Through-Screen Computing

Ye, Hanting

DOI
10.4233/uuid:d904b49f-51bb-4635-9323-44faa50fd72e

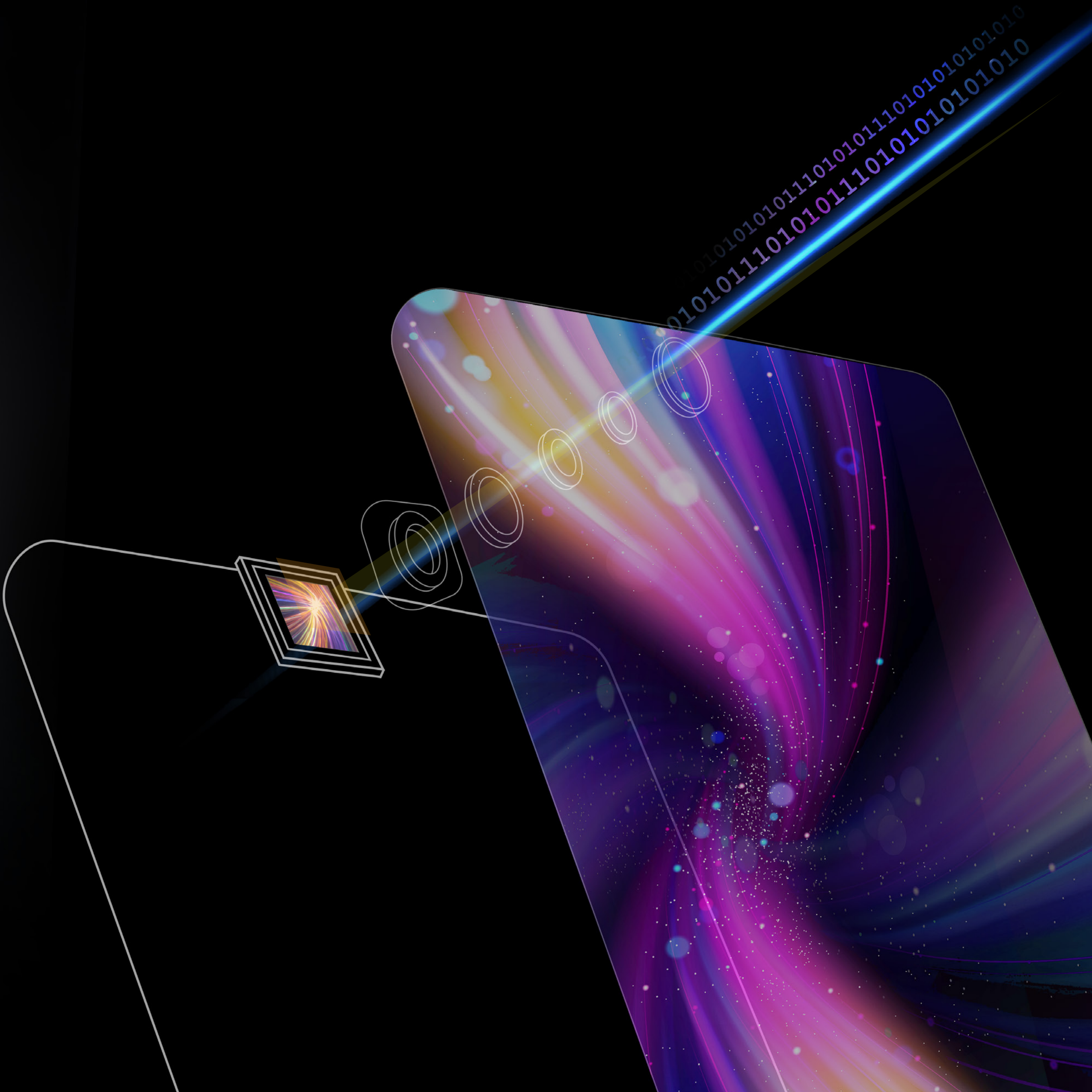
Publication date
2025

Document Version
Final published version

Citation (APA)
Ye, H. (2025). *Through-Screen Computing*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:d904b49f-51bb-4635-9323-44faa50fd72e

Through-Screen Computing

Hanting Ye 叶汉霆



Through-Screen Computing

THROUGH-SCREEN COMPUTING

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology,
by the authority of the Rector Magnificus Prof. dr. ir. T.H.J.J. van der Hagen,
chair of the board for Doctorates,
to be defended publicly on
Wednesday 22 January 2025 at 10:00 o'clock

by

Hanting YE

Master of Engineering in Electronics and Communication Engineering,
University of Electronic Science and Technology of China, China,
born in Yuexi, Sichuan, China.

The work presented in this dissertation was carried out at the Embedded Systems (ES) Group of Delft University of Technology, The Netherlands, and was partially funded by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska Curie grant agreement ENLIGHT'EM No. 814215.

Keywords: Ubiquitous Computing, Transparent Screen, Under-Screen Sensors, Visible Light Communication, Privacy and Security

Printed by: Ipskamp Printing

Front & Back: Our Friday Goldfish Design Studio

ISBN 978-94-6366-981-8

An electronic version of this dissertation is available at
`http://repository.tudelft.nl/`.

To Hanting in his lowest moments,
and to those lights passed through the darkness to reach him.

CONTENTS

SUMMARY

The advancement in transparent screen technology has promoted adoption of full-screen design on mobile devices, reducing the area occupied by optical sensors to maximize the devices' screen-to-body ratio. In modern smartphones, front-facing optical sensors, such as ambient light sensor and camera, now must be placed under the transparent screen to capture ambient light and visual information. Motivated by this trend, we propose **Through-Screen Computing** in this dissertation. It is a new concept that refers to *the processing of light signals for various computing purposes such as communication, sensing, and imaging, where the light comes from the physical world and passes through a special medium – the transparent screen – before reaching the under-screen optical sensors.* This concept opens up new challenges and opportunities in connectivity, privacy, and security of future devices equipped with transparent screens. In this dissertation, we outline a vision for through-screen computing and address the challenges of transparent screens acting as both passive blockers and active interferers of input light signals.

This dissertation focuses on two subsystems in the context of through-screen computing: *Through-Screen Visible Light Communication (VLC)* and *Screen Perturbation for Visual Privacy Protection.* In the context of VLC, the full-screen trend challenges the deployment of this technology. We propose *Through-Screen VLC* with under-screen optical sensors as receivers. To address the attenuation of the light by the transparent screen, we develop SpiderWeb, a system exploiting the color domain to mitigate the color-pulling effect introduced by the transparent screen. We also leverage the Under-Screen Camera (USC) for VLC and design novel demodulation algorithms to reduce multi-pixel screen interference and improve performance. Experimental results show significant improvements in both data rate and transmission range, using a prototype we build with two commercial smartphones. For privacy protection, we propose *Screen Perturbations*, modifying pixels displayed on the transparent screen to embed speckled color blocks and color shifts in the final image captured by the USC. Screen perturbations can be exploited to disrupt advanced deep neural networks used on image classification and face recognition tasks. We first design two image-specific methods to add screen perturbations to the images captured by USC. Next, we develop Unicorn, a universal screen perturbation method optimized for robustness in various conditions. All these designed perturbations work successfully against various deep neural network-based image classification services with high success rates.

Through these two subsystems, as well as the proposed theoretical and experimental approaches and results, we demonstrate the transformative potentials of through-screen computing, setting the stage for future research and development on various computing purposes in the era of transparent screen and full-screen devices.

1 INTRODUCTION

Mobile devices, such as smartphones, tablets, laptops, smartwatches, e-readers, and handheld gaming consoles, have become ubiquitous worldwide. Now it is hard to imagine a world without these mobile devices. By 2022, there were more smartphones than people in the world, and the number of mobile devices continues to grow at nearly five times the rate of the global human population [1]. The academic and industrial communities envision an exciting ubiquitous computing future where mobile devices play an increasingly significant role in daily life. The rapid evolution in mobile devices leads to the fact that a flagship device from just a few years ago now seems outdated. Take the smartphone screens for an example. The screen has become the indispensable interactive interface between users and their smartphones since the launch of the first iPhone in 2007, which revolutionized the industry by eliminating most of the physical buttons. In the past decade, various innovative smartphone screen designs have been further realized to minimize the bezels and the notch area taken up by front cameras and other optical sensors to increase the screen-to-body ratio. These designs include the notch screen, teardrop notch screen, and through-hole screen of Android phones, as illustrated in Figure 1.1, and the "dynamic island" of iPhones. The ultimate goal is to achieve a borderless *"full-screen device"*, unifying user interaction functions and aesthetic design with the potential wide adoption of *transparent screens*.

1.1. THE RISE OF TRANSPARENT SCREEN

Transparent screen technology utilizes transparent electrode materials to maintain display functionality while being visually transparent. This technology has enabled a wide range of innovative applications. For instance, it has been used in large transparent TVs (Figure 1.2) and car windshields (Figure 1.3) that function as display screens. Moreover, transparent screens are also being explored for sustainable smart windows, which can serve dual purposes by providing illumination and displaying information. In more advanced use cases, they are key to enhancing Augmented Reality (AR) and Mixed Reality

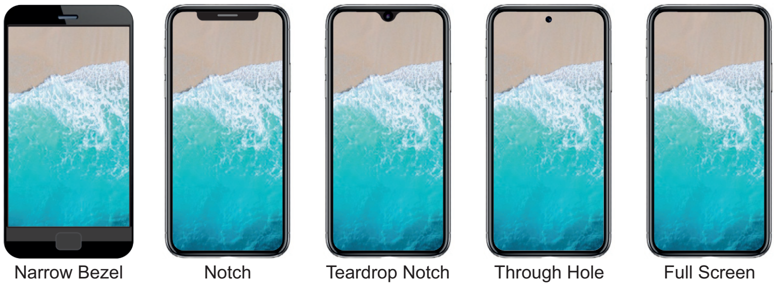


Figure 1.1: Illustration of the screen evolution of smartphones: efforts made to eliminate the notch and bezel.

Figure 1.2: LG's transparent OLED TV [4].

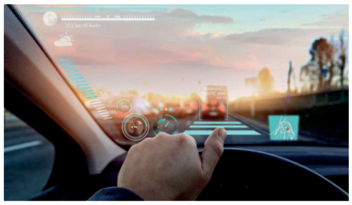


Figure 1.3: Head-up display via transparent screen [5].

(MR) applications. In addition to these applications, transparent screens have revolutionized mobile devices, leading to the development of full-screen devices such as laptops (e.g., Thunderobot T-BOOK and Samsung Blade Bezel) and smartphones (e.g., ZTE AXON 20/30/40, Xiaomi MIX 4, and Samsung Galaxy Z Fold) [2]. These full-screen devices, with their larger screen-to-body ratios, provide a better user experience by offering a more immersive and intelligent interface [3].

To achieve this goal, the screen of full-screen devices comprises a *Transparent Screen Region* and a Normal Screen Region, as illustrated in Figure 1.4. The transparent screen region is built with transparent electrode materials, serving two purposes: 1) displaying various contents, similar to the normal screen, and 2) allowing light to pass through the screen to reach under-screen optical sensors. The transparent screen's pixel layout is therefore optimized to balance the display functionality and the light transmittance to meet these two purposes [6], as shown in Figure 1.4(c). This innovative design allows placing optical sensors under the transparent screen without sacrificing their functionality, leading to the so-called *Under-Screen Sensors* [7], such as the Under-Screen Ambient Light Sensors [8] and Under-Screen Cameras (USC)[1].

1.2. THROUGH-SCREEN COMPUTING

Transparent screens are revolutionizing our visual experience of mobile devices. However, it also changes the traditional ubiquitous computing of light-based signals since optical sensors now must be placed *under* the transparent screen instead of traditionally *on* the screen. Motivated by this paradigm shift, in this dissertation, we propose the

[1]It is also referred as Under-Display Camera (UDC) or Under-Panel Camera (UPC) in the literature [9]–[12].

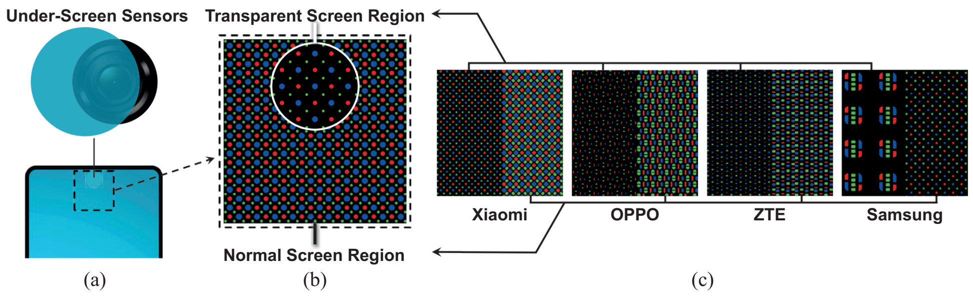


Figure 1.4: Illustration of (a) a full-screen smartphone with under-screen sensors; (b) magnified micrograph of the transparent screen region and the normal screen region; (c) screen diversity: the comparison of transparent screen regions and normal screen regions designed by different smartphone manufacturers.

concept of **Through-Screen Computing**, which we define as, *through-screen computing refers to the processing of light signals for various computing purposes such as communication, sensing, and imaging, where the light comes from the physical world and passes through a special medium – the transparent screen – before reaching the under-screen sensors.* Below, we present its essential components.

- **Light Source:** Through-screen computing uses light signals as the computing input. We mainly consider two types of light sources: (1) *Passive sources:* referring to the objects that reflect light when illuminated, containing spatial information about the object and its environment. (2) *Active sources:* referring to the LED luminaires present in the lighting infrastructure, which are not only beneficial for illumination but also can be modulated in light intensity or colors at a high frequency to transmit information.

- **Transparent Screen:** In through-screen computing, the transparent screen significantly affects light propagation. It influences the visual imaging of light reflected by passive objects (e.g., cats, see Figure 1.5(left)) or the intensity and color of light emitted by active sources (e.g., LEDs, see Figure 1.5(right)). Additionally, the transparent screen can act as an active source, featuring an array of RGB pixels in various layouts, shapes, and sizes. This array displays dynamic contents on mobile devices and brings challenges to through-screen computing, such as attenuation, color shift, and interference with light from other passive and active sources.

- **Under-Screen Sensors:** We consider two types of under-screen sensors in this dissertation: (1) *Single-pixel under-screen sensors*, such as an under-screen photodiode and an under-screen ambient light sensor. Both are semiconductor devices that convert the detected light into electrical information. While an under-screen photodiode only measures light intensity, an under-screen ambient light sensor, which combines a photodiode and a color filter, can measure both the intensity and the color of through-screen light signals; and (2) *Multi-pixel under-screen sensors*, such as an under-screen camera, which can also detect through-screen light signals but in an intuitive and understandable multi-pixel visual image output.

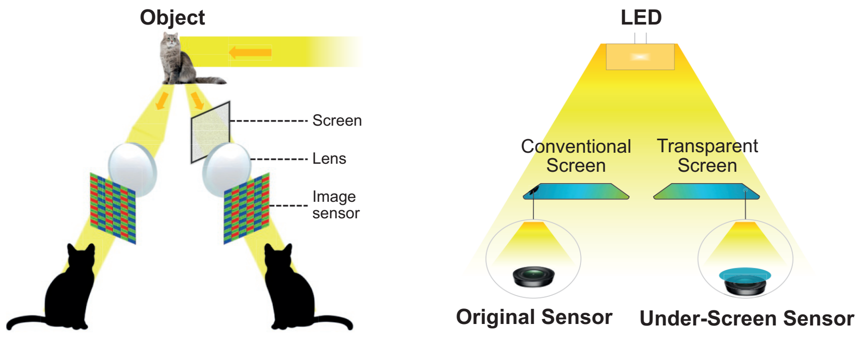


Figure 1.5: The concept of through-screen computing: (left) passive source; (right) active source.

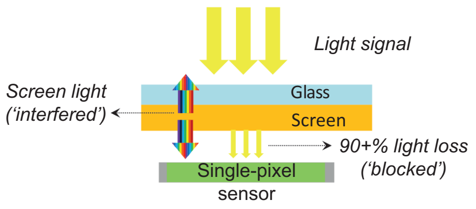
Through-screen computing encompasses various interdisciplinary technologies that involve different domains: materials, structural, electrical, embedded systems, and software engineering. We do not intend to cover all these domains in this dissertation. Given that full-screen design represents the next frontier for innovation in mobile devices, we focus on the computing behind the transparent screen of full-screen devices, addressing several critical challenges that can advance through-screen computing. Although the considered screen is "transparent", it still introduces new challenges to under-screen sensors. Under-screen sensors now need to overcome the screen's interference to detect and recover input light signals. For instance, the imaging of passive objects becomes blurrier when viewed through the under-screen camera compared to the original camera, and the detection of input light emitted by active sources may not be as accurate with under-screen sensors as with the original 'on-screen' optical sensors.

1.3. CHALLENGES

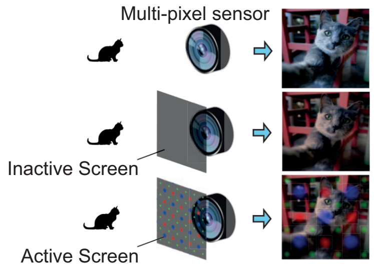
The transparent screen is not merely a passive medium like glass; it is also a dynamic, active light source that changes unpredictably based on the displayed content. Thus, adopting conventional mobile system designs for through-screen computing is impractical. Designing, implementing, and deploying through-screen computing brings new challenges. Next, we discuss several primary challenges of realizing through-screen computing on full-screen devices, as shown in Figure 1.6. We focus on the transparent screen's different states (OFF and ON), design (pixel layout), and the characteristics of different types of under-screen sensors, including single-pixel and multi-pixel sensors.

Challenge 1: Screen as a Passive 'Blocker'

Although the screen may appear transparent to naked eyes, it is actually not completely transparent. The light transmittance of the transparent screen used on current mobile devices is merely 2.9% [10], [13]. This strong attenuation of light significantly reduces the Signal-to-Noise Ratio (SNR) of light 'signals' at the under-screen optical sensors, severely degrading the system performance. Furthermore, the basic principle of camera imaging is the "pinhole camera", which has a small aperture to allow a narrow beam of light rays to pass through and reach the camera. The optimal imaging dis-



(a) From the perspective of *communication signals*.



(b) From the perspective of *vision*.

Figure 1.6: Major challenges in realizing through-screen computing.

tance of a pinhole camera is its focal length. However, in through-screen computing, the aperture of the under-screen camera can be regarded as a combination of the finite screen openings in the transparent screen and the under-screen camera aperture. This combined small aperture changes the focal length and the amount of light reaching the under-screen camera, and it is comparable in size to the wavelength of visible light, resulting in light scattering, diffraction, and absorption [10]. These optical impacts caused by the screen lead to lower SNR, blurred visuals, and color shifts in the image captured by under-screen cameras [10], [14], [15].

Challenge 2: Screen as an Active 'Interferer'

The activated transparent screen region of a full-screen device significantly impacts the captured signal in the intensity and color domains. In the intensity domain, the detected screen light often far exceeds the input light signal, as shown in Figure 1.6(a), since the under-screen sensor is much closer to the screen than to the light source. This proximity poses a challenge in removing dynamic screen light interference from the detected light signals. In the color domain, single-pixel photodiodes as sensors can have differential sensitivity to light of varying frequencies, and thus the interference induced by different screen colors is not uniform [16]. The strong and inconsistent interference from the screen (depending on the displayed content) is superimposed on the attenuated input signal, resulting in a low Signal-to-Interference-plus-Noise Ratio (SINR) at the under-screen sensors. Furthermore, the pinhole camera alone is not sufficient to fully capture a clear image. Modern cameras use convex (converging) lenses at small apertures to collect light, refracting light rays from a reflecting object or an emitting active source to form an image. The presence of lenses allows the object being photographed to maintain its original shape while enlarging its size on the camera's sensor. However, this enlargement causes the object in the image to lose sharp boundaries, resulting in a diffuse rendering effect with blurred edges. Hence, when the screen is lit up, different tiny R/G/B screen pixels lead to *optical perturbations: magnified and inverted speckled color blocks and color shifts of the R/G/B screen pixels on the captured image.*

Challenge 3: Screen Diversity

Variations in the design of transparent screens across different full-screen devices affect the performance of through-screen computing in multiple ways. Firstly, differences in the intrinsic material and manufacturing properties of device screens lead to varying light transmittance. Secondly, different manufacturers customize screen parameters, such as maximum screen brightness and screen color gamut, introducing different interference detected by the under-screen sensor. Furthermore, both academia and industry have invested effort in designing diverse screen pixel shapes and layouts to optimize the screen resolution and enhance the screen's light transmittance [6], [14]. As a result, manufacturers have developed distinctive pixel configurations within their screen designs (cf. Figure 1.4(c)). As mentioned, camera optics [17] indicate that even tiny illuminated screen pixels can result in optical perturbations on the captured image. These specialized designs introduce unique optical perturbations on the images captured by the under-screen camera. So far, manufacturers can only deactivate the screen display to eliminate this interference when using the under-screen cameras for imaging. However, achieving reliable through-screen computing at all times, regardless of whether the screen is activated or deactivated, remains a significant challenge.

1.4. PROBLEM STATEMENT

In the previous section, we have described the high-level challenges in through-screen computing. The goal of this thesis is to tackle these challenges brought by the transparent screens and under-screen sensors to enhance the light-based connectivity, privacy, and security performance of future full-screen devices. Hence, this thesis focuses on the following research question:

How to build innovative through-screen systems to improve light-based connectivity, privacy, and security performance in the context of through-screen computing?

While this thesis does not solve all the aforementioned issues, we cherry-pick a few subsystems of through-screen computing to demonstrate the necessity of addressing the issues and portraying ways to achieve this. We first tackle the *connectivity* issue rising along with deploying the new wireless communication paradigm, visible light communication, on Commercial Off-The-Shelf (COTS) mobile systems equipped with transparent screens. Further, the programmable illumination of screen pixels can perturb the optical path between the under-screen camera and the object without altering the object itself. We leverage this capability to protect sensitive visual information captured by the under-screen camera (improving the *privacy & security* performance). Through a series of comprehensive studies, this thesis explores this intricate landscape, shedding light on the challenges and opportunities of *Through-Screen Visible Light Communication* and the broader implications of *Screen Perturbation for Visual Privacy Protection*.

1.4.1. THROUGH-SCREEN VISIBLE LIGHT COMMUNICATION

Visible Light Communication (VLC) is considered a key enabler for future wireless networks such as 6G [18], [19]. VLC offers several advantages over traditional Radio Frequency (RF)-based wireless communications. The visible light frequency band is about 10,000 times larger than the RF band, and does not penetrate walls, allowing VLC to achieve higher spatial multiplexing and provide massive and secure connections to a

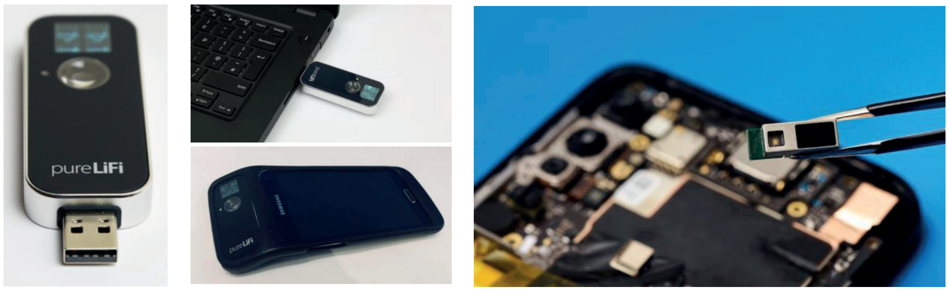
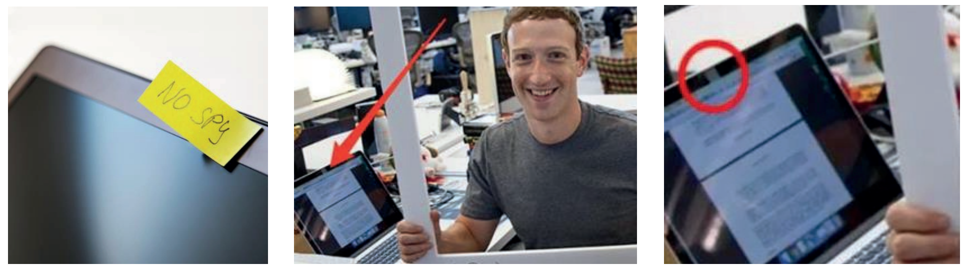

(a) LiFi-XC [31] (b) LiFi-XC integrated in laptops and smartphones [20] (c) Light Antenna ONE [21]

Figure 1.7: VLC receiver modules from pureLiFi.

large number of future mobile devices to solve connectivity issues. Several leading VLC companies, such as pureLiFi and Signify, have developed VLC products for mobile devices, including smartphones and laptops. pureLiFi has launched the world's first certified USB VLC receiver, LiFi-XC, as shown in Figure 1.7(a); two examples that integrate the VLC receiver into mobile devices are shown in Figure 1.7(b): smartphones and laptops at Mobile World Congress (MWC) 2018 [20]. Five years later, pureLiFi launched the IEEE 802.11bb-compliant Light Antenna ONE at MWC 2023, as shown in Figure 1.7(c), a VLC module that is ready for integration into mobile devices [21].

Although significant progress has been made in the industry for the development, deployment, and standardization of VLC systems, a key direction has been largely neglected by the VLC community: *how to practically deploy VLC on current/future mobile devices, like smartphones that are heavily used every day by billions of people?* Here, *where* to place VLC receivers on smartphones is of paramount importance. Since VLC transmitters are typically placed on the ceiling (used for both illumination and communication) and VLC signal quality is dominated by line-of-sight links, placing optical receivers on the front of smartphones is both beneficial and essential. Therefore, ambient light sensors, front cameras, and other front-facing optical sensors widely available on commodity smartphones have significant potential as VLC receivers since these sensors are predominantly oriented toward the LEDs in the surrounding environment during conventional smartphone usage. Some studies [22]–[24] have explored the use of ambient light sensors in mobile devices as VLC receivers. Other studies [25]–[30] have proposed utilizing the front camera on COTS devices as the VLC receiver, and enhancing the sampling rate of the VLC receiver through the rolling-shutter effect of the camera.

However, full-screen devices introduce new challenges because there is *no space on the smartphone screen* to place VLC receivers. Consequently, whether it is a dedicated VLC receiver like Light Antenna ONE, which needs to be placed on the notch screen area of smartphones, or re-purposed COTS VLC receivers like traditional optical sensors (ambient light sensors and front cameras) typically placed on the screen, these are being relocated under the screen, giving rise to *under-screen sensors*. The advent of under-screen sensors introduces new challenges for the deployment of VLC receivers on COTS devices. In light of these developments, a novel communication paradigm – termed *"Through-Screen Visible Light Communication"* – is proposed and will be investigated in this thesis.



(a) Put sticker over the laptop camera [39] (b) Meta CEO Mark Zuckerberg covers the device cameras with tape [40] (c) Zoom in tape cover [40]

Figure 1.8: A simple pre-capture protection method is to directly cover the cameras on mobile devices with tape or stickers, which is effective but inconvenient.

1.4.2. SCREEN PERTURBATION FOR VISUAL PRIVACY PROTECTION

Cameras are prevalent in mobile devices, raising concerns about the potential misuse of sensitive visual information, such as faces and behaviors, by hackers for purposes such as profit or censorship. The critical visual security and privacy issues on mobile devices have led to extensive discussions [32]–[34]. Related work can be classified into: Post-capture Protection and Pre-capture Protection.

Post-capture protection involves safeguarding visual content after it has been digitized by the camera. Many discussions assume the camera subsystem on mobile devices is trustworthy and focus on other aspects of privacy protection [32]–[34], such as computational overhead [35], and timing channels [36]. However, cameras on mobile devices are subject to various attacks in practice, making the camera subsystem insecure. Security flaws in the camera's firmware, processing software, and network protocols can be exploited by attackers to access raw video content or compromise the entire camera subsystem. Trusted computing, based on isolation and verification techniques, can ensure that specific computing components are not easily tampered with. Early work utilized the Trusted Platform Module (TPM) chip to build cameras [37], [38]. TPM allows for secure boot and has a sealed key for hardware encryption, but it does not protect video processing components, presenting a weaker threat model. Little work has been done to realize trusted cameras due to the complexity of the camera driver and video processing stack (e.g., OpenCV, TensorFlow). Hence, a better approach is to protect images/videos on mobile devices before capturing them, which is pre-capture protection.

Pre-capture protection strategies aim to secure visual information before it is captured by the camera. Common methods include covering the front camera with stickers or tape, a practice even adopted by tech leaders like the Meta CEO, as shown in Figure 1.8. However, this approach is aesthetically unpleasing, can easily fall off, and thus compromises privacy. It also requires manual removal whenever the camera is needed, which is inconvenient. Other approaches involve using optical filters or lenses [41], [42] to blur or defocus objects, thus obscuring sensitive information, such as faces, while still allowing for the detection of human actions. However, these methods primarily focus on specific techniques and components and lack flexibility, making them difficult to adapt to different types of sensitive information. Our goal is to develop a flexible and trustworthy camera subsystem for future mobile devices.

Compared to the state of the art, the rise of full-screen mobile devices offers an opportunity for innovative protective strategies. Motivated by the new feature of placing the transparent screen region above the under-screen camera, we aim to use this region to introduce unique optical perturbations: *Screen Perturbations*, to the image formation in through-screen computing. These perturbations will be embedded into the images captured by the under-screen camera, effectively ensuring visual privacy in full-screen mobile devices without affecting normal screen use.

1.5. Thesis Contributions and Outline

This dissertation contains four chapters, detailing how we design and implement through-screen VLC and screen perturbation. The contributions and outline are as follows:

SpiderWeb: Enabling Through-Screen Visible Light Communication - Chapter 2. Intensity-based modulations are widely used in VLC systems. In through-screen VLC, visible light is greatly attenuated when it passes through the screen because the transmittance of transparent screens is still low [10], [43]. Additionally, the detected screen light intensity at the receiver is much higher than the attenuated modulated visible light, making it difficult to eliminate the interference of dynamic screen light on intensity-based VLC signals. These issues discourage the use of intensity-based modulations in through-screen VLC. Compared with intensity-based modulations, the demodulation of Color-Shift Keying (CSK) signals relies less on signal strength and more on detected color positions in the standard color space. However, a common color sensor on mobile devices, such as an ambient light sensor, needs a certain amount of time, known as the integration or conversion time, to collect enough photons to output the detected color signals. Therefore, the signal received by the color sensor is not an “instantaneous” CSK signal but an *integrated* signal. In through-screen VLC, the color sensor is placed under the screen, integrating not only the modulated CSK signal but also the ‘*random*’ light emitted by the screen. In practice, canceling this screen light interference is impossible because the screen’s ON and OFF states and the modulated CSK signals cannot be synchronized. Thus enabling reliable through-screen VLC is very challenging.

In Chapter 2, we investigate the characteristics of through-screen VLC channel using an off-the-shelf transparent OLED screen. The transparent screen is placed between the VLC transmitter (i.e., LEDs) and the receiver (i.e., under-screen color sensors). This setup helps explore various properties of the through-screen VLC channel, such as through-screen light attenuation and screen interference. We identify a unique phenomenon termed *“color-pulling effect”*, which pulls the original modulated color towards the screen’s color. For example, when an LED emits green light and there is no screen covering the color sensor, the detected color is green. However, when a red screen is placed between the green LED and the color sensor, the detected color appears yellow, which is between green and red. Further measurements show that the received color is a mixture of different proportions of the modulated color and the screen color. This effect is attributed to the under-screen color sensor acting as an integrator, collecting signals of two colors within the integration time and outputting their superposition. Based on this observation, we design the SpiderWeb CSK (SWebCSK) modulation scheme to counteract the color-pulling effect. We prototype the first through-screen VLC system and rigorously evaluate its performance in various scenarios. The results demonstrate that SWebCSK significantly reduces the bit error rate by two orders of magnitude and achieves a 3.4× increase in data rate compared to existing solutions.

When VLC Meets Under-Screen Camera - Chapter 3. Although using color sensors as VLC receivers can successfully realize through-screen VLC, color sensors can be a bottleneck as they have a relatively low sampling rate. This is because color sensors are usually used as ambient light sensors on mobile devices, which are typically optimized for dynamic range rather than response speed. Meanwhile, when multiple transmitters send data concurrently to improve throughput or enable multi-user communication, the CSK symbol received at the single-pixel color sensor could be a superposition of the CSK symbols sent from multiple transmitters, causing interference. Without a particular design of the transceivers, the system performance can degrade significantly. A possible solution is to exploit under-screen multi-pixel cameras as the receiver to leverage light spatiality and overcome interference from multiple transmitters. Hence, we explore the Under-Screen Camera (USC) as the receiver to enhance the throughput and communication range of through-screen VLC. However, we observe a severe performance degradation in through-screen VLC with the USC since it is susceptible not only to the color-pulling effect but also to new multi-pixel interference from the screen. The interference at the multi-pixel level originates from different areas of the screen, significantly degrading the performance of through-screen VLC with USC.

In Chapter 3, we leverage the unique spatiotemporal features of the rolling shutter effect on USC to design an algorithm to identify the sampling points with minimal interference from the transparent screen. Additionally, we propose a novel demodulation method to address the color shift caused by the screen leakage interference. We build a proof-of-concept prototype using two COTS smartphones. Experimental results show that the proposed designs reduce the BER by two orders of magnitude on average and improve the data rate by 59×: from 914 b/s to 54.43 kb/s. The transmission range is also extended by approximately 100×: from 0.04 m to 4.2 m.

Screen Perturbation: Adversarial Attack and Defense on USC - Chapter 4. Cameras with exceptional visual sensing capabilities, ubiquitous in consumer mobile devices, have raised significant privacy concerns. Motivated by the interference that through-screen VLC brought in full-screen devices, we notice a new security vector: the transparent screen is placed above the USC, inevitably introducing optical perturbations on the image formation of the USC. When the screen is lit up for display, the lighting of different R/G/B subpixel sets embeds various perturbations – speckled color blocks and color shifts – in the final image captured by the USC. These perturbations can be exploited for both attacking and safeguarding purposes from a security perspective.

In Chapter 4, we propose *Screen Perturbation*, which modifies the pixels displayed on the transparent screen to nullify deep learning models such as image classification and face recognition models. These perturbations can affect deep learning models in image classification and be employed for user privacy protection or, in the case of malicious software, disrupt the front camera's functionality. We first model the under-screen image formation process of USC, considering perturbations introduced by the transparent screen. Then, we design two image-specific perturbation methods to add perturbations to the images captured by USC and successfully fool various deep learning models. Evaluations with three commercial full-screen smartphones on testbed datasets and synthesized datasets show that our screen perturbations can significantly decrease the average image classification accuracy.

Unicorn: Securing USC with Universal Screen Perturbation - Chapter 5. *Camfecting* [44] refers to unauthorized access and activation of device cameras without the owner's consent. Through camfecting, hackers can remotely capture images, record videos, or conduct real-time monitoring. Sensitive information, such as facial data and password notes, are often at risk of exposure through front-facing cameras [45]–[47]. Furthermore, using advanced deep neural networks, hackers can now efficiently mine images for sensitive information [48], [49]. With the emergence of full-screen devices, their equipped transparent screen offers a good opportunity to tackle camfecting.

In Chapter 5, we propose *Unicorn*, utilizing the transparent screen region above the USC to develop a universal, scene-independent screen perturbation that can prevent hackers' machine learning models from accurately recognizing user identity through facial images and mining password notes or other sensitive content. This perturbation is robust and effective in various conditions including different shooting distances, angles, and camera settings. It provides an imperceptible yet effective defense against camfecting, preserving user privacy without affecting the user experience. Experimental results show that Unicorn achieves over 90% protection against image classification under various hacker controls and shooting scenarios. We further achieve 100% success against advanced image classification services from Google (Google Vision API [50]) and OpenAI (ChatGPT Vision API [51]).

The contributions in these chapters have resulted in the following publications.

- **Hanting Ye**, Qing Wang, *"Computing behind Transparent Screen"*, In Proceedings of the International Conference on Embedded Wireless Systems and Networks **(EWSN'24)**, 2024. **(Chapter 1 and Chapter 6)**
- **Hanting Ye**, Qing Wang, *"SpiderWeb: Enabling Through-Screen Visible Light Communication"*, In Proceedings of the ACM Conference on Embedded Network Sensor Systems **(SenSys'21)**, 2021. **(Chapter 2)**
- **Hanting Ye**, Jie Xiong, Qing Wang, *"When VLC Meets Under Screen Camera"*, In Proceedings of the ACM Conference on Mobile Systems, Applications, and Services **(MobiSys'23)**, 2023. **(Chapter 3)**
- **Hanting Ye**, Guohao Lan, Jinyuan Jia, Qing Wang, *"Screen Perturbation: Adversarial Attack and Defense on Under-Screen Camera"*, In Proceedings of the ACM Conference on Mobile Computing and Networking **(MobiCom'23)**, 2023. **(Chapter 4)**
- **Hanting Ye**, Guohao Lan, Jinyuan Jia, Qing Wang, *"Unicorn: Securing Under-Screen Camera via Universal Screen Perturbation"*, under submission. **(Chapter 5)**
- **Hanting Ye**, Niels van der Kolk, Qing Wang, *"NIRF: Detecting Cameras That Hide Behind Screen"*, Conditionally Accepted by the ACM Conference on Mobile Computing and Networking **(MobiCom'25)**, 2025. **(Chapter 6)**

2

SpiderWeb: Enabling Through-Screen Visible Light Communication

In the previous chapter, we presented that light and radios are expected to complement each other in 6G networks, while radio communication still dominates in 4G/5G. The key advantage of light-based communication is that it mitigates the severe issue of spectrum shortage, and its wide frequency band can be utilized to support a high data rate [52], [53]. Hence, Visible Light Communication (VLC) has been considered one of the key enablers for future wireless networks such as 6G [18], [19]. Because of these advantages, VLC has attracted attention from both academia and industry.

Although significant progress on research, development, deployment, and standardization of VLC is being witnessed, a direction has been largely neglected by the community: *how to deploy VLC practically on current/future devices, especially on the smartphones that are heavily used by billions of people?* Here, *where* on mobile devices, such as smartphones, to place VLC receivers is ultra important. In state-of-the-art trials of VLC on smartphones, the receivers are mainly deployed on the back and top edge of smartphones, and/or on the smartphone screen [54], [55]. Since VLC access points are usually placed on the ceiling (used for both illumination and communication) and the VLC signal quality is dominated by line-of-sight links, placing VLC receivers onto the front part of smartphones will be beneficial and indispensable.

Motivation. In today's most advanced *narrow-bezel* and *no-bezel* mobile devices, traditional sensors such as fingerprint sensors, ambient light sensors, and even cameras have been deployed or are being evaluated *under the screen* [56], [57]. Motivated by this existing technical evolution on mobile devices, in this chapter, we ask the following question:

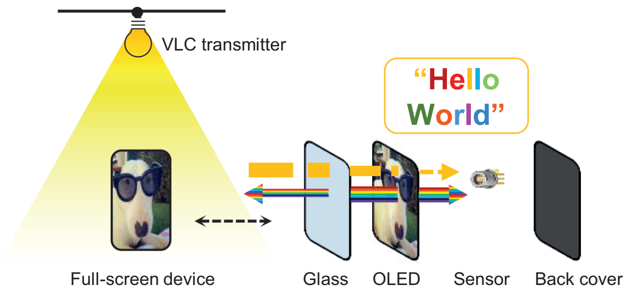


Figure 2.1: Motivation of this chapter: there is no space to deploy VLC receivers on the screen of narrow- and no-bezel smartphones. A possible solution could be to deploy VLC receivers under the screen, but the screen will become a 'blocker' and an interferer for the line-of-sight light communication.

Is it possible to also place visible light receivers under the screen to create through-screen VLC systems, as depicted in Figure 2.1?

If this exciting objective can be achieved, we can boost VLC deployment on future smartphones because it does not sacrifice their full-screen designs.

Why CSK? Although through-screen VLC looks promising, the Organic LED (OLED) screen in between the VLC transmitter and the receiver might seriously degrade the system performance because the screen could become a blocker. After the modulated light passes through the glass of an OLED screen, the light intensity is sharply reduced. Moreover, the modulated light signals are seriously interfered with by the light emitted from the screen when the screen is on. Since the VLC receiver is closer to the screen than to the VLC transmitter, the intensity of the light emitted by the screen is often several orders of magnitude higher than the intensity of the modulated light from the transmitter. Without perfect synchronization between the transmitter and the screen, it would be difficult to demodulate intensity-based VLC signals, such as ON-OFF Keying (OOK) [58], Pulse Position Modulation (PPM) [59], Pulse Width Modulation (PWM) [60], and Pulse Amplitude Modulation (PAM) [61]. Compared with intensity-based modulations, color-based modulations such as Color-Shift Keying (CSK) only care about the chromaticity of visible light and is not sensitive to the light intensity. Moreover, by utilizing CSK modulation, more bits can be encoded, and the through-screen VLC data rate can be increased. For this reason, we exploit the color domain to realize through-screen VLC.

Challenges and Contributions. It is challenging to realize reliable CSK-based VLC because the screen content dynamically changes, interfering with the CSK signals. If we adopt the CSK constellation points specified in the IEEE 802.15.7 VLC standard [16], these points will be severely offset. In Chapter 1, we listed the primary challenges related to the screen in through-screen computing in brief. This section comprehensively explains the concerns in achieving the through-screen communication link between the LED transmitters and the under-screen receivers. Thus, the first challenge in designing a through-screen VLC system is

Challenge 1: *How to decode the signal that has been interfered by the screen light? In*

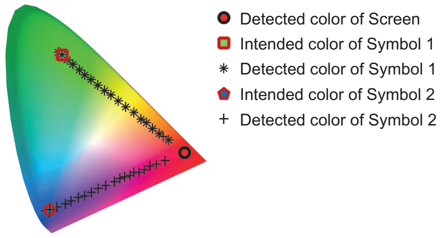


Figure 2.2: Color-pulling effect caused by the transparent screen at the under-screen sensor of VLC receiver.

particular, the screen color changes dynamically when displaying different contents.

In this chapter, we observe from measurements that the received CSK constellation points shift toward the screen color point after being interfered with by the screen, as illustrated in Figure 2.2. The positions of the received symbols are distributed between the transmitted CSK symbol constellation points and the detected screen color point. That is, the transmitted CSK symbols are *pulled* to the screen color point. We refer to this phenomenon as *color-pulling effect* introduced by the screen on the CSK signals. Motivated by the observation that the received symbols form lines from the CSK symbol constellation points to the screen color point, we propose using a slope-based demodulation method to distinguish the received symbols. It works well for the CSK with low modulation levels, such as 4-CSK.

Although the slope-based demodulation algorithm can reduce the Bit Error Rate (BER) in CSK-enabled through-screen VLC, we encounter the second challenge:

Challenge 2: *With higher-level CSK modulations specified in the IEEE 802.15.7 standard (e.g., 8-CSK and 16-CSK) and under many screen colors, the slope-based demodulation alone cannot distinguish the received constellation points.*

In higher-level CSK modulations such as 8-CSK and 16-CSK, some of the lines (caused by the color-pulling effect) that connect their constellation points with the screen color point could *overlap* with each other, making the demodulation of these constellation points almost impossible.

To tackle this challenge, we design a new color-based modulation scheme named SpiderWeb Color-Shift Keying (SWebCSK). The design is motivated by the structure of spiderweb. We set the detected screen color at the sensor as the hub of the spiderweb, and search for the suitable constellation points from each ray of the spiderweb. We maximize not only the Euclidean distances among constellation points, but also maximize the minimal angle between any two lines caused by the color-pulling effect. This design ensures that the lines will not overlap with each other, benefiting the demodulation of received color-modulated symbols at the receiver.

We have also tackled other system-level challenge:

Challenge 3: *How to practical design and implement a through-screen VLC system?*

When generating CSK/SWebCSK signals at the transmitter, a linear current drive could be leveraged. However, the normalized radiant power of the tri-color RGB LEDs versus the DC drive current is not linear, additionally considerable hardware overhead and complexity will be introduced. In this chapter, we use three high-frequency PWM signals to control each channel of the tri-color LED and we identify the formulation to calculate the duty circles of each PWM signal, considering the power/non-flickering constraint and the attenuation of visible light in the air as well as when the light passes across the OLED screen. We also solve at the receiver an over-saturation problem that is specific in through-screen VLC.

We successfully prototype the proposed system using off-the-shelf hardware. Combining all the design components, we successfully enable through-screen VLC under different screen colors and different screen brightness. The proposed SWebCSK modulation and the slope-based demodulation could achieve very low BER in different scenarios. Below we summarize our main contributions:

- We propose the concept of through-screen VLC and validate its feasibility by prototyping with off-the-shelf devices.
- We observe a color-pulling effect of transparent screens on color-based signals. This effect greatly degrades the BER performance of through-screen VLC systems.
- We design a color-based and spiderweb-inspired SWebCSK modulation scheme. Together with our proposed slope-based demodulation, we can improve the BER performance greatly under different screen color and brightness.
- We thoroughly evaluate the performance of our system in different scenarios. The results demonstrate that compared to existing solutions, our methods can reduce the BER by two orders of magnitude and achieve a 3.4× data rate.

2.1. Fundamentals of Screen and Color

Before we present discoveries and proposed modulation and demodulation techniques for through-screen VLC systems, we briefly explain the fundamentals of screens used in mobile devices, standard color spaces, and CSK modulation used in VLC systems.

2.1.1. Types of Screen

There are currently two main types of screens: Liquid Crystal Display (LCD) and OLED. Below we briefly introduce them.

LCD screen. It has three main components: backlight panel, liquid crystal, and two polarizing plates called Polarizer and Analyzer, as illustrated in Figure 2.3(a). The liquid crystal does not emit light; thus, a backlight panel is needed (except in reflective displays) as a light source to illuminate the display panel. The liquid crystal layer can change the polarization direction of incident light when applied different voltages, thereby controlling the light/dark states of the screen. This feature has been used for passive VLC [62]–[65].

OLED screen. It also has a sandwich structure. The difference is an OLED screen does not need a backlight panel, and the thick liquid crystal layer is replaced by a thin organic emissive layer wrapped by the anode and cathode, as shown in Figure 2.3(b). Thus, an OLED screen is thinner than an LCD screen. To display content, the anode of OLED is built with transparent materials and covered with glass. An OLED screen is self-luminous.

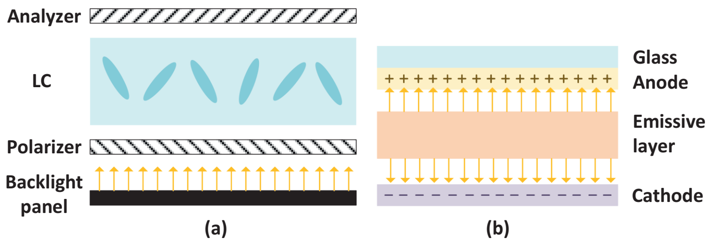


Figure 2.3: Simplified diagrams of screens: (a) LCD; (b) OLED. An OLED screen is thinner and lighter than an LCD screen.

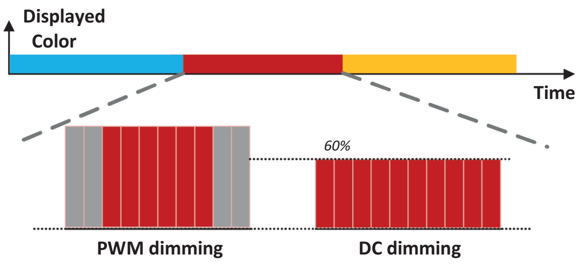


Figure 2.4: OLED screen refresh rate and brightness control.

The cathode can also be built with transparent electrode materials such as transparent indium tin oxide to form a *transparent screen*.

2.1.2. SCREEN REFRESH RATE AND BRIGHTNESS CONTROL

The refresh rate is an important screen parameter that determines the quality of the screen. The screen refresh rate of common devices such as smartphones, monitors, and TVs, is 60 Hz. Some high-end monitors have supported a refresh rate of 144 Hz. In recent years, the refresh rate of smartphone screens is being increased to 90 Hz and 120 Hz. For any screens, the content displayed on the screen remains unchanged between two consecutive screen refreshes.

There are two ways to control the brightness of a screen. One is Direct Current (DC) dimming, which changes circuit current to control the power of the emitted light. The other one is Pulse-Width Modulation (PWM) dimming that changes the number of ON and OFF states of the screen in a unit of time to have various brightness, as shown in Figure 2.4. PWM dimming is widely used in OLED screens.

2.1.3. STANDARD COLOR SPACE

A popular color space chromaticity diagram is CIE 1931 [66],[1] which mimics human perception of color. *The diagram maps all colors perceivable by human eyes to two chromaticity parameters - x and y -* as shown in Figure 2.5, where the monochromatic light with different wavelengths forms the line on the periphery of the horseshoe shape. Each

[1]CIE is the French abbreviation for the International Commission on Illumination.

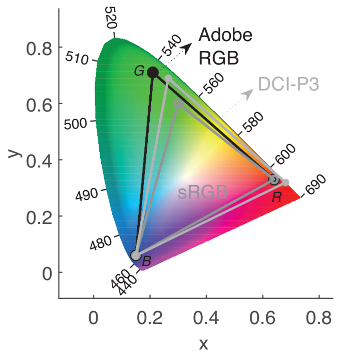


Figure 2.5: CIE color space.

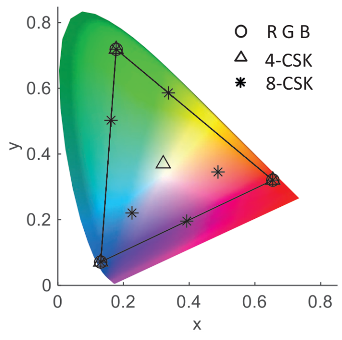


Figure 2.6: CSK modulation.

screen might have a different color display range, as shown in Figure 2.5. The sRGB and DCI-P3 color gamuts are widely used in devices; the Adobe RGB color gamut is mostly used in professional photography.

2.1.4. Color-Shift Keying Modulation

The Color-Shift Keying (CSK) modulation scheme was proposed in the IEEE 802.15.7 standard for VLC [16]. CSK exploits three separate (Red (R), Green (G), and Blue (B)) LED channels in LED luminaires to generate white light. With three LED channels R/G/B, such luminaires can be configured to provide a variety of colors using a mixture of Red, Green, and Blue. CSK modulates the signal by modifying the output power combinations of the three colors. Since different devices use different RGB color spaces, we need a standard color space to unify color standards. Depending on the operating frequency of the red, green, and blue LEDs of the luminary, a *modulation triangle* is formed within the standard color space, as indicated in Figure 2.6. Regardless of whether the screen uses sRGB or Adobe RGB, most of its color space is included in the modulation triangle of this chapter. The constellation symbols are chosen inside the triangle. In addition, given the RGB coordinates of the modulation triangle,[2] the CSK constellation points, as shown in Figure 2.6, can be directly calculated from the specification in [16]. Compared with intensity-based modulations (e.g., OOK, PPM, PWM, and PAM), the demodulation of color-based CSK signals does not depend on the signal strength, but exploits the position of the detected CSK signals mapped in the CIE color space.

2.2. SpiderWeb Color-Shift Keying

In this section, we first introduce our observation of OLED screen's impact on traditional CSK signals. Then, we present SpiderWeb CSK (SWebCSK), a key scheme we designed to enable through-screen VLC.

[2]In IEEE 802.15.7, IJK is used to represent the variable apex coordinates of the modulation triangle. For simplification in this chapter, I is represented as the red coordinate point R, J as the green coordinate point G, and K as the blue coordinate point B.

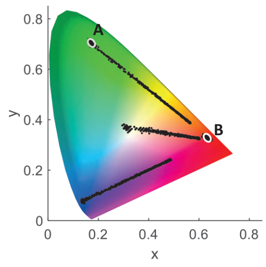


Figure 2.7: 4-CSK.

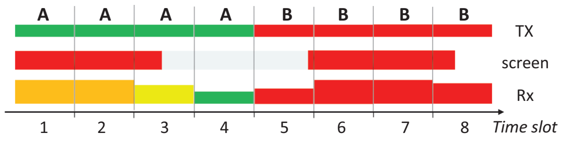


Figure 2.8: Time domain illustration of the screen's color-pulling effect on CSK signals.

2.2.1. Screen's Color-Pulling Effect on CSK Signal

Equipped with a true color sensor, the VLC receiver can directly read the X, Y, and Z values of detected CSK signals, which are the imaginary three primary colors after the R, G, and B transformations to meet the CIE 1931 standard chromaticity observer function. The transformation of coordinates from X, Y, and Z values to the CIE 1931 space is

$$x = X \big/ (X+Y+Z)\,, \quad y = Y \big/ (X+Y+Z)\,. \tag{2.1}$$

When the visible light passes through the color filter of a color sensor, a portion of the photons is filtered out. To collect enough photons, a common color sensor needs to wait for a certain amount of time, known as the integration time or the conversion time (cf. the datasheet of the sensor TCS34725 for example [67]). Therefore, what we receive through the color sensor is not an "instantaneous" CSK signal but an *integral* signal. As introduced in Section 2.1.2, due to the simplicity of hardware, PWM dimming is usually adopted by OLED screens for brightness control. To avoid flickering, the PWM controlling signal has a much higher rate than the screen refresh rate. Thus even when the color displayed on the screen does not change between two screen refreshing actions, the state of an OLED screen actually switches very fast between ON and OFF. In through-screen VLC, the color sensor (ambient light sensor) is placed under the screen. Therefore, the color sensor integrates not only the transmitted CSK signal but also the 'random' light emitted by the screen. In practice, it could be impossible to cancel the screen light interference because the ON/OFF state of the screen and the transmitted CSK signals cannot be practically synchronized.

A. Color-pulling effect

We use two illustration figures to show this effect. The transmitter sends the 4-CSK symbols calculated according to [16]. The transparent OLED screen is placed between the transmitter and the receiver, and the color sensor of the receiver is deployed under the screen. The experimental color gamut results are shown in Figure 2.7. We could observe a phenomenon in which the detected CSK signals at the color sensor are always on the line segment connecting the sending symbol point and the screen color point.[3]

[3]Most modern devices, such as smartphones, use professional programs to calibrate their screen color to the standard color spaces as introduced in Section 2.1.3. Thus, we do not need additional calibration procedures to get the accurate screen color coordinates.

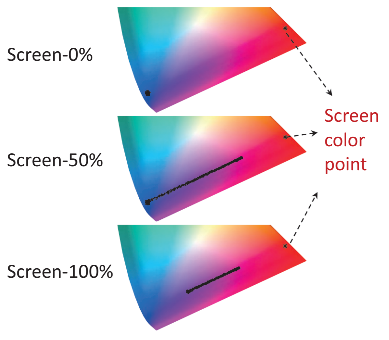
We denote the CSK symbol for the green light as A, and the CSK symbol for the red light as B. As shown in Figure 2.7, when symbol B is transmitted and the screen color is red, the reception of symbol B is not affected. Nevertheless, when symbol A is transmitted and the screen color is red (corresponding symbol B), we come with a line segment from A to B, where the starting point of the line segment is symbol A, and the endpoint is close to B. This means that when the screen is illuminated, the original symbol A (when the screen is not illuminated) is pulled closer to the symbol B. We call this phenomenon as the *color-pulling effect* of the screen on CSK signals, *which pulls the originally sent CSK symbol points to the color point of the screen.*

From the time-domain's perspective, we assume that the transmitter sends symbol A and symbol B successively within 8 time slots (the length of each time slot is equal to the integration time of the color sensor). Figure 2.8 shows the color signal receiving process, which contains the color symbols sent by the transmitter, the color displayed on the screen,[4] and the color detected at the under-screen sensor. Because the transmitter does not know the specific period of the screen signal, the transmitted signal and the screen signal are not synchronized. In addition, the screen adopts PWM dimming. In other words, the screen is ON within a certain period of time, and is OFF in another period of time. Also, the frequency of PWM is much lower than transmitted signal frequency. As shown in Figure 2.8, in the first two time slots, the symbols sent are wholly disturbed by the screen, so the original green symbol point of the sensor output is pulled to the orange symbol point. In the third time slot, a jump occurred in the middle of the screen, changing from the original red light to no light, and thus the sending symbol A is only disturbed by part of the red light of the screen. Therefore, the symbol points collected by the sensor in the third time slot are yellow symbol points. The transmitter starts to send the red symbol point B from the 5th time slot. Because the screen is transparent or red at this time, from the 5th to the 8th time slot, the output of the sensor in the receiver is still at the red symbol point. That is, the red symbol point A is not affected when the screen color is red. The above analysis in the time domain explains in principle the occurrence of screen's color-pulling effect on the CSK signals.

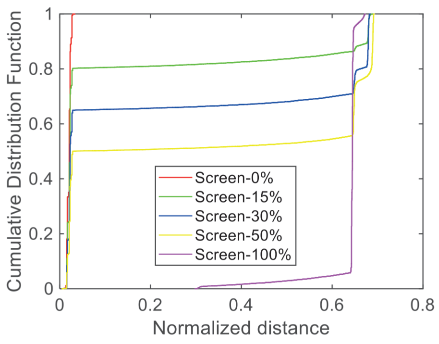
B. Impact of the screen brightness.

The distribution of the points on the line connecting the constellation point and the screen color point depends on the screen brightness. Interestingly, when the screen brightness increases, more points are shifted from the original constellation point to distribute on the line. This can be observed from our measurements shown in Figure 2.9(a). The result in Figure 2.9(b) is to analyze the distribution of constellation points under different screen brightness. When the screen is OFF, there is no interference from the screen color. Thus, the sampled points are distributed around the original constellation points. When the screen is fully ON, there is already interference from the screen, and thus, the received points are nearly distributed around the screen color. When the screen is ON but its brightness is less than 100%, the points are distributed on the line as well as the constellation and screen color points. Also, we calculate the normalized

[4]The screen usually displays a multi-pixel image so that the overall screen shows a variety of colors. However, the sensor has a small sensing area under the screen. Thus, we assume that the screen color perceived by the sensor is the color of a pixel.



(a)



(b)

Figure 2.9: Color-pulling effect vs. screen brightness.

distance between interfered point and constellation point with respect to the distance between the constellation point and the screen color point.

C. IS A SLOPE-BASED DEMODULATION METHOD ENOUGH FOR TACKLING THE COLOR-PULLING EFFECT ON CSK SIGNALS?

Since the points are distributed on the lines, we could not use the traditional method to demodulate the received data. But this line has motivated us to design a slope-based scheme to demodulate the received signals. The color-pulling effect from the screen inspires us not only to use the distance between each constellation point in the CIE 1931 space to demodulate by implementing minimum Euclidean distance detection, but also to use the slope from each constellation point to the screen color point for demodulation, that is, use the angular space of CIE 1931 space. This *only* works with low-level CSK modulation under certain screen colors such as the 4-CSK under the red displayed on the screen (cf. Figure 2.7), where the slopes are different. However, when the screen color is yellow-green, the slopes of the blue symbol point and the white symbol point of 4-CSK are overlapped. Especially, in CSK with higher modulation levels such as 8-CSK under red screen as shown in Figure 2.10, there are more possibilities for multiple CSK symbols to overlap with each other, which is completely indistinguishable. Therefore, we need to redesign the distribution of the constellation points.

2.2.2. SWEBCSK OVERVIEW

Different from the traditional CSK where the constellation points in the constellation diagram are fixed for a certain modulation level, in our proposed SWebCSK, the constellation points are optimized dynamically based on the current screen color detected at the under-screen sensor of the VLC receiver. By doing this, we could significantly alleviate the color-pulling effect caused by the screen and thus enable through-screen VLC.

Defining the line connecting the current screen color point and a SWebCSK constellation point as a *ray*. To ensure the through-screen VLC signals have a low bit error rate, we should maximize the distances between the constellation points. Due to the color-pulling effect of the screen, we should also separate the rays as much as possible. Based

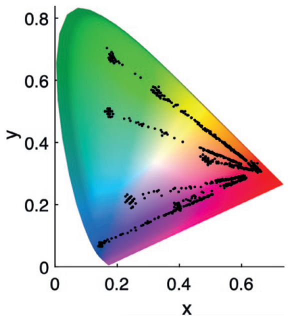


Figure 2.10: 8-CSK.

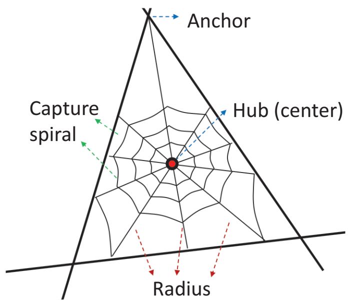


Figure 2.11: Structure of a spiderweb.

on these intuitions, we have the following designing principles for M-SWebCSK, where M is the modulation level:

1. The minimum Euclidean distances among the M constellation points should be as large as possible;
2. Rays should be separated from each other to the largest extent. In particular, two rays should not overlap with each other (recall the issue in 8-CSK of Figure 2.10).[5]

Although we can form an optimization problem to find the constellation points, the complexity is usually high because the problem would be non-conductive and non-convex [68]. To reduce complexity, in this chapter, we instead propose a heuristic to obtain the proper SWebCSK constellation points. Next, we present the details.

2.2.3. SWebCSK Constellation Design

Our solution is motivated in part by the structure of the spiderweb. In a spiderweb as shown in Figure 2.11, *hub/center* is a place for the spider to rest; *radius* is composed of non-stick lines for the spider to crawl on; *capture spiral* is built with sticky silk for catching insects. The constellation design in SWebCSK is to select the most suitable intersections of the capture spiral and the radius to meet with the two design principles presented in Section 2.2.2.

Before introducing the details, we first make the following definitions. For the triangle that confines the modulation space, we define the three apexes as $R \triangleq (x_R, y_R)$ (Red), $G \triangleq (x_G, y_G)$ (Green), $B \triangleq (x_B, y_B)$ (Blue). These points are fixed. We define the screen color at a time slot as $S \triangleq (x_S, y_S)$. This point changes based on the color displayed on the screen. For the M-SWebCSK, we need to find M constellation points in the modulation space. Denote the designed CSK constellation points as $u_i \triangleq (x_i, y_i), \forall i \in \{1, \cdots, M\}$.

Motivated by how a spider weaves its web, in our algorithm, we first establish a spiderweb-alike frame within the modulation space. This frame will give us the list of potential constellation points. Then we use a greedy algorithm to choose the optimal constellation points for our SWebCSK. Without the loss of generality and for explanation simplicity, we assume the screen is displaying Red color. That is, S and R overlap with

[5]An exception is when the screen color point falls in between two constellation points.

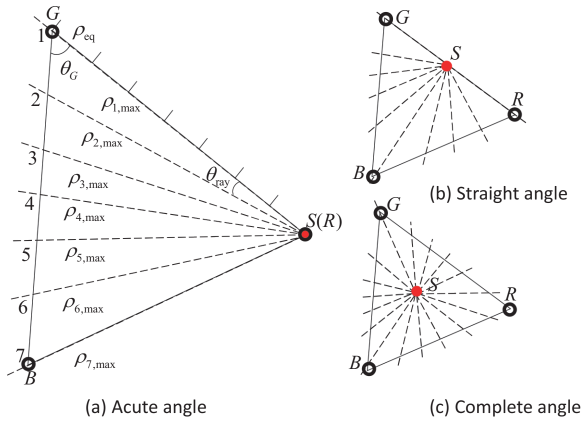


Figure 2.12: Model and modulation angle.

each other in the constellation space, as illustrated in Figure 2.12(a). We also use this figure to explain our proposed algorithm. The algorithm consists of three steps:

Step 1) Establish the spiderweb-alike framework. To build such a framework, we first need to find a primary constellation point, which is similar to the hub or center of a spiderweb where the spider rests. Since the candidate constellation point that overlaps with the screen color point is least affected by the color-pulling effect of the screen, we use this one as the primary constellation point. We number it as the M-th constellation point, that is, $u_M = (x_S, y_S)$.

Once we identify the hub constellation point, we calculate the modulation angle θ. This angle determines how many rays can we embed into the modulation space, i.e., affecting the modulation level of our proposed SWebCSK. The larger the modulation angle θ, the higher M will be. A higher M means more bits can be transmitted in the unit time/symbol. But note that with a specific modulation angle, a higher M usually leads to higher BER. For the modulation angle, there are only three cases: 1) *Acute angle,* when the screen color point overlaps with one of the vertexes of the modulation space; 2) *Straight angle ($\theta = 180°$),* when the screen color point falls on the edges of the modulation space (excluding the three vertexes); 3) *Complete angle ($\theta = 360°$),* when the screen color point is within the triangle. These three cases are illustrated in Figure 2.12.

To meet the second requirement (cf. Section 2.2.2), the minimum angle can be maximized by evenly splitting the modulation angle θ. Therefore, the screen coordinates can be the origin, and each ray equally divides the modulation angle. Note that to eliminate the color-pulling effect, only a point on each ray can be selected as a constellation point in the proposed SWebCSK to guarantee that there is a minimum angle between any two constellation points and the screen color point. In addition to the point u_M, we still need $(M-1)$ rays to extract another $(M-1)$ constellation points. When θ is an acute angle or a straight angle as shown in Figure 2.12(a) and (b), we need $(M-1)$ rays to form $(M-2)$ angles; when θ is a complete angle, $(M-1)$ rays form $(M-1)$ angles. Denoting θ_{ray} as the angle between each two neighboring rays, we have

$$\theta_{\text{ray}} = \begin{cases} \theta/(M-2), & \text{if } \theta \le 180° \\ \theta/(M-1), & \text{if } \theta = 360° \end{cases}. \tag{2.2}$$

Therefore, if the screen color point S is on the three edges of the modulation triangle, then 180 degrees could be leveraged to split the $(M-1)$ rays, and thus, a higher level of SWebCSK could be formed. Furthermore, if the screen color point S is within the modulation triangle, 360 degrees can be used, and the level of the SWebCSK could be further improved. In summary, the current screen color has a significant impact on the levels of SWebCSK we would form.

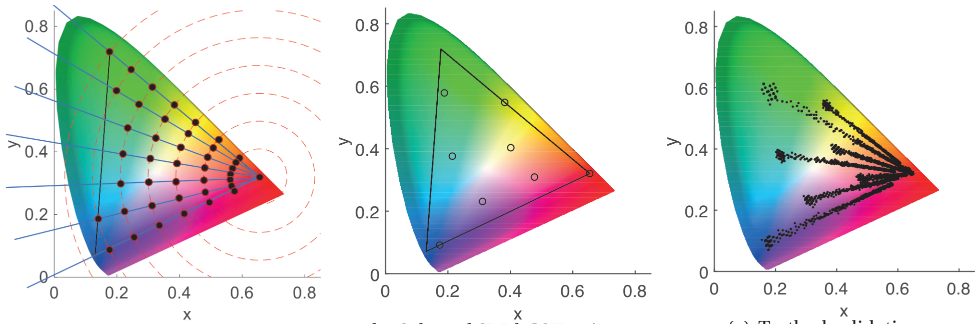
After obtaining the rays, we continue to find the suitable constellation points on each ray. Again, we follow the building process of a spiderweb. In the framework with the $(M-1)$ rays, we draw concentric circles to connect the rays, all centered around the primary constellation point (screen color point). The maximum radius of the circle that has an intersection with the modulation triangle is $\rho_{\max} \triangleq \max\{d_{SR}, d_{SG}, d_{SB}\}$. Let K denote the number of circles. The larger the K, the more potential constellation points we will have. For simplicity, we assume the inter-circle distance is fixed and denoted by ρ_{eq}. Then we have $\rho_{\text{eq}} = \frac{\rho_{\max}}{K-1}$. The intersection of each ray and each circle can be a candidate CSK constellation point for CSK on that ray. According to the law of sines, the maximum length of the i-th ray within the modulation triangle can be calculated as

$$\rho_{i,\max} = \begin{cases} d_{\text{SG}} \dfrac{\sin\theta_G}{\sin(\theta_G + (i-1)\theta_{\text{ray}})}, (i-1)\theta_{\text{ray}} \le 90°, \\ d_{\text{SB}} \dfrac{\sin\theta_B}{sin(\theta_B + (i-1)\theta_{\text{ray}} + \theta_{\angle GSB})}, 90° < (i-1)\theta_{\text{ray}} \le 180°, \qquad \forall i \in \{1, \cdots, M-1\} \\ d_{\text{SR}} \dfrac{\sin\theta_R}{sin(\theta_R + (i-1)\theta_{\text{ray}} + \theta_{\angle GSB} + \theta_{\angle BSR})}, (i-1)\theta_{\text{ray}} > 180°, \end{cases} \tag{2.3}$$

The number of candidate constellation points that can be placed on each ray is computed as $C_i = \left\lfloor \frac{\rho_{i,\max}}{\rho_{\text{eq}}} \right\rfloor + 1, \forall i \in \{1, \cdots, M-1\}$.

As a result, the set of available constellation points in the modulation triangle area is obtained. All the candidate CSK constellation points and the constructed spiderweb are shown in Figure 2.13(a).

Step 2) Select constellation points from the candidate sets. In Step 1, we have already selected the screen color point as the primary constellation point. For u_M, $\rho_M = 0$. We then put point u_M into the set $\mathscr{A}$, which is the set to store the selected SWebCSK constellation points. To select the remaining $M-1$ points, we use the "ray-point" procedure. We first identify on which ray to select the constellation point, then find the best constellation point on that ray. The constellation points should be picked from the ray with the smallest number of candidate constellation points first. Therefore, the index of the starting ray is selected as $j = \arg\min_{i\in\{1,\cdots,M-1\}} C_i$. Then we use the Max-min metric to select a point from the jth ray as a constellation point. The Max-min metric maximizes the minimum distance between the currently selected point and the previous points in



(a) All possible points (b) Selected SWebCSK points (c) Testbed validation

Figure 2.13: SWebCSK constellation design ($M = 8$).

Algorithm 1 Select constellation points from the candidate sets

```
Input: $\mathscr{A} = \{\rho_M\}$
       $C_i, i \in \mathscr{B} \triangleq \{1, \cdots, M-1\}$
Output: $\mathscr{A}$
1: Set the screen color point as a constellation point $u_M$
2: while $\mathscr{B} != \emptyset$ do
3:     Select the ray $j$ with the least number of candidate points: $j = \arg\min_{i \in \mathscr{B}} C_i$
4:     Use the max-min metric to select the point on the $j$th ray.
5:     $\mathscr{A} \leftarrow \mathscr{A} \cup \rho_j$
6:     $\mathscr{B} \leftarrow \mathscr{B} / j$
7: end while
```

set $\mathscr{A}$. According to the law of cosines, the Max-min metric can be expressed as follows:

$$n = \arg\max_{n \in \mathscr{C}_j} \min_{m \in \mathscr{A}} \sqrt{\rho_m^2 + \rho_n^2 - 2\rho_m \rho_n \cos(\theta_{mn})}\,, \tag{2.4}$$

where $\mathscr{C}_j \triangleq \{1, \cdots, C_j\}$. The distance from the selected constellation point on the j-th ray to the screen color point S is as $\rho_j = (n-1)\rho_{\text{eq}}$. Also, the angle between ith ray and jth is $\theta_{ij} = |i - j|\theta_{\text{ray}}$.

We repeat the above procedure until we find all the M constellation points. The whole procedure is shown in Algorithm 1.

Step 3) Convert the constellation points from polar coordinates to xy coordinates. The SWebCSK constellation points obtained from Step 2 are expressed in the polar coordinate with the screen color point S as the origin. They need to be further transformed into the xy coordinates in the CIE 1931 coordinate system. When the modulation angle θ is an acute angle, the conversion from the polar coordinates to the xy coordinates can be done as follows:

$$x_i = \widetilde{\rho}_i \cos\widetilde{\theta}_i\,, \quad y_i = \widetilde{\rho}_i \sin\widetilde{\theta}_i\,, \tag{2.5}$$

where $\widetilde{\rho}_i = \sqrt{\rho_i^2 + \widetilde{\rho}_S^2 - 2\rho_i\widetilde{\rho}_S \cos\lambda_i}$, $\lambda_i \triangleq \arctan|\frac{y_S - y_G}{x_S - x_G}| - \theta_{\text{ray}}(i-1) + \arctan(y_S/x_S)$, $\widetilde{\rho}_S \triangleq \sqrt{x_S^2 + y_S^2}$, and $\widetilde{\theta}_i$ is expressed as

$$\widetilde{\theta}_i = \begin{cases} \arctan\dfrac{y_S}{x_S} + \arccos\dfrac{\widetilde{\rho}_i^2 + \widetilde{\rho}_s^2 - \rho_i^2}{2\widetilde{\rho}_i\widetilde{\rho}_s}, & \lambda \geq 0, \text{or}, \lambda < -\pi \\ \arctan\dfrac{y_S}{x_S} - \arccos\dfrac{\widetilde{\rho}_i^2 + \widetilde{\rho}_s^2 - \rho_i^2}{2\widetilde{\rho}_i\widetilde{\rho}_s}, & -\pi \leq \lambda < 0 \end{cases}. \tag{2.6}$$

Preliminary validation. We use the designed 8-SWebCSK constellation points as shown in Figure 2.13(b) at the transmitter. We place the color sensor under the red screen to detect the transmitted CSK signals. The received constellation points are shown in Figure 2.13(c). It can be seen that due to the influence of the color-pulling effect, each constellation point of 8-SWebCSK shifts to the red screen point to form seven non-overlapping line segments. The endpoints of the seven lines formed simultaneously do not overlap with the origin of the red screen, which also helps detect the eighth CSK constellation point at the origin of the screen.

2.3. SYSTEM DESIGN

In this section, we present the system design of our SpiderWeb. We consider a through-screen VLC with a transmitter and a receiver, which are presented in Section 2.3.1 and Section 2.3.2, respectively. The transmitter is equipped with an RGB LED and a driver circuit. The visible light emitted from the LED is modulated in the color domain to transmit data wirelessly. At the receiver, we use a true color sensor that can directly output the X, Y, and Z values of the detected light. We consider an OLED screen at the receiver. The color sensor is placed under the OLED screen, similar to the front camera and ambient light sensor placed in the most advanced smartphones [56], [57]. The screen color detected by the color sensor of the receiver is shared with the transmitter. Next, we present the design details of the transmitter and the receiver.

2.3.1. TRANSMITTER DESIGN

The block diagram of the VLC transmitter is shown in Figure 2.14. We use three PWM signals to separately control the R/G/B channels of the LED to generate different colors. The controller generates three high-frequency PWM signals with different duty cycles through internal timers to control the average output power of the RGB LED, thereby generating different colors. Although the linear variable current driver has been used to generate different driving currents for producing different colors, it often requires the polynomial fitting of more than third order to obtain accurate current combinations[68]. Leveraging PWM signals can avoid tackling the non-linearity of LEDs. The circuit is also simple, easy to implement, and compatible with existing transmitter infrastructures.

A. GENERATING THE SWEBCSK SIGNALS

The transmitter receives the binary data and codes them by assigning designed SWebCSK constellation points in the CIE-1931 space. In order to reduce the bit error rate of

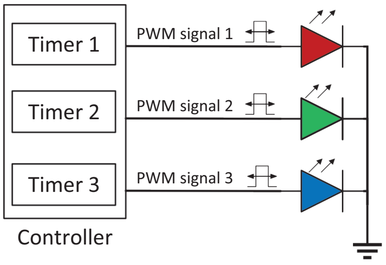


Figure 2.14: Circuit diagram of the transmitter.

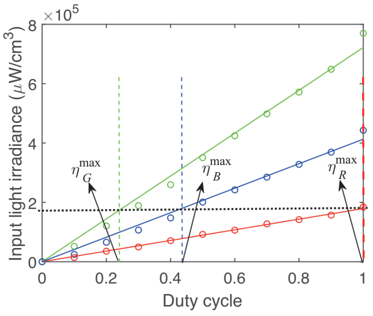


Figure 2.15: Electro-optical response of the RGB LED.

the system, SWebCSK symbols are coded by gray code from symbol 1 to symbol M. According to the current color of the screen, the transmitter calculates SWebCSK constellation points based on the method presented in the last section. For a given color constellation point, we need to know the output optical power of R, G, B LED. In order to avoid flickering, the output optical power of R, G, B LED should satisfy $P_R + P_G + P_B = 1$. Suppose that, given a RGB LED in the transmitter, a desired output chromaticity of (x_m, y_m) is required (the constellation point coordinates are obtained using the three steps presented in Section 2.2.3). Substitute the measured x_p and y_p, $p \in \{R, G, B\}$, the output optical power P_R, P_G and P_B required for RGB light can be calculated as

$$\begin{bmatrix} P_R \\ P_G \\ P_B \end{bmatrix} = \begin{bmatrix} x_R & x_G & x_B \\ y_R & y_G & y_B \\ 1 & 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} x_m \\ y_m \\ 1 \end{bmatrix}, \tag{2.7}$$

where x_m and y_m are the coordinates of the mth constellation point. Note that in the system, we do not need to calculate the inverse of the matrix in (2.7). When the LED in the transmitter is fixed, x_p and y_p, $p \in \{R, G, B\}$ are all known to the transmitter, and the inverse of the entire matrix can be calculated in advance. However, we are still unable to realize accurate CSK constellation point coordinates in the transmitter based on P_R, P_G and P_B. For example, if we realize the white point in the CIE 1931 diagram, the output power of the corresponding R, G, and B LEDs should all be 33.3%. In fact, if the PWM duty cycle is configured to (33%, 33%, 33%), the light emitted by the transmitter is not real white light. This is because the luminous efficiency of different colors of light is different. Therefore, we need to know the luminous efficiency of the RGB LEDs.

Since Y represents Luminosity in the CIEXYZ space, we have tried to use Y to simulate the result of the change in the PWM signal duty cycle. However, the actual performance is abysmal, and the modulation of any CSK constellation point cannot be achieved. Nevertheless, by consulting the datasheet of the true color sensor, X channel detects visible light with a wavelength of 600 nm, which is close to the red wavelength range. Similarly, Y channel can detect green light with a wavelength of 555 nm, and Z channel can detect blue light with a wavelength of 445 nm. Therefore, we use the X, Y, and Z channels to calculate the input red, green and blue light irradiance regarding the photodiode's area within the conversion time interval (E_e), which is expressed as $E_e = \frac{\mathrm{FSR}_{E_e}}{N_{\mathrm{CLK}}}\mathrm{MRES}$, where

FSR_{E_e} is the full-scale range of detectable input light irradiance E_e, in μW/cm^3 (can be found in [69]); N_{CLK} denotes the number of clock cycles within the predefined conversion time interval; MRES is the digital output value of the conversion (X, Y, Z).

Figure 2.15 shows the measured input light irradiance of the photodiode's area within the conversion time interval, comparing different duty cycle signals and the corresponding fitted RGB three-color power response. It is observed that red light has the lowest and green light has the highest luminous efficiency. Consequently, we set the maximum luminous power of the red light to equal the total emission power of the emitter. The symbols $\eta_{\text{R}}^{\text{max}}$, $\eta_{\text{G}}^{\text{max}}$, and $\eta_{\text{B}}^{\text{max}}$ in the figure denote the achievable duty cycle for R, G, B when total emission power is reached.

Therefore, after substituting the required CSK symbol coordinates, we can get the normalized RGB LED radiation power from (2.7). Finally, the duty cycle of the PWM signal that controls the RGB LED is calculated as: $\text{PWM}_p = P_p \eta_p^{\text{max}}, p \in \{R, G, B\}$.

B. Channel precompensation

There are two types of visible light attenuation in the system: 1) the attenuation of visible light propagating in the air, and 2) the attenuation of visible light when it passes through the screen medium, known as the Glass Attenuation Factor [70]. Different colors of light have different attenuation under these two attenuation situations. Also, the attenuation faced by different colors of light depends on the distance. In this chapter, we regard these two types of attenuation as the attenuation of the CSK symbol in the communication channel. In RF communication, the transmitted signal is usually pre-coded to resist channel fading. Similarly, three dynamic compensation factors can be added to the transmitter of the SpiderWeb system to compensate for the lower light level at the receiver according to varying distances. The calibrated duty cycle signal of the RGB LED is $\overline{\text{PWM}_p} = \text{PWM}_p + c_p, p \in \{R, G, B\}$.

To complete channel compensation, the transmitter periodically sends a pilot containing a flag and three symbols of R, G, and B light in sequence. After receiving the flag, the receiver calculates c_R, c_G, c_B, respectively, and then feeds them back to the transmitter. The transmitter will use this calibration factor until the next channel calibration. Since the pilot is sent out periodically, the transmitter can quickly adapt to changing channel conditions, allowing the receiver to detect accurate CSK constellation points.

2.3.2. Receiver Design

As shown in Figure 2.16, the receiver is composed of a transparent OLED screen, and a true color sensor is placed directly under the screen where it emits light. The sensor outputs the X, Y, Z from three channels after detecting the light. The X, Y, and Z channel data streams received by the receiver first pass through the initial module to perform oversaturation detection and restore outliers. Then through energy detection, the received signal is divided into the part interfered by the screen color signal and the part not interfered by it. Next, we use different demodulation methods to demodulate different parts of the received signal. Finally, we detect the preamble of the demodulated signal, find the starting point of the data, and map received symbols to the data bit stream.

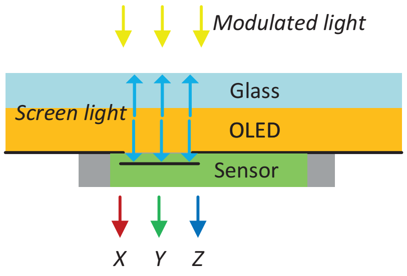


Figure 2.16: Diagram of the receiver hardware.

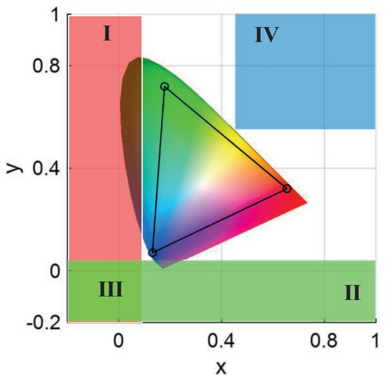


Figure 2.17: Diagram of over-saturation recovery area.

A. OVER-SATURATION RECOVERY

Since the sensor is placed directly under the screen at a very close distance, the screen light intensity detected by the sensor is extremely high. Because the ADC output accuracy range of the sensor is n bits and sometimes the ambient light intensity is too strong, the sensor at the receiver is in over-saturation, resulting in outliers in the received CSK symbols. Although we can eliminate this kind of over-saturation by reducing the gain of the sensor, it will also greatly reduce the performance of the system because the screen light intensity is much higher than the light intensity of the modulated signal. Fortunately, we find that the occurrence of over-saturation outliers often caused one or more of the X, Y, and Z channels of the output signal to recount from 0. As a result, points not located in the CIE 1931 chromaticity diagram are those over-saturation outliers. We design an over-saturation recovery method to handle occasional over-saturation outliers by compensating for them based on their specific positions. We denote the over-saturation state of the three channels of X, Y, and Z as $\hat{X}$, $\hat{Y}$, and $\hat{Z}$, respectively. As shown in Figure 2.17, the outliers are all in areas I to IV. When the outlier is in area I, the x coordinate of the outlier is very small at this time. It may be that only the X channel is over-saturated, or both X and Z channels are over-saturated. It can be denoted as $\{\hat{X}\}$ and $\{\hat{X},\hat{Z}\}$. When the outlier is in area II, the y coordinate is extremely small. At this time, the saturation is $\{\hat{Y}\}$ or $\{\hat{Y},\hat{Z}\}$. When the outlier is in area III, the saturation situation is $\{\hat{X},\hat{Y}\}$ or $\{\hat{X},\hat{Y},\hat{Z}\}$. Finally, since the saturation of the Z channel will increase the x and y coordinates, the outlier saturation corresponding to area IV is $\{\hat{Z}\}$.

It can be found that the Z channel may be saturated from regions I to III. Therefore, we follow the sequence of I+III, II, IV to detect outliers and compensate X, Y, Z channel, respectively. Each compensation value is the ADC maximum output value $2^n - 1$. Over-saturation outliers can then be restored to the normal CSK symbol points.

B. DEMODULATION

Since CSK signals are all normalized maximum power, although the channel attenuation for different colors of light is different, power compensation is performed. Nevertheless, the signal power received at the receiver remains at a constant level. Moreover,

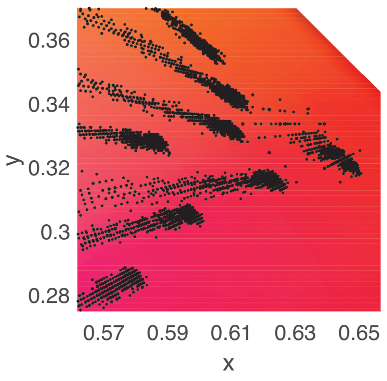


Figure 2.18: The received CSK signal cluster.

because the screen is closer to the receiver, the detected screen light intensity is much higher than the intensity of the modulated light signal after channel (screen + air) attenuation, far exceeding it by one to two orders of magnitude. Therefore, the threshold can be set according to the average power of the modulated light at the transmitter, and all received signal points can be divided into the modulated signal + screen OFF signal points, and the modulated signal + screen ON signal points.

For the modulated signals + screen OFF signal points, we calculate ΔE [71] to measure the difference between the colors of two sampling points in the CIE 1931 diagram. ΔE is the Euclidean distance between two colors in the x, y-plane of the CIE 1931 diagram. We select the smallest ΔE to match the color of a symbol to carefully picked colors via the SpiderWeb algorithm (cf. Section 2.2.3). The received sampling point is denoted as $r_i \triangleq (x_i^r, y_i^r)$, and the SWebCSK constellation coordinates for reference are denoted as $u_j \in \mathcal{M} \triangleq \{1, \cdots, M\}$. The optimal minimum distance decoder can be expressed as $u_i^* = \arg\min_{u_j \in \mathcal{M}} \left\| r_i - u_j \right\|_2$, where u_i^* is the decoded symbol.

The modulated signals + screen ON signal points are usually clustered together, as shown in Figure 2.18. If we simply recover the required signal through slope detection, when the signal point interfered by the screen is very close to the screen color signal point, the coordinates will swing slightly affected by the noise. It will have a greater impact on slope detection. Therefore, we again divide the slope detection into two parts: for the first part when the transmitted signal is most affected by the screen color and when it is located in a smaller modulation triangle compressed after the superimposition of the screen signal, we still use the minimum Euclidean distance detection; for the other part that is outside the smaller modulation triangle, we use the minimum slope detection. The optimal slope decoder can be expressed as

$$s_i^* = \begin{cases} \underset{\overline{s_j} \in \overline{\mathcal{M}}}{\arg\min} \left\| r_i - \overline{u}_j \right\|_2, & d_i \le d_{\text{th}}, \\ \underset{s_j \in \mathcal{M}}{\arg\min} \left(\left| \frac{y_i^r - y_S}{|x_i^r - x_S|} - \frac{y_j - y_S}{|x_j - x_S|} \right| + \left| \frac{|y_i^r - y_S|}{x_i^r - x_S} - \frac{|y_j - y_S|}{x_j - x_S} \right| \right), & d_i > d_{\text{th}}, \end{cases} \tag{2.8}$$

where $\overline{s_j} \in \overline{\mathcal{M}}$ is the set of CSK symbol points that are fully interfered by the screen, d_{th} is

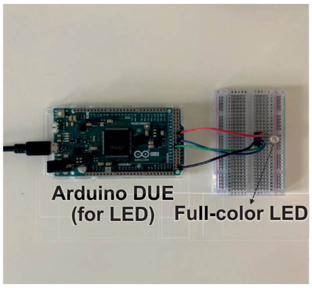


Figure 2.19: Our implemented SpiderWeb transmitter.

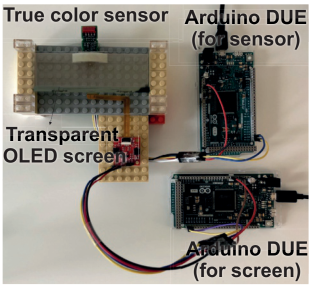


Figure 2.20: Our implemented SpiderWeb receiver.

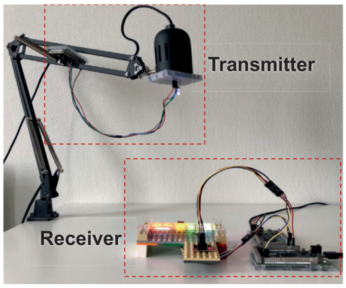


Figure 2.21: The basic experimental setup.

the predefined threshold, and $d_i \triangleq \|r_i - u_M\|_2$ is the distance between received sampling point and screen color point in the CIE 1931 diagram.

C. Preamble detection

To decode data from the received signal, we first need to detect the preamble of the frame. In VLC, the preamble is designed for frame detection and synchronization [72]. The preamble pattern is usually fixed with a total of $2K$ alternating ONs and OFFs (i.e., $2K$ ON-OFF patterns). For example, the OpenVLC platform adopts 24 alternating ONs and OFFs as the preamble [58], [73]. However, due to potential screen interference, it is impossible to confirm whether the pilots that send all CSK symbols can be successfully decoded and demodulated, and confirm the correct starting position of the data symbol. Thus, we send several CSK symbols of the same color on the screen as pilots.

2.4. Implementation

In this section, we present the implementation of VLC transmitter, screen module, and our under-screen receiver.

Transmitter. A snapshot of our implemented SpiderWeb transmitter is given in Figure 2.19. We use an off-the-shelf full-color LED (Kingbright WP154 [74]) in the transmitter front-end. The peak wavelengths of R, G, B light of this LED are 640 nm, 515 nm, and 461 nm, respectively. The measured color coordinates (x, y) associated with this RGB LED are $(x_R = 0.6544, y_R = 0.3205)$, $(x_G = 0.1776, y_G = 0.7188)$, and $(x_B = 0.1307, y_B = 0.0711)$. An Arduino DUE, a low-cost and open-hardware embedded development platform with an 84 MHz processor, is used as the processing unit at the transmitter. We leverage three independent PWM ports of the Arduino DUE to control the RGB LED separately for generating various colors. The frequency of the PWM signals is set to 1 MHz.

OLED screen. We use the transparent OLED screen Sparkfun LCD-15079 [75] in our prototype.[6] The colors supported at different pixels of this screen are fixed, and a large part of the screen is just transparent and cannot display any colors. From the rest of this screen, we identify three parts that can display different colors and are large enough to cover our color sensor when the sensor is placed under that part: red $(0.6544, 0.3205)$, yellow-green $(0.4400, 0.5000)$, and white $(0.3350, 0.3722)$. The OLED screen is controlled

[6]Although its name has "LCD", it is indeed an OLED screen as specified in [75].

Table 2.1: Comparison under different screen colors ($M = 4$)

Screen color	$d_{\min}$		$\theta_{\min}$ (rad)	
	4-CSK	4-SWebCSK	4-CSK	4-SWebCSK
Red	0.3372	0.2330	0.4942	0.5702
Yellow-green	0.3372	0.1763	0	1.5708
White	0.3372	0.2538	1.9896	2.0944

Table 2.2: Comparison under different screen colors ($M = 8$)

Screen color	$d_{\min}$		$\theta_{\min}$ (rad)	
	8-CSK	8-SWebCSK	8-CSK	8-SWebCSK
Red	0.1934	0.1209	0	0.1901
Yellow-green	0.1934	0.1220	0	0.5236
White	0.1934	0.1416	0	0.8976

by a dedicated Arduino DUE. The control range of the screen brightness is from 0 to 31, where 0 corresponds to no light on the screen, 31 corresponds to a duty cycle of 100% for the PWM control signal of the screen, and the screen brightness is the highest.

Receiver. The prototype of the VLC receiver is shown Figure 2.20. We use the true color sensor AMS AS73211 [69], controlled by an Arduino DUE, to detect the color-based VLC signals. The response of AS73211 conforms to the CIE 1931 standard. Its maximum internal clock frequency is 8.2 MHz, and the maximum supported I2C clock frequency is 400 kHz. Under the maximum clock frequency, the adjustable gain range of AS73211 is $\{1\times, 4\times, 16\times, 64\times, 256\times\}$, and the sampling integration time is from 125 μs to 2048 ms. In our implementation, we set the gain to $64x$ and the sampling integration time to 125 μs.[7] The sensed data is transmitted through a serial port to a laptop for processing, which has an Intel i7 CPU and 32 GB memory. The proposed signal processing and the slope-based demodulation are implemented in Matlab.

2.5. PERFORMANCE EVALUATION

In this section, we present the evaluation of SpiderWeb in different scenarios. We use Bit Error Rate (BER) and normalized data rate as the metrics. The default experimental setup is shown in Figure 2.21.

2.5.1. SWEBCSK VS. CSK

Before we present the evaluations of SpiderWeb, we first compare the performance of SWebCSK with that of traditional CSK.

[7]A very short integration time of the color sensor will lead to the detection failure of the modulated signals; however, with a very long integration time, the color sensor will be over-saturated. Similar results apply to the gain setting of the color sensor. In our implementation, we use the minimum sampling integration time, together with the color sensor gain of $64x$, to obtain the highest data sampling rate and under these settings, we can still detect the modulated visible light signals after screen attenuation.

Table 2.3: 8-SWebCSK constellation points (under red screen)

Gray code	Constellation coordinates	$(\overline{\mathrm{PWