time, respectively producing black (darker) and white (lighter) stripes in the captured image.

QR code, thereby making the information impervious to extraction unless viewed from a specific position. Nonetheless, those methods necessitate the exchange of key images with a server or restrict the user to a pre-determined position to recover the original QR code.

Unlike the methods mentioned above, our approach utilizes hardware fingerprint to avoid STLS attacks, which does not require additional equipment and allows the user complete freedom in the use of their device.

C. Screen Watermarking Schemes

Digital media requires protection through internet or other mediums. The key idea of watermarking schemes [25] is to embed an imperceptible image within the target file, which have no connection with the hardware information of victims' device. Without the connection with the hardware, the watermark can be displayed on any device, so it cannot prevent replay attacks. Most existing approaches [26], [27], [28] focus on embedding an imperceptible image on spatial domain. Abraham and Paul [28] used the RGB color space to conceal watermark signals within color images. Riad *et al.* [29] and Gourrame *et al.* [30] proposed a robust watermarking method based on Discrete Fourier Transform (DFT). Thongkor and Amornraksa [31] presented a watermarking method that remains robust even after printing. These watermarking techniques are utilized to ensure the ownership or the source of an image. However, attackers need not to erase the watermarks, but only to directly show the image containing the watermark to achieve the attacks. In other words, it need a two-factor authentication method to verify if the watermark and the ownership of the QR code is correspond. Therefore, the attackers do not need to erase the watermark, but directly show the image containing the watermark to achieve the attack. Only a single watermarking scheme does not work in preventing from Replay and STLS attacks.

III. BACKGROUND

A. Pulse Width Modulation (PWM) Dimming Control

There are two mechanisms commonly used to control screen brightness. One is analog dimming [32], which adjusts screen brightness by regulating the voltage output from a DC power supply (i.e., DC dimming) (Fig. 2). The other is pulse width modulation (PWM) dimming [32] which adjusts screen brightness by digital encoding an analog signal to appear for a given period of time, wherein the power is either fully on or fully off. Essentially, the screen brightness is adjusted by regulating the on-time (T_{ON}) in each cycle (T_{pwm}) [33]. Since human eyes are insensitive to changes of high frequency, we do not notice the flickering, but rather perceive a difference in brightness. Essentially, the screen appears brighter when the on:off ratio is larger.

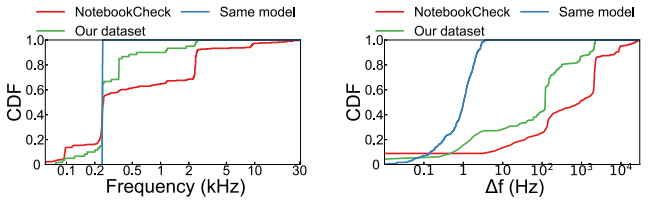
PWM dimming has several advantages over analog dimming. PWM dimming does not impose chromatographic shifts (color cast due to lighting unit response sensitivity), as the current is always equal to the full-amplitude current I_{max} or 0. PWM also permits high dimming resolution over a wide range of values. This has led to the widespread adoption of PWM dimming for OLED screens. Note that it is rapidly gaining popularity for mobile devices.

B. Rolling Shutter Camera

Complementary metal-oxide semiconductor (CMOS) technology is widely used to fabricate the cameras used in modern smartphones. The sequential sampling by different rows in the camera sensor is referred to as a rolling shutter [35]. Fig. 3 illustrates the sampling process when capturing a video. The latency between the start of each exposure in each row is denoted by Δ_C. Note that Δ_C remains a constant regardless of the ISO and exposure time and whether the sensor is used to capture a still image or video segment. The sampling rate f_c of a rolling shutter camera is calculated as follows: $f_c = 1/\Delta_C$.

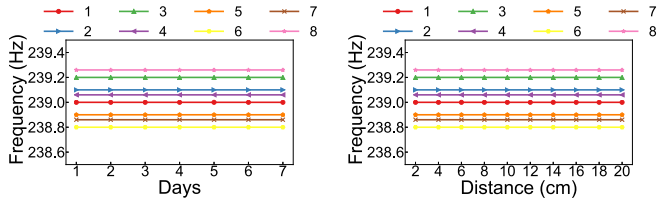
While capturing a video, the time during which one frame is captured is denoted by T_f. Thus, for a video at $30fps$, $T_f = 1/30$. Note that there is a time gap between the end of the last row in one frame and the start of the first row in the following frame. The time gap can be derived as follows: $T_{gap} = T_f - T_e - (X_c - 1)\Delta_C$, where X_c indicates number of rows of camera sensor and T_e represents the exposure time.

When using a rolling shutter sensor to capture an image from a screen with PWM-induced flicker, the light intensity indicates the total number of photons received by a given row throughout the duration of the exposure. Fig. 3 shows the black (darker) and white (lighter) stripes created by the sensor rows, while recording an image in which PWM signals periodically turn the screen on and off. Examples of this phenomenon are shown in Fig. 6. Details pertaining to the modeling of the stripes and their use in computing the PWM frequency are presented in Section VI.



(a) PWM frequencies. (b) Pairwise PWM frequency differences.

Fig. 4. CDF of PWM frequencies and pairwise differences of 300 screens reported in NotebookCheck [34] and 50 screens (30 are of the same model) we collected.



(a) Time. (b) Distance.

Fig. 5. We measure the PWM frequencies of 8 screens of the same model across days and at various distances to show its stability.

IV. Preliminary Study

If PWM dimming frequency is to be used as a feature by which to identify screens, then we must first verify its uniqueness and stability.

A. Uniqueness of PWM Frequency

We first conducted a preliminary study to assess the uniqueness of PWM frequencies across screens from different and the same models.We used a light sensor ADPD2212 [36] sampling at $80kHz$ to measure the PWM frequencies of 50 phone screens. We crawled 300 screen benchmarks collected by NotebookCheck [34] (covering mainstream smartphones from 2016 to 2019). Fig. 4(a) illustrates the distribution of PWM frequencies among smartphone screens. We can see that 97% of the PWM frequencies were below $10kHz$ and 68.3% were below $2kHz$. Note that most current smartphones support video capture at 1920×1080 using a frame rate of $30fps$. This means that it should be possible to achieve sampling rates of at least $32.4kHz$, which is far beyond the Nyquist sampling rate ($> 2 \times 10kHz$).

Fig. 4 shows the CDF of the pairwise differences in PWM frequency among screens. Note that the frequency resolution in the NotebookCheck dataset is $0.1Hz$ so it cannot distingush screens using a close PWM frequency. In our dataset, the frequency resolution is $0.01Hz$. We can see that among all screens and screens of the same model, 95% pairwise differences are larger than $1.1Hz$ and $0.18Hz$, respectively. The results revealed that a frequency resolution of $0.1Hz$ should be sufficient to differentiate among 99.3% screens.

B. Stability of PWM Frequency

We also assessed the uniformity of PWM frequencies under various conditions. Fig. 5 shows the PWM frequencies of 8 screens (Samsung S7 [37]) of the same model measured in various days and from various distances. The variation in PWM frequency was at most $0.01Hz$ which implies the PWM frequency is stable.

V. Threat Model

In this paper, we aim to enhance the security of a QR code by embedding the screen fingerprint into the displayed QR code. An adversary may destroy the security by launching a STLS attack as following:

An adversary can intercept a QR code without the awareness of the victim during a transaction, for example. Then the stolen QR code can be replayed by the attacker to achieve a successful fraudulent transaction. The whole replay attack process should be finished before the token expires. The attackers may interrupt or delay the legitimate progress for attack by using some social-engineering methods,e.g., talking to the victim or the cashier [3], [38], [39].

We make the following assumptions on the adversary.

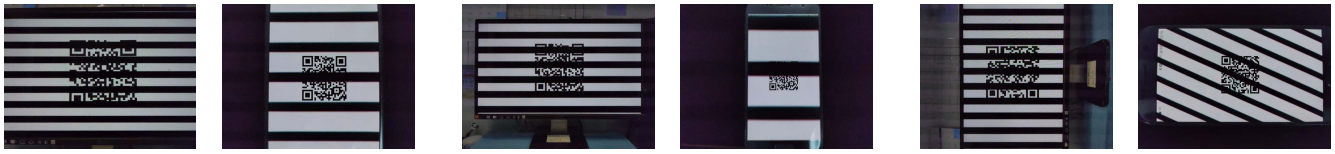
- **No access to the victim device**. We consider the adversary can physically obtain the QR code displayed on the victim device from a certain distance. For example, she may capture the QR code using a smartphone or a dedicated camera present at the payment scene.
- **Malware attack**. We assume that the adversary could install a malicious app to screenshot or take camera permissions to obtain the QR code. For instance, the reflection of QR code on the glass of POS scanner may be sniffed by the malicious app using front camera [3].
- **No limitation to software and hardware.** The adversary may utilize state-of-the-art image photographing, recording and synthesis software techniques. She can also utilize any devices such as high-quality digital cameras and high-speed networks for a replay attack.
- **No cooperation with payment apps or the cashier.** We assume that the adversary cannot cooperate with payment apps to decrypt the key information encoded in the QR code. We also assume that the adversary cannot cooperate with the cashier as he cannot obtain the camera configuration.

VI. Measuring PWM Frequency Using a Camera

In this section, we examine the process of measuring PWM frequency using rolling shutter effect. Note that our use of the rolling shutter for frequency measurement must contend with many light sources flickering asynchronously. We addressed this issue by modeling the screen-camera interaction in temporal as well as spatial domain.

A. PWM Dimming of LCD and OLED

LCD and OLED screens both use PWM dimming control; however, the actual dimming methods differ. LCD screens use a single backlight, such that the screen can be regarded as a single light source flickering at its PWM frequency. Figs. 6(a)(c)(e) show the black and white bands caused by the interaction between LCD screen flicker and the rolling shutter. Note the lack of variation in the width and angle of the bands despite variations in the distance and angle between



(a) lcd screen($20cm$) (b) oled screen($5cm$) (c) lcd screen($40cm$) (d) oled screen($10cm$) (e) lcd screen($90°$) (f) oled screen($90°$)

Fig. 6. Images of LCD and OLED screens captured using a camera. The position of the camera was fixed, while the position and direction of the screens was varied. The stripes in the LCD screen remained the same in all cases, whereas they varied in the OLED screen in terms of width and angle.

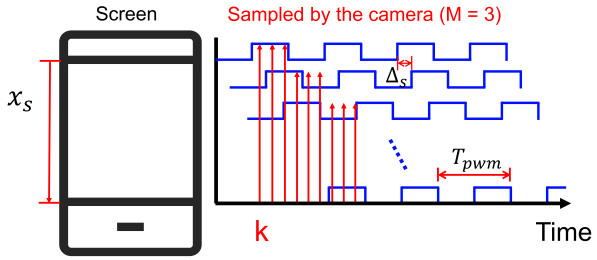


Fig. 7. Asynchronous PWM dimming in OLED screens and example of sampling using a rolling shutter camera where the projection ratio $M = 3$.

the camera and screen. This indicates the estimates of flicker frequency vary only as a function of rolling shutter speed.

By contrast, the bands in Figs. 6(b)(d)(f) from an OLED screen varied in terms of width and angle, depending on the position of the screen. This can be attributed to the fact that each pixel in an OLED screen is controlled by an LED light source and the PWM cycle of each row is asynchronous. As shown in Fig. 7, the OLED pixels are grouped into rows, the flickering of which is delayed by latency Δ_S from the previous row. Note that PWM latency Δ_S is kept constant among rows in order to prevent fluctuations[1] in current. [40]

B. Using a Camera for Sampling

The fact that the brightness of OLED screens varies as a function of time and space means that the rolling shutter effect can be viewed as a process of sampling in the temporal and spatial domains. PWM dimming control produces square wave signals. Since we are primarily interested in frequency f_{pwm}, without a loss of generality, a sinusoidal wave can be used to describe the PWM signals we are interested in, as follows:

$$T(x_s, t) = \cos(2\pi f_{pwm}(t + x_s\Delta_S) + \theta) \tag{1}$$

where $T(x_s, t)$ denotes the signal emitted from the x_s^{th} row of the screen at time t. x_s ranges from $0, 1, \ldots, X_s - 1$ where X_s indicates the number of rows across the entire screen. θ denotes the initial phase of the signal in the first row.

When using a rolling shutter camera to capture an image from an OLED screen, the screen with X_s rows is mapped to $X_{s\to c}$ rows in the captured picture (as shown in Fig. 8). In other words, we assume that M rows on the camera sensor capture one row from the screen, where the projection ratio

[1]A pixel requires a larger current during its PWM duty cycle. By introducing a phase latency between rows, the total current required by the all pixels remain more stable across time.

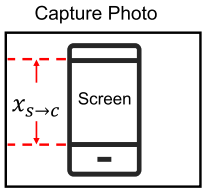


Fig. 8. Projection ratio between screen and camera.

M can be denoted as follows:

$$M = \frac{X_{s\to c}}{X_s}$$

Fig. 7 presents an example of image sampling in which the screen and camera are placed in parallel to induce flicker and sampling is performed sequentially from top to bottom. Note that we later remove this assumption and model the interaction under arbitrary screen placements. The red arrows indicate the samples obtained by individual camera rows. $M = 3$ indicates that three camera rows capture the same row on the screen. Assuming that the camera begins sampling at time k, the captured signals can be written as follows:

$$\begin{aligned} R(x_c, t) &= \cos(2\pi f_{pwm}((t + k + x_c\Delta_C) + x_s\Delta_S) + \theta) \\ &= \cos(2\pi f_{pwm}(t + k) + 2\pi f_{pwm}(x_c\Delta_C \\ &\quad + \lfloor\frac{x_c}{M}\rfloor\Delta_S) + \theta) \end{aligned} \tag{2}$$

where $R(x_c, t)$ represents the signal received at camera row x_c and time t. x_c ranges from $0 \ldots X_c - 1$ where X_c indicates the number of camera rows. Δ_C represents the rolling shutter interval (Fig. 3).

Thus, the intensity of pixels in x_c row, $I(x_c)$, is the sum of photons received throughout the entire exposure, which can be derived as follows:

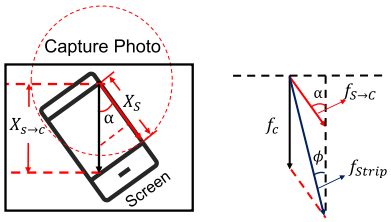
$$I(x_c) = \int_{t=k+x_c\Delta_C}^{k+x_c\Delta_C+T_e} R(x_c, t)dt \tag{3}$$

where T_e represents the exposure duration.

C. Sampling Under Arbitrary Screen Placement

It would be unreasonable to expect all cellphone users to hold their devices at precisely the same angle while the QR code is being read. Thus, we considered sampling with the mobile device held at an arbitrary angle relative to the camera. Fig. 9(a) shows the situation in which the screen is placed at angle α relative to the direction of the rolling shutter. For this arbitrary case, we can obtain a new projection ratio as follows:

$$M = \frac{X_{s\to c}\cos\alpha}{X_s} \tag{4}$$



(a) Projection ratio (b) Stripe angle

Fig. 9. Projection ratio and resulting stripe angle in arbitrary direction.

As in Eq.2, the received signal from line x_c at cross-angle α ($0° \leq \alpha \leq 360°$) can be modeled as follows:

$$R(x_c,t,\alpha) = \cos(2\pi f_{pwm}(t+k) + 2\pi f_{pwm}(x_c\Delta_C + \lfloor\frac{x_c cos\alpha}{M}\rfloor\Delta_S) + \theta)$$

The intensity of pixels in row x_c indicated by $I(x_c)$ throughout the exposure can be derived as follows:

$$I(x_c) = \int_{t=k+x_c\Delta_C}^{k+x_c\Delta_C+T_e} R(x_c,t,\alpha)dt$$

Specifically, the stripe pattern changes as a function of device rotation and distance from the camera (Sec. VI-A). Fig. 9(b) indicates the variation in angle due to rotation and projection ratio M. The scanning performed by the rolling shutter camera and the PWM signal can be considered frequency vectors f_c and f_s, which are derived as follows:

$$f_c = \frac{1}{\Delta_C}, f_s = \frac{1}{\Delta_S}$$

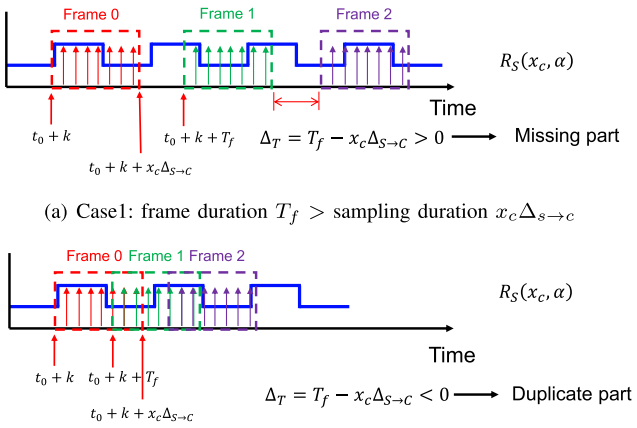
Due to the degree of projection ratio between the screen and camera, it is possible to convert the actual PWM signal frequency vector into a capture coordinate as $f_{S\to C} = f_s M$. The combined stripe pattern vector (denoted by f_{Strip}) is shown in Fig. 9(b). Thus, the resulting angle of the stripe patterns relative to the horizontal (denoted by ϕ) can be derived as follows:

$$\tan\phi = \frac{f_{S\to C}\cos\alpha + f_c}{f_{S\to C}\sin\alpha} = \frac{\Delta_C M\cos\alpha + \Delta_S}{\Delta_C M\sin\alpha} \quad (5)$$

when $\alpha = 0°$ or $180°$, the refresh direction is the same or the inverse direction of the rolling shutter, such that the stripe patterns are aligned parallel to the bottom of the image.

D. Sampling Using Multiple Frames

The frequency resolution is limited by the length of the data; If we capture only one photo, the frequency resolution is insufficient to determine PWM frequencies of different screens. One intuitive solution is to concatenate multiple frames of a video. However, concatenating frames from an OLED screen is a nontrivial problem, due to the presence of gaps between the frames. The start of exposure in the first row in the previous frame is denoted as t, whereas the start of exposure in the first row in the subsequent frame is $t+T_f$, where T_f denotes the frame duration, as shown in Fig. 3.



(a) Case1: frame duration T_f > sampling duration $x_c\Delta_{s\to c}$

(b) Case2: frame duration T_f < sampling duration $x_c\Delta_{s\to c}$

Fig. 10. Problems associated with desynchronization due to fluctuations in frame rates and sampling intervals.

Thus, the received signal associated with continuous captures through multiple frames can be denoted as follows:

$$R(x_c,t,\alpha,n) = \cos(2\pi f_{pwm}(t+nT_f+k) + 2\pi f_{pwm}(x_c\Delta_C + \lfloor\frac{x_c\cos\alpha}{M}\rfloor\Delta_S) + \theta) \quad (n\in 0,1,2,\ldots) \quad (6)$$

where n denotes the frame number beginning at 0.

Note that Eq. 6 can be treated as a spatial frequency related to row index x_c at specific time $t+nT_f+k$. $\cos(2\pi f_{pwm}(t+nT_f+k)$ can be regarded as the initial phase in the spatial domain, denoted by θ_0. Thus, the received signal in spatial domain R_S can be modeled as follows:

$$R_S(x_c,\alpha) = \cos(2\pi f_{pwm}(x_c\Delta_C + \lfloor\frac{x_c\cos\alpha}{M}\rfloor\Delta_S) + \theta_0 + \theta) \quad (7)$$

By transforming Eq. 7 into the time domain using one sample per line, we can derive the true sampling interval in the communication between screen and camera as follows:

$$\Delta_{s\to c} = \frac{x_c\Delta_C + \lfloor\frac{x_c\cos\alpha}{M}\rfloor\Delta_S}{x_c} \quad (8)$$

where the sampling rates for extracting frequency is denoted by $f_s = 1/\Delta_{s\to c}$.

However, variations among cameras in terms of rolling shutter effects lead to differences in Δ_C and interval Δ_S between the lines in the PWM signal. Variations in frame rates and sampling intervals can also lead to *information duplication* or *information loss*. Essentially, situations involving desynchronization can be divided into two cases:

- **Case1:** Frame duration T_f exceeds the sampling duration $x_c\Delta_{s\to c}$, leading to information loss.
- **Case2:** Frame duration T_f is shorter than the sampling duration $x_c\Delta_{s\to c}$, leading to information duplication.

Fig. 10 presents an example of desynchronization due to variations in frame rates and sampling intervals in three continuous frames. As shown in Fig. 10(a), there exists a time gap $\Delta_T = T_f - x_c\Delta_{s\to c}$ between frame 0 and frame 1 when $\Delta_T > 0$. The signal transmitted during this gap period is not

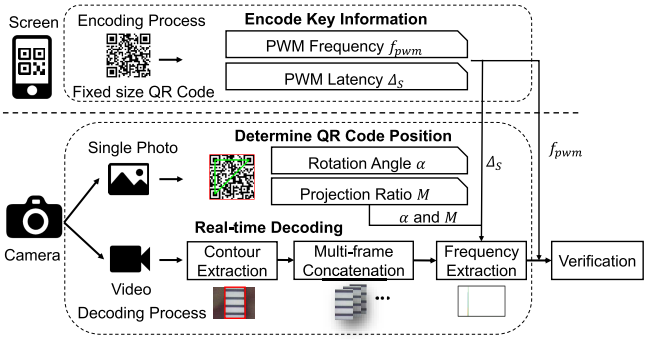


Fig. 11. Overview of SCREENID system.

detected in any of the received frames, such that the captured video is non-continuous (incomplete).

Information duplication can occur in two sequential frames when $\Delta_T < 0$. As shown in Fig. 10(b), frame 0 and frame 1 contain a portion of the tail replicated from frame 0. In these situations, the camera receives identical stripe signals in two sequential frames. Recovering the original signal requires interpolation to deal with *information loss* or the removal of duplicated frame sections and concatenation of the remaining sections to deal with *information duplication*, where the length of the lost or duplicated data is denoted by $\frac{\Delta_T}{\Delta_{s\to c}}$. Specifically, we insert a specific number of zeroes (matching the length of the lost data) into the interval for *information loss*. Conversely, we remove a specific number of tails (matching the length of the data duplication) for *information duplication*.

VII. SYSTEM DESIGN

Fig. 11 illustrates the architecture of SCREENID system, comprising two parts: encoding by the sender (smartphone screen) and decoding by the receiver (camera).

A. Encoding

Verifying the authenticity of a screen requires that additional information be embedded in the QR code, namely (i) PWM frequency f_{pwm} and (ii) PWM latency between two rows Δ_S. The size of the QR code to be generated is adjusted automatically based on the resolution and size of the screen, such that the length of the edge is of the fixed number of pixels (580 pixels in our implementation) across screens.

Using SCREENID requires that f_{pwm} and Δ_S are both known in advance. Before the user first register his PWM frequency f_pwm, the screen needs to obtain its PWM latency Δ_S. For Δ_S, it can be obtained by being captured by a known configuration camera (cashier). We proposed a method below to estimate Δ_S (Sec.VII-B). After obtaining Δ_S, the user need to register the PWM frequency for the first time. The registration process is that, the user shows his screen for 11 seconds, then the cashier captures the video and split it into 10 pieces of 2 seconds (1 second overlap). For each piece, ScreenID system computes a f_{pwm} using proposed Frequency Extraction scheme (Sec. VII-G), then selects the median f_{pwm} among 10 measurements as the authenticated f_{pwm}, and finally encoded into the QR code encryptedly. This process only needs to be finished at the first time. The user only needs to show his smartphone for a period of time, and the rest of process is done by the software in the background.

B. Determining PWM Latency Δ_S

PWM latency Δ_S is determined from an image captured at an angle of 90° (i.e., $\alpha = 90\circ$ in Fig. 9) using a camera with a known configuration. As indicated in Eq. 5, when $\alpha = 90°$, the angle of stripe ϕ is a function of Δ_S that satisfied $\tan\phi = \frac{\Delta_S}{\Delta_C M}$, therefore $\Delta_S = \Delta_C M \tan\phi$, where Δ_C is the rolling shutter interval and M is projection ratio computed using Eq. 4.

In our preliminary analysis, we found little variation in PWM latency Δ_S among phones of the same model; therefore, it was necessary to measure Δ_S only once for each phone model. This could easily be achieved using crowdsourcing where one user of a given phone model performs the calibration and shares the results. Likewise, this information could be provided by the manufacturer when the device is first released.

C. Decoding

Verifying that the captured QR code is authenticated involves embedding the rolling shutter interval Δ_C in camera and PWM latency Δ_S in the QR code for use in estimating the PWM frequency. The QR code is accepted only if the estimated PWM frequency and the PWM frequency embedded in the QR code are a close match lower than a threshold.

The decoding process is performed in two steps: (i) determining the position of the QR code and (ii) extracting the PWM frequency.

D. Measuring Rolling Shutter Interval Δ_C

Rolling shutter interval Δ_C must be known a priori in order to compute the PWM frequency. However, we observed inaccuracies in the rolling shutter interval provided by Android Camera2 API [41]. Therefore, we programmed SCREENID to calibrate camera to obtain an accurate indication of rolling shutter interval (denoted as $\widehat{\Delta_C}$). Calibration involves capturing an image of an LCD screen with a known PWM frequency f_{pwm}. Based on Eq. 6, the wavelength (i.e., the width of a pair of black and white stripes) denoted by W_{lcd} can be derived as $W_{lcd} = 1/f_{pwm}/\widehat{\Delta_C}$. Thus, the accurate rolling shutter interval can be estimated as $\widehat{\Delta_C} = 1/f_{pwm}W_{lcd}$. Note that it is easy to calibrate for camera on cashier when carrying out factory examination.

E. Optimizing the Camera Configuration

The images obtained using the camera serve two purposes. The first purpose involves estimating the position of the QR code and extracting the information embedded. We set the camera to auto-mode to obtain the optimal configuration for ambient lighting conditions to obtain a clear image. The second purpose involves measuring the PWM frequency, which requires a clear indication of the stripes. The following configuration was adopted for capturing video frames:

(a) Original code. (b) Rotation angle. (c) Contour.

Fig. 12. Methods used to obtain the rotation angle and projection ratio using a QR code of fixed size.

Exposure time. Many smartphones use PWM dimming only under a low brightness ratio, which tends to extend the exposure time. Note however that an excessively long exposure time would allow the occurrence of more than one PWM cycle, thereby compromising stripes clarity. Thus, we configured SCREENID to operate at an exposure time of $0.3ms$.

ISO. SCREENID was configured to use the highest available ISO, as this tends to enhance contrast between the black and white stripes. High ISO operations also tend to generate noise; however, the fact that most of the noise it at lower frequencies means that it's easily filtered out using a highpass filter.

Aperture. SCREENID was configured to use the largest available aperture as this tends to blur the background, making it easier to extract the contours of the screen.

F. Determining QR Code Position

Estimating the distance and angle between the screen and the camera involves taking a single photo in auto-mode in order to detect the QR code. The position symbols in the the QR code (blue blocks in Fig. 12(b)) to form a right-angle triangle (green in Fig. 12(b), which can be used to estimate rotation angle α.

Locating the position markers makes it possible to extract the contours of the QR code (red in Fig. 12(c)) and measure the size of the QR code in the photo. The size of the displayed QR code is fixed; therefore, it is possible to compute projection ratio M as a fraction of the length of a pixel along the both sides of the original QR code.

G. PWM Frequency Extraction Using Multiple Frames

After extracting essential information from the QR code, the camera immediately switches to video mode for real-time frequency extraction. As shown in Fig. 11, this process includes four steps: Contour Extraction, Multi-frame Concatenation, Frequency Extraction, and Verification.

1) Contour Extraction: Each frame is first transformed into a gray-scale image to enhance contrast. The image then undergoes denoising and binarization, before the boundaries are detected using a function based on the Sobel operator [42].

2) Multi-Frame Concatenation: From among the multiple columns in the contour, we select the longest column as a data sample of screen in a frame to maximize the data length. The frequency resolution (f_{res}) of the Fourier transform is $f_{res} = f_{sample}/N$ [43], where f_{sample} indicates the sampling rates and N indicates the number of samples. Identifying the characteristic PWM frequency requires a frequency resolution of $f_{res} = 0.1Hz$ or smaller. This can be achieved by concatenating columns from multiple frames to increase N, as described in Sec. VI-D.

3) Frequency Extraction: For a usual condition, the camera of smartphones can sample at $f_{sample} = 80kHz$ with 4000 columns at frame rates of $30fps$. Thus, it requires $\frac{f_{sample}}{f_{res} \times 4000 \times 30} = 6.7$ seconds. Improving system efficiency requires reducing the time required to capture frames, while estimating the PWM frequency with high precision.

This was achieved by exploiting the fact that the sampling rate (e.g., $80kHz$) usually exceeds the Nyquist rate required to reconstruct PWM signals (e.g., $500Hz$). Thus, we down sample a sequence of samples of length N (e.g., $[x_1, x_2, x_3, \cdots, x_N]$) by a factor of K by taking a sample at K intervals, which results in K segments. (e.g., $[x_1, x_{K+1}, x_{2K+1}, \cdots]$, $[x_2, x_{K+2}, x_{2K+2}, \cdots]$, $\cdots$) from the original sequence. The sampling rate of each segment is becomes f_{sample}/K. We then concatenate all of the segments into a single sequence of length N, which retains the same PWM frequency as long as the new sampling rate is $\frac{f_{sample}}{K} > 2f_{pwm}$. We apply the Hanning window [44] to make the concatenation smoother before applying the Chirp-Z transform (CZT) [45] to improve spectral resolution in the frequency range of interest (i.e., f_{pwm}). We named the whole process *Down-sampling CZT* in following text.

Fig. 13 is an example illustrating the benefits of using down-sampling CZT. When there are two close frequencies, FFT is unable to differentiate between them when the frequency resolution is limited by the data length. CZT can be used to improve spectral resolution within a specific frequency range; however, zooming in at a set resolution can introduce additional errors [45]. Down-sampling CZT provides a finer frequency granularity with sharper peaks, thereby making it easier to distinguish two close peaks without introducing errors. When using down-sampling CZT in the proposed system, we can use the true PWM frequency f_{pwm} (from the information embedded in the QR code) to compute down-sampling rate K and select a frequency range for CZT.

4) Verification: SCREENID compares the estimated PWM frequency with the one embedded in the QR code. When the difference is less than a given threshold ($0.03Hz$ in current study), the QR code is accepted. Otherwise, it is rejected.

VIII. EVALUATION

A. Experiment Setup

We generate version-3 (29×29) QR codes[2] using SCREENID.The size of QR codes is 580×580 pixels. 50 smartphone screens and 5 smartphone cameras are used in our evaluation. Among 50 screens, 10 are LCD and 40 are OLED screens. 30 screens are of the same model.

Unless otherwise stated, we evaluated SCREENID as follows. For each screen, we displayed 40 QR codes where 10 embedded the correct PWM frequency of the screen for authentication while the other 30 embedded that of another screen randomly selected from 50 screens for attack.

[2]Alipay uses version-2 (25×25) QR codes [46] while WeChat uses version-1 (21×21) [46] QR codes. Version-3 QR codes which carry more data are representative of the amount of data needed by these systems.

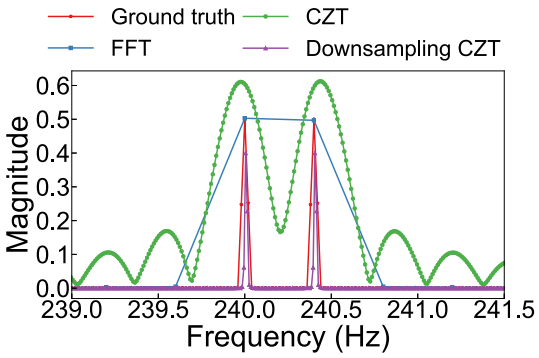


Fig. 13. Comparison of spectra between down-sampling CZT, CZT, and FFT.

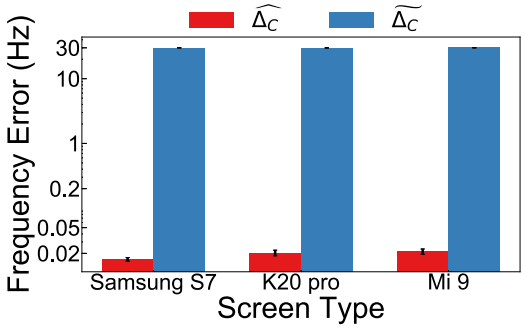


Fig. 14. Frequency error of the calibrated rolling shutter interval $\widehat{\Delta_C}$ and that from Android API $\widetilde{\Delta_C}$.

The default settings for the cameras are listed as follows: the exposure time is set $0.3ms$, the ISO is set to maximum, the aperture is set to the largest, the lens focal length keeps the same as that of auto-focus mode when taking pictures (Refer to Sec.VII-E and Sec.VIII-B). As for the settings for the screens, the brightness ratio is set 50% to ensure PWM dimming is worked. For each screen, we displayed correct and incorrect QR codes respectively to simulate legal and attack attempts. ScreenID takes a $2s$ video for each authentication or attack attempt. The threshold is set $0.03Hz$. The details of camera model is listed in Appendix. C.

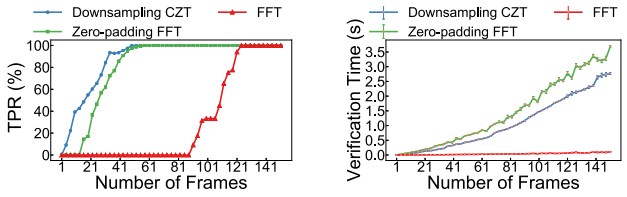
We utilize *TPR (True Positive Rate)* and *FPR (False Positive Rate)* as the metrics for performance evaluation, which are defined as follows:

- $TPR = \frac{Number\ of\ accepted\ authorized\ QR\ code}{Number\ of\ verification\ attempts}$
- $FPR = \frac{Number\ of\ accepted\ unauthorized\ QR\ code}{Number\ of\ attack\ attempts}$

The higher *TPR* is and meanwhile the lower *FPR* is, the better SCREENID performs.

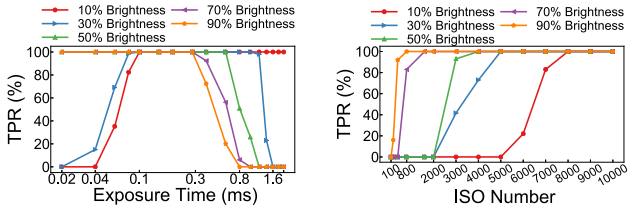
B. Microbenchmark

1) Rolling Shutter Interval Δ_C: We first evaluate the scheme to estimate rolling shutter interval Δ_C. Since Δ_C cannot be measured directly, we use Δ_C estimated by SCREENID ($\widehat{\Delta_C}$) and returned by Android API ($\widetilde{\Delta_C}$) to compute the PWM frequency and use the PWM frequency error as the metric. The smaller PWM frequency error is, the more accurate Δ_C is. We use one LCD screen to estimate Δ_C of 5 cameras. Then the 5 cameras are used to measure the PWM frequency of another 10 LCD screens.[3] For each pair of the camera and the screen, we repeat the measurement for 10 times and reported the average and standard variation in Fig. 14. We can see that the error is reduced from $30Hz$ to $0.02Hz$ using our scheme.



(a) Verification TPR. (b) Verification time.

Fig. 15. Verification TPR and time of different frequency extraction schemes.



(a) Impact of exposure time. (b) Impact of ISO number.

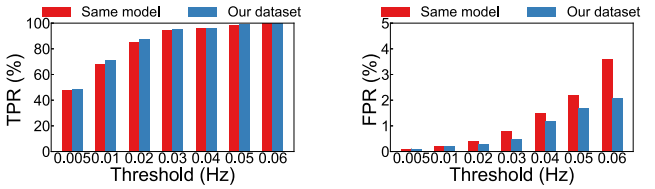
Fig. 16. The TPR on impact of exposure time and ISO number under different brightness ratio.

2) PWM Latency Δ_S: We then evaluate the scheme to estimate PWM latency Δ_S. Since Δ_S cannot be measured directly, we use the estimated Δ_S to compute the PWM frequency and use the PWM frequency error as the metric. The smaller PWM frequency error is, the more accurate Δ_S is. We use a camera with the known rolling shutter interval to estimate Δ_S of 50 screens. Then the 50 screens are used to compute their PWM frequency by another 4 cameras. We compare the PWM frequencies computed using estimated Δ_S and the true PWM frequencies. The average error is $0.02Hz$ which suggests our scheme to estimate Δ_S is accurate.

3) Number of Required Frames: Here we evaluate how many frames are required to accurately estimate the PWM frequency. We used 5 cameras to capture a 60-second video of each of 50 screens. We compare the proposed down-sampling CZT with FFT and zero-padding FFT. The average TPR under various number of frames is reported in Fig. 15(a). FFT needed 121 frames to get enough frequency resolution to achieve 100% TPR. Zero-padding FFT [47] is a well-known method to refine the frequency granularity by padding zeros. Zero-padding FFT and down-sampling CZT perform similarly and needed 65 and 60 frames, respectively. However, the computation complexity of zero-padding FFT is larger than down-sampling CZT [45], so that down-sampling CZT performs faster than zero padding FFT as shown in Fig. 15. In the following evaluation, we take a 60-frame video and used down-sampling CZT to compute PWM frequency.

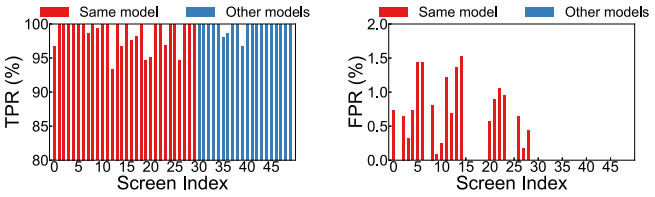
4) Camera Configuration: To find the best exposure time and ISO, we measure the TPR under various settings and show the results in Fig. 16. For the exposure time, we balance the brightness ratios and set exposure time to $0.3ms$. For ISO,

[3]PWM latency Δ_S of LCD screens is 0 so we can evaluate one variable at a time.



(a) TPR on the impact of threshold (b) FPR on the impact of threshold

Fig. 17. The TPR and FPR on the impact of threshold of 50 screens we collected (30 are of the same model).



(a) TPR for each screen. (b) FPR for each screen.

Fig. 18. The TPR and FPR of 50 screens where the first 30 (red) are of the same model. The two groups are ordered by their true PWM frequencies.

we can see that the higher ISO is, the higher the TPR is; therefore, we set ISO to the highest available for the camera.

5) Threshold: SCREENID is authorized only when the difference between estimated frequency and true frequency smaller than the threshold. However, when the threshold become larger, the TPR and the FPR both become higher. Fig. 17 shows the results of TPR and FPR on the impact of threshold. The average TPR is above 94.3% and the FPR is below 0.8% across the same model when the threshold is $0.03Hz$. If the threshold is determined as $0.02Hz$, the TPR is down to 90%. If the threshold is determined as $0.04Hz$, the FPR is up to 1.5%. To consider a better trade-off, we select $0.03Hz$ as the threshold. When the dataset expands, the manufacture of the third-party apps can dynamically adjust the threshold according to the distribution of PWM frequencies of the phone they collected.

6) Processing Time: We measure the processing time of SCREENID on 5 phones (made in 2015 to 2019) and report the average processing time of each componet in Table. I. The total computation time is $0.857s$. The feature extraction scheme takes $0.605s$ which accounts for 70.6% of the total processing time. Currently we implement down-sampling CZT using OpenCV [48] which can be further optimized by using native DSP SDK [49]. We leave it in our future works.

C. Overall Performance

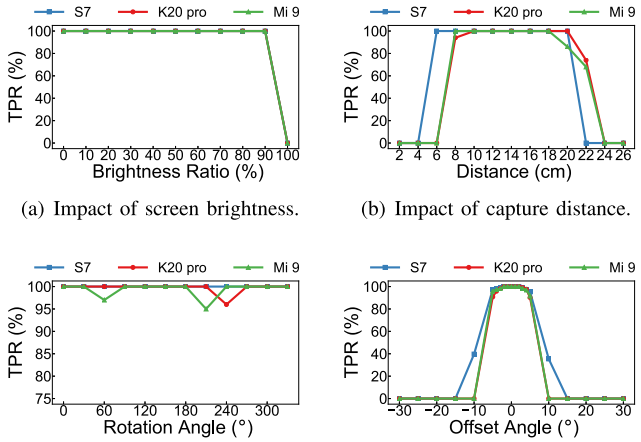
We evaluate the overall performance using 5 cameras and 50 screens. Fig. 18 shows the TPR and FPR of 50 screens. The first 30 screens are of the same model. The average TPR is 99.7% and 95.0% for all screens and phones of the same model. The FPR for each screen in the same model is lower than 1.5%, and for other models, the FPR is 0%, respectively.

D. Robustness of SCREENID

1) Impact of Screen Brightness: Fig. 19(a) shows the TPR under various brightness ratios. We can see the TPR is above 99.5% when the brightness ranges from $10\% - 90\%$. When the brightness is 100%, the screen is always on so it doesn't flicker. Therefore, SCREENID automatically adjusts the brightness to 50% to ensure PWM dimming is applied.

TABLE I
PROCESSING TIME OF SCREENID

Component	Time(s)
Determine QR Code Position	0.033
Contour Extraction	0.054
Multi-frame Concatenation	0.165
Feature Extraction	0.605
Total	0.857



(a) Impact of screen brightness. (b) Impact of capture distance.

(c) Impact of rotation angle. (d) Impact of offset angle.

Fig. 19. The TPR under various impact factors.

2) Impact of Capturing Distances: We evaluate SCREENID under various distances ranging from $0cm$ to $26cm$. Fig. 19(b) shows that TPR is 100% when the distance is $10cm - 18cm$. When the captured distance is smaller than $8cm$, some cameras cannot capture the complete QR code so the TPR drops to 0%. When the distance is longer than $22cm$, the true sampling interval $\Delta_{s\to c}$ becomes larger while the projection ratio M becomes smaller. As the result, the wavelength of stripes is larger than the length of the camera sensor so SCREENID cannot estimate the PWM frequency accurately. This result implies SCREENID works well at the distance from $10-18cm$, which is enough for the mobile payment and can prevent from being candid in a much further distance.

3) Impact of Rotation Angles: We evaluate the impact of rotation angles (i.e., α in Fig. 19(a)) by rotating the screen from $0^\circ - 360^\circ$ clockwise. Fig. 19(c) shows that TPR is above 95.2% in all cases. This implies that SCREENID is robust against how users hold the phone.

4) Impact of Offset Angles: When the camera is not held right above the screen, the captured photo may be distorted. To evaluate the impact of the distortion, we measure the TPR under various offset angles and show the results in Fig. 19. We can see when the offset angle is less than 5°, TPR is above 91.8%. However, when the offset angle is larger than 10°, TPR drops to 0%. As we show in the user study, the restriction does not impair the usability of SCREENID in daily use as the users can easily put the phone with an offset angle within 5°.

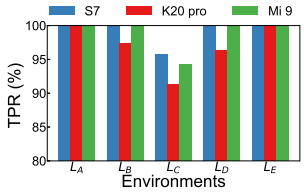


Fig. 20. The TPR on the impact of lighting conditions.

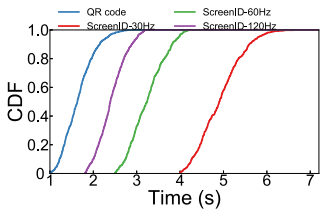


Fig. 21. The required decode time of traditional and SCREENID QR codes.

The result also demonstrates SCREENID can successfully prevent from being sniffed in other viewing angles.

5) Impact of Ambient Light: QR codes are widely used in a variety of environments and the lighting condition is one of the key factor affecting screen to camera communications. We evaluate the impact of lighting condition under following environments: L_A:indoor with light off, L_B, L_C, L_D: outdoor at $9am$,$12am$,$5pm$, L_E:indoor with light on. The results are shown in Fig. 20. It suggests that SCREENID works well under various lighting conditions but the TPR may slightly degrade if the light is too strong (91.3% while using outdoor at $12a.m.$)

E. User Study

We recruited 10 participants to use SCREENID (7 male and 3 female, ages from $22-56$.) Each participant was requested to hold a smartphone to decode 30 traditional QR codes and 30 SCREENID QR codes. Two types of codes were given alternately to avoid the bias.

First, all the traditional and SCREENID QR codes were correctly decoded. We measured the time from a QR code was given to that it was successfully decoded and show the CDF in Fig. 21. We can see that traditional and SCREENID QR codes take $1.6s$ in average. SCREENID QR codes need a longer time because SCREENID needs to capture lots of samples to improve frequency resolution for the target of differentiating more screens. As the camera is getting better, the area of CMOS is getting bigger, thus the camera resolution is getting higher. Except for the camera resolution, the camera's shooting frame rate is also increasing (e.g., 60Hz and 120Hz). Fig. 21 shows the required decode time of traditional and SCREENID QR codes with various capturing rates. We could see that when the capturing rates is $120Hz$, it only needs around $2.4s$ for decoding, which is $0.8s$ longer than traditional QR codes. As the development of slow motion capturing in todays' smartphone, the capturing rate can reach above $200Hz$, which indicates that the decoding time can potentially reduce further.

As for the hand shaking effect in using SCREENID, there is inevitably a certain amount of camera shake (usually less than $2mm$) [50]. All SCREENID QR codes are correctly verified implying SCREENID is robust against the small shaking. It's because we extract a column from each frame independently so the shift between frames do not impair the accuracy.

Finally, we evaluated if participants can put the phone at the right positions. As shown in Sec. 19(b) and 19(d), the best working distances and offset angles is $10-18cm$ and -5° to 5°, respectively. We measured the phone positions while participants presented QR codes and show the CDF of distances and offset angles in Fig. 22. We can see that of 95% offset angles are within 4.5° and 95% distances are from $10-16cm$, which implies SCREENID can successfully work.

IX. SECURITY ANALYSIS

In this section, we discuss various attacks and the ability of SCREENID to deal with them. We assume that the attacker is able to determine the PWM frequency of the victim (e.g., by using a light sensor to remotely measure it), but is not cooperating with the cashier responsible for scanning and verifying the QR code, which means the attacker cannot obatain the rolling shutter interval Δ_C in the cashier either or intercept the processing data in the cashier.

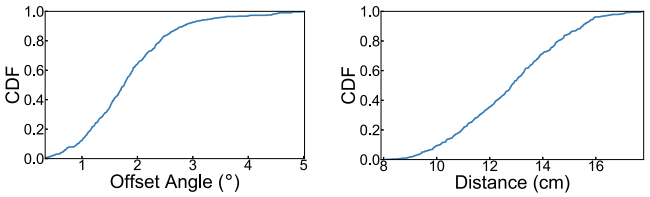
A. Can Attackers Mimic the Victim's PWM Frequency by Changing Their Phone's PWM Frequency?

PWM dimming is applied to an on-chip display controller [32] using one of two methods [51]. The first method has the CPU encode the brightness ratio and send it to the display controller via I^2C interface. The display controller then produces corresponding PWM signals for dimming control. When the victims' smartphone uses this method, there is no way for the attacker to change their PWM frequency without replacing the display controller chip.

The other PWM dimming method involves the CPU outputting a PWM signal through a pin connected to display controller, which internally senses the PWM duty cycle and uses it for dimming control. If the attacker is able to identify the CPU pin being used and has the root permission to modify low-level drivers, then he/she might be able to alter the PWM frequency. However, increasingly strict security restriction on the Android system [52] and the difficulties involved in identifying the correct CPU pin and commands make it nearly impossible to modify the PWM frequency in this way. Note that changing a new screen will not change the CPU pin connected to display controller or modify the low-level drivers either. Therefore, the PWM frequency will not change if the user changes a new screen.

B. Can Attackers Mimic the Victim's PWM Frequency by Adding Rolling Stripes Over the QR Codes?

A possible attacking method is that the attacker may reproduce the PWM pattern on his phone (e.g. embed a video with the rolling stripe patterns on top of screen to fool the camera). However, according to Eq. 6, the stripes on the screen (i.e., the victim's screen) are a function of PWM frequency f_{pwm} as well as the rolling shutter interval Δ_C. The fact that we assume the attacker cannot cooperate with the cashier, which



(a) Offset angle. (b) Distance.

Fig. 22. The CDF of initial offset angle and distance.

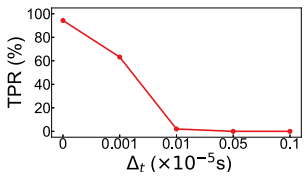


Fig. 23. Impact of estimation error of Δ_C.

means the attacker does not know the configuration of the camera on cashier (i.e., Δ_C) makes it impossible to produce the correct rolling stripes over the QR codes; Another possible attacking way is to use the reflection of QR code on glass of cashier. However, the stripe pattern is produced by the rollign shutter interval Δ_C of its front camera, not the camera of cashier with the PWM latency Δ_S. Hence, this attacking way is meaningless.

C. Can Attackers Mimic the Victim's PWM Frequency by Using an External Lighting Device?

If we assume that the attacker has a lighting device capable of flickering at an arbitrary rate, it might be possible to light their screen at a rate mimicking the PWM frequency of the victim. However, we do not view this kind of attack method as practical, as it would be easily detected by the cashier. Furthermore, this type of lighing device would also produce rolling stripes throughout the captured video, rather than just on the screen, as shown in Fig. 6. Another way is to use a screen connected to a CPU on a development board, and show the QR code on the screen, which can be disguised as a real smartphone to cheat the cashier. If the attackers have the ability to modify the PWM frequency, it can indeed bypass the presented ScreenID system. In order to modify the PWM dimming frequency, the attackers have to identify the CPU pin connected to display controller, further apply the root permission to modify low-level drivers, and finally encode and send the brightness ratio and PWM frequency to the display controller via I^2C interface. It is difficult to identify the correct CPU pin and commands so we assume that the attackers do not have the ability to modify the PWM frequency.

D. Can Attackers Happen to Have a Phone With the Same PWM Frequency as That of the Victim's Phone?

In the proposed system, a QR code is designated legal when the difference between the estimated PWM frequency and characteristic frequency reported in the QR code is less then $0.03Hz$. As shown in Fig. 4, 99.8% of the pairwise difference in frequency is larger than $0.03Hz$. This would require that an attacker use 450 phones to achieve 90% success rate. Such an attack would indeed be possible; however, it would be prohibitively expensive. SCREENID is meant to enhance the security of QR codes with the hardware dimming feature; however, it cannot guarantee security. Note that SCREENID does not need customized techniques so that it can meanwhile incorporate with existing authentication schemes. For example, [23] uses the brightness pixel as fingerprint for differentiating screens. The future research could also focus on the extraction of additional screen characteristics as a complement to SCREENID to formulate fingerprints of greater depth.

E. Can Attackers Enumerate as Small Number of Δ_C Values and Bypass ScreenID System With a Few Attempts?

Δ_C is an important parameter to accurately estimate the PWM frequency. The more accurate Δ_C is, the smaller PWM frequency error is. In order to address the question, we measured the average frequency estimation error of calibrated Δ_C compared to uncalibrated on three different screen models (shown in Fig. 14). It shows that an uncalibrated Δ_C can lead to a $30Hz$ error for PWM frequency estimation, resulting in worse performance when using ScreenID system. Moreover, we also consider it impossible for an adversary to enumerate Δ_C for two reasons. The first reason is that Δ_C varies from camera to camera. For example, the Δ_C for Nexus 5 is $1.3342 \times 10^{-5}s$, for K20pro is $1.1111 \times 10^{-5}s$, and for Mi10ultra is $1.0879 \times 10^{-5}s$. It is hard for the attackers to know an approximately value of Δ_C for all kinds of screen model. The second reason is that a slight difference of Δ_C can lead to huge estimation errors, resulting in worse TPR. Fig. 23 shows the TPR under impact of estimation error of Δ_C. We can see that the estimation error of only $10^{-8}s$ caused a huge drop in TPR (from 94.3% to 63.2%). When the difference is larger than $10^{-7}s$, the TPR drops to 0. Therefore, it is indeed difficult for an adversary to estimate Δ_C with a few attempts.

In summary, SCREENID provides robust protection against attacks using hardware (not easily manipulated) and joint features (PWM frequency f_{pwm}, PWM latency Δ_S, and rolling shutter interval Δ_C), which are difficult to compromise at the same time. At present, SCREENID is limited to screens that use PWM dimming method. There are still a number of devices, such as some Apple iPhone models that still use analog dimming control and would not be protected using our system (however, Apple's higher end models, like iPhone 11 pro, use PWM dimming.) Nonetheless, it is expected that PWM dimming will become the overwhelming standard as OLED gain popularity [53] thanks to their light weight, wide viewing angle, and high color accuracy [32].

X. CONCLUSION

In this study, we proposed SCREENID to enhance the security of QR codes by fingerprinting the screens displaying the QR codes. Only QR codes displaying on its specific screen are accepted. SCREENID exploits PWM frequency as a unique

screen fingerprint. Extensive experiments demonstrate the efficacy of SCREENID in differentiating screens to enhance the security of QR codes in real-world situations.

APPENDIX A
DETAILS OF CHIRP-Z TRANSFORM

In this part, we supplement the details of using Chirp-Z transform (CZT) to refine frequency spectrum to obtain the PWM frequency. CZT is a method for spectrum fining. It only needs to analyze the frequency band where the signal is located. It is hoped that the sampling of the frequency spectrum will be concentrated in this frequency band to obtain a higher resolution, and the part outside the frequency band can be ignored. The working principle is that:

For a known sequence $x(n), 0 \leq n \leq N-1$, its Z-transform is defined as

$$X(z) = \sum_{n=0}^{N-1} x(n) z^{-n}$$

its sampling points can be

$$z_k = AW^{-k}, k = 0, 1, \cdots, M-1.$$

where $A = A_0 e^{j\theta_0}, W = W_0 e^{j\phi_0}$. A_0 is the radius length of the starting sampling point on the spiral line, θ_0 represents the phase angle, ϕ_0 represents the equivalent angle between two adjacent points, W_0 represents the spiral stretch rate. Therefore, we can derive that

$$\begin{aligned} X(z_k) &= \sum_{n=0}^{N-1} x(n) A^{-n} W^{nk}, k = 1, \cdots, M-1 \\ nk &= \frac{1}{2}[n^2 + k^2 - (k-n)^2] \\ X(z_k) &= \sum_{n=0}^{N-1} x(n) A^{-n} W^{\frac{n^2}{2}} W^{-\frac{(k-n)^2}{2}} W^{\frac{k^2}{2}} \\ &= W^{\frac{k^2}{2}} \sum_{n=0}^{N-1} [x(n) A^{-n} W^{\frac{n^2}{2}}] W^{\frac{(n-k)^2}{2}} \end{aligned}$$

Let

$$g(n) = x(n) A^{-n} W^{\frac{n^2}{2}}, h(n) = W^{-\frac{n^2}{2}}$$

Therefore, we can derive that

$$\begin{aligned} X(z_k) &= W^{\frac{k^2}{2}} \sum_{n=0}^{N-1} g(n) h(k-n) \\ &= W^{\frac{k^2}{2}} [g(k) * h(k)], k = 0, \cdots, M-1 \end{aligned}$$

This means that the original formula can be achieved by linear convolution of two sequences. $g(n)$ is a finite sequence, and $h(n)$ can be an infinite sequence. Since we only focus an interval with M points, we can refine the spectrum in these M points. Note that the equivalent angle ϕ_0 can be arbitrary, the frequency resolution can be arbitrary. Therefore, we can refine the spectrum and improve frequency resolution in our concentrated frequency band. For the detailed information, please refer to [45].

TABLE II
DETAILS OF CAMERA MODEL

Model	Manufacture	Resolution
K20pro Premium	Redmi	4000*3000
Nexus 5	LG Google	3280*2464
Xiaomi 8	Xiaomi	4000*3000
Oneplus 7pro	OnePlus	4000*3000
Xiaomi 9	Xiaomi	4000*3000

APPENDIX B
APPROXIMATE SQUARE WAVE WITH SINUSOIDAL WAVES

In this part, we give the proof that it is reasonable to approximate the square wave with sinusoidal waves. Considering the square wave is an even function with a period of T, the amplitude is A_m, the pulse width is αT (i.e., the duty cycle). The function of this square wave in the range of $[-\frac{T}{2}, \frac{T}{2}]$ is

$$f(t) = \begin{cases} A_m, |t| \leq \dfrac{\alpha T}{2} \\ 0, |t| > \dfrac{\alpha T}{2} \end{cases}$$

According to the parity, the Fourier series expansion of the waveform in the range of $[-\frac{T}{2}, \frac{T}{2}]$ is

$$f(t) \sim \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n cos(\frac{2\pi n t}{T})$$

where Fourier coefficient is

$$a_n = \begin{cases} \dfrac{2}{T} \displaystyle\int_{-\frac{T}{2}}^{\frac{T}{2}} f(t) dt, n = 0 \\ \dfrac{2}{T} \displaystyle\int_{-\frac{T}{2}}^{\frac{T}{2}} f(t) \cdot cos(\dfrac{2\pi n t}{T}), n = 1, 2, 3 \cdots \end{cases}$$

Substituting the $f(t)$ function into the Fourier coefficient expression, we can get:

$$\begin{aligned} a_0 = 2\alpha A_m a_n &= \frac{2}{T} \int_{-\frac{\alpha T}{2}}^{\frac{\alpha T}{2}} A_m cos(\frac{2\pi n t}{T}) \\ &= \frac{2A_m}{n\pi} sin(\alpha n \pi), n = 1, 2, 3, \cdots \end{aligned}$$

Therefore, we can derive the Fourier series expansion as follows:

$$f(t) \sim \alpha A_m + \frac{2A_m}{n\pi} \sum_{n=1}^{\infty} \frac{sin(\alpha n \pi)}{n} cos(n\omega t), n = 1, 2, 3, \cdots$$

where the frequency $\omega = \frac{2\pi}{T}$, which represents PWM frequency.

Since we only focus the peak value of frequency spectrum, and the duty cycle α is constant (the bright ratio is constant), we can simplify the expression when $n = 1$, the expression can be:

$$f(t) = \alpha A_m + \frac{2A_m}{\pi} sin(\alpha\pi) cos(\omega t) = \alpha A_m + B cos(\omega t)$$

where $B = \frac{2A_m}{\pi} sin(\alpha\pi)$ is a constant value.

So far, we proof the square wave can be simplified to a sine wave.

TABLE III
DETAILS OF SCREEN MODEL

Model	Manu-facture	Num.	Type	Size (inch)	Resolution	Refre-sh rate
iphoneX	Apple	1	OLED	5.8	2436*1125	60hz
Mate30pro	Huawei	4	OLED	6.53	2400*1176	90hz
Mate30(5g)	Huawei	3	OLED	6.62	2340*1080	60hz
Mi10	Xiaomi	1	OLED	6.67	2340*1080	90hz
Mi10pro	Xiaomi	3	OLED	6.67	2340*1080	90hz
Mi10 Lite	Xiaomi	1	OLED	6.57	2400*1080	60hz
Nova7	Huawei	1	OLED	6.53	2400*1080	60hz
Nova7pro	Huawei	1	OLED	6.57	2340*1080	60hz
Oneplus7pro	Oneplus	1	OLED	6.67	3120*1440	90hz
OppoR11	Oppo	1	OLED	5.5	1920*1080	60hz
P40	Huawei	6	OLED	6.58	2640*1200	90hz
P40pro	Huawei	6	OLED	6.58	2640*1200	90hz
P40pro+	Huawei	6	OLED	6.58	2640*1200	90hz
Redmi10x	Redmi	1	OLED	6.53	2340*1080	60hz
Redmi10x pro	Redmi	1	OLED	6.53	2340*1080	60hz
K30pro	Redmi	1	OLED	6.67	2400*1080	60hz
Samsung S7	Samsung	2	OLED	5.1	2560*1440	60hz
Redmi k30	Redmi	2	LCD	6.67	2400*1080	60hz
Xperia XZ1	Sony	2	LCD	5.8	1920*1080	60hz
Mi6	Xiaomi	4	LCD	5.15	1920*1080	60hz
Mix 2S	Xiaomi	2	LCD	5.99	2160*1080	60hz

APPENDIX C
EXPERIMENTAL DEVICES

In this part, we describe the information of experimental devices. We select 5 smartphones as the camera, the parameters of the camera model are listed in Tab. II. As for the screen model, we generated QR codes on 50 screens. Among 50 screens, 10 are LCD and 40 are OLED screens. 30 screens are of the same model. The parameters of the screen model are listed as Tab. III.

REFERENCES

[1] *Alipay: Experience Fast, Easy and Safe Online Payments*, Alipay, Pudong, China, 2020.

[2] *WeChat Pay: Beyond Payment, Leading Mobile Payment Platform*, WeChatPay, Shenzhen, China, 2020.

[3] X. Bai *et al.*, "Picking up my tab: Understanding and mitigating synchronized token lifting and spending in mobile payment," in *Proc. 26th USENIX Secur. Symp.*, 2017, pp. 593–608.

[4] JobTubeDaily, "Your mobile money could be stolen by this attack!" JobTube, 2018. [Online]. Available: https://freewechat.com/a/MzIzNzcyNzkzNQ==/2247491423/

[5] Y.-W. Chow, W. Susilo, G. Yang, M. H. Au, and C. Wang, "Authentication and transaction verification using QR codes with a mobile device," in *Proc. Int. Conf. Secur., Privacy Anonymity Comput., Commun. Storage*. Springer, 2016, pp. 437–451.

[6] J. Lu, Z. Yang, L. Li, W. Yuan, L. Li, and C.-C. Chang, "Multiple schemes for mobile payment authentication using QR code and visual cryptography," *Mobile Inf. Syst.*, vol. 2017, 2017.

[7] C. Chen, "QR code authentication with embedded message authentication code," *Mobile Netw. Appl.*, vol. 22, no. 3, pp. 383–394, Jun. 2017.

[8] C.-N. Yang, J.-K. Liao, F.-H. Wu, and Y. Yamaguchi, "Developing visual cryptography for authentication on smartphones," in *Proc. Int. Conf. Ind. IoT Technol. Appl.* Springer, 2016, pp. 189–200.

[9] X. Cao, L. Feng, P. Cao, and J. Hu, "Secure QR code scheme based on visual cryptography," in *Proc. 2nd Int. Conf. Artif. Intell. Ind. Eng. (AIIE)*. Amsterdam, The Netherlands: Atlantis Press, 2016, pp. 433–436.

[10] W.-P. Fang, "Offline QR code authorization based on visual cryptography," in *Proc. 7th Int. Conf. Intell. Inf. Hiding Multimedia Signal Process.*, Oct. 2011, pp. 89–92.

[11] S. Thamer and B. Ameen, "A new method for ciphering a message using QR code," *Comput. Sci. Eng.*, vol. 6, no. 2, pp. 19–24, 2016.

[12] H. Pan, Y.-C. Chen, L. Yang, G. Xue, C.-W. You, and X. Ji, "MQRCode: Secure QR code using nonlinearity of spatial frequency in light," in *Proc. 25th Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2019, pp. 1–18.

[13] J. M. McCune, A. Perrig, and M. K. Reiter, "Seeing-is-believing: Using camera phones for human-verifiable authentication," in *Proc. IEEE Symp. Secur. Privacy*, May 2005, pp. 110–124.

[14] A. Davis, M. Rubinstein, N. Wadhwa, G. Mysore, F. Durand, and W. T. Freeman, "The visual microphone: Passive recovery of sound from video," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 79:1–79:10, 2014.

[15] C. Danakis, M. Afgani, G. Povey, I. Underwood, and H. Haas, "Using a CMOS camera sensor for visible light communication," in *Proc. IEEE Globecom Workshops*, Dec. 2012, pp. 1244–1248.

[16] C. Zhang and X. Zhang, "LiTell: Robust indoor localization using unmodified light fixtures," in *Proc. 22nd Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2016, pp. 230–242.

[17] S. Zhu and X. Zhang, "Enabling high-precision visible light localization in today's buildings," in *Proc. 15th Annu. Int. Conf. Mobile Syst., Appl., Services*, Jun. 2017, pp. 96–108.

[18] C. Zhang and X. Zhang, "Pulsar: Towards ubiquitous visible light localization," in *Proc. 23rd Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2017, pp. 208–221.

[19] S. Zhu, C. Zhang, and X. Zhang, "Automating visual privacy protection using a smart LED," in *Proc. 23rd Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2017, pp. 329–342.

[20] K. Zhang *et al.*, "ChromaCode: A fully imperceptible screen-camera communication system," *IEEE Trans. Mobile Comput.*, vol. 20, no. 3, pp. 861–876, Mar. 2021.

[21] M. Zhou, Q. Wang, T. Lei, Z. Wang, and K. Ren, "Enabling online robust barcode-based visible light communication with realtime feedback," *IEEE Trans. Wireless Commun.*, vol. 17, no. 12, pp. 8063–8076, Dec. 2018.

[22] K.-C. Liao and W.-H. Lee, "A novel user authentication scheme based on QR-code," *J. Netw.*, vol. 5, no. 8, p. 937, Aug. 2010.

[23] Z. Zhou, D. Tang, W. Wang, X. Wang, Z. Li, and K. Zhang, "Beware of your screen: Anonymous fingerprinting of device screens for off-line payment protection," in *Proc. 34th Annu. Comput. Secur. Appl. Conf.*, Dec. 2018, pp. 77–88.

[24] P. Punithavathi and S. Geetha, "Visual cryptography: A brief survey," *Inf. Secur. J., Global Perspective*, vol. 26, no. 6, pp. 305–317, Nov. 2017.

[25] L. K. Saini and V. Shrivastava, "A survey of digital watermarking techniques and its applications," 2014, *arXiv:1407.4735*.

[26] M. M. Yeung and F. Mintzer, "An invisible watermarking technique for image verification," in *Proc. Int. Conf. Image Process.*, vol. 2, Oct. 1997, pp. 680–683.

[27] N. Nikolaidis and I. Pitas, "Robust image watermarking in the spatial domain," *Signal Process.*, vol. 66, no. 3, pp. 385–403, 1998.

[28] J. Abraham and V. Paul, "An imperceptible spatial domain color image watermarking scheme," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 31, no. 1, pp. 125–133, Jan. 2019.

[29] R. Riad, F. Ros, R. Harba, H. Douzi, and M. El Hajji, "Pre-processing the cover image before embedding improves the watermark detection rate," in *Proc. 2nd World Conf. Complex Syst. (WCCS)*, Nov. 2014, pp. 705–709.

[30] K. Gourrame *et al.*, "Robust print-cam image watermarking in Fourier domain," in *Proc. Int. Conf. Image Signal Process.* Springer, 2016, pp. 356–365.

[31] K. Thongkor and T. Amornraksa, "Robust image watermarking for camera-captured image using image registration technique," in *Proc. 14th Int. Symp. Commun. Inf. Technol. (ISCIT)*, Sep. 2014, pp. 479–483.

[32] M. Barr, "Pulse width modulation," *Embedded Syst. Program.*, vol. 14, no. 10, pp. 103–104, 2001.

[33] H. Sugiyama, S. Haruyama, and M. Nakagawa, "Brightness control methods for illumination and visible-light communication systems," in *Proc. 3rd Int. Conf. Wireless Mobile Commun. (ICWMC)*, Mar. 2007, p. 78.

[34] *PWM Ranking—The Best Displays for the Eyes*, NotebookCheck, Vienna, Austria, 2019.

[35] C.-K. Liang, L.-W. Chang, and H. H. Chen, "Analysis and compensation of rolling shutter effect," *IEEE Trans. Image Process.*, vol. 17, no. 8, pp. 1323–1330, Aug. 2008.

[36] *ADPD2212 Datasheet*, Analog Devices, Wilmington, MA, USA, 2016.

[37] *Device Specifications*, 2019. [Online]. Available: https://www.devicespecifications.com/

[38] B. O'Donnell, "Steals money sneakily by scanning people's QR code," Int. Cultural Commun., Shanghai, China, 2019. [Online]. Available: https://www.thatsmags.com/shanghai/post/27482/man-steals-money-sneakily-scanning-people-s-qr-codes-in-shanghai

[39] T. Li, "QR code scams rise in China, putting e-payment security in spotlight," JianShu, 2019. [Online]. Available: https://www.jianshu.com/p/b9657161933a

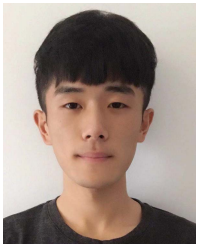
[40] J. Genoe *et al.*, "30.2 digital PWM-driven AMOLED display on flex reducing static power consumption," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2014, pp. 488–489.

[41] *Sensor Rolling Shutter Skew of Camera2*, GoogleDevelopers, Mountain View, CA, USA, 2020.

[42] N. Kanopoulos, N. Vasanthavada, and R. L. Baker, "Design of an image edge detection filter using the Sobel operator," *IEEE J. Solid-State Circuits*, vol. JSSC-23, no. 2, pp. 358–367, Apr. 1988.
[43] A. V. Oppenheim, *Discrete-Time Signal Processing*. London, U.K.: Pearson, 1999.
[44] A. Testa, D. Gallo, and R. Langella, "On the processing of harmonics and interharmonics: Using Hanning window in standard framework," *IEEE Trans. Power Del.*, vol. 19, no. 1, pp. 28–34, Jan. 2004.
[45] L. Rabiner, R. Schafer, and C. Rader, "The chirp z-transform algorithm," *IEEE Trans. Audio Electroacoustics*, vol. AE-17, no. 2, pp. 86–92, Jun. 1969.
[46] J. Steeman, "QR code data capacity," 2019. [Online]. Available: https://blog.qr4.nl/page/QR-Code-Data-Capacity.aspx
[47] S. Hilbert, "FFT zero padding," BitWeenie, 2013. Accessed: Nov. 30, 2020. [Online]. Available: https://www.bitweenie.com/listings/fft-zero-padding
[48] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision With the OpenCV Library*. Sebastopol, CA, USA: O'Reilly Media, Inc., 2008.
[49] *Hexagon DSP SDK*, Qualcomm, San Diego, CA, USA, 2019.
[50] K. J. Connolly, *The Psychobiology of the Hand*. Cambridge, U.K.: Cambridge Univ. Press, 1998.
[51] *LM36923 Highly Efficient Triple-String White LED Driver*, Texas Instruments, Dallas, TX, USA, 2015.
[52] T. Mohini, S. A. Kumar, and G. Nitesh, "Review on Android and smartphone security," *Res. J. Comput. Inf. Technol. Sci.*, vol. 2320, p. 6527, Jan. 2013.
[53] *Shipments of Flexible and Rigid Amoled Panels Worldwide From 2015 to 2022*, Statista, Hamburg, Germany, 2018.

Guangtao Xue (Member, IEEE) received the Ph.D. degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, in 2004. He is currently a Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China. His research interests include vehicular *ad-hoc* networks, wireless networks, mobile computing, and distributed computing. He is a member of the IEEE Communication Society.

Yijie Li (Student Member, IEEE) received the bachelor's degree from the Hongyi Class of the Department of Computer Science, Wuhan University, China, in 2018. He is currently pursing the Ph.D. degree with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China. His current research interests include mobile computing.



Hao Pan (Student Member, IEEE) received the bachelor's degree from the Yingcai Honors College, University of Electronic Science and Technology of China, in 2016. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Shanghai Jiao Tong University. His current research interests include mobile computing, with special focuses on wireless communication and sensing, security for mobile systems, and mobile human–computer interaction.

Lanqing Yang received the bachelor's degree from the Department of Software Engineering, University of Electronic Science and Technology of China, in 2017. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Shanghai Jiao Tong University. His current research interests reside in mobile computing and intelligent sensing.

Yi-Chao Chen (Member, IEEE) received the B.S. and M.S. degrees from the Department of Computer Science and Information Engineering, National Taiwan University, in 2004 and 2006, respectively, and the Ph.D. degree in computer science from the University of Texas at Austin in 2015. He joined Shanghai Jiao Tong University as a Tenure-Track Assistant Professor at the Department of Computer Science and Engineering in 2018. Prior to joining SJTU, he spent a year as a Researcher at Huawei Future Network Theory Laboratory, Hong Kong, and then worked as a Co-Founder at Hauoli LLC. His research interests focus on networked systems and span the areas of wireless networking, network measurement and analytics, and mobile computing.

Xiaoyu Ji (Member, IEEE) received the B.S. degree in electronic information and technology and instrumentation science from Zhejiang University, Hangzhou, China, in 2010, and the Ph.D. degree from the Department of Computer Science, The Hong Kong University of Science and Technology, in 2015. From 2015 to 2016, he was a Researcher at Huawei Future Networking Theory Laboratory, Hong Kong. He is now an Associate Professor with the Department of Electrical Engineering, Zhejiang University. His research interests include the IoT security, wireless communication protocol design, especially with cross-layer techniques. He won the Best Paper Awards of ACM CCS 2017, ACM ASIACCS 2018, and IEEE TrustCom 2014.

Jiadi Yu (Senior Member, IEEE) is currently an Associate Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. His research interests include cyber security and privacy, mobile and pervasive computing, cloud computing, and wireless sensor networks.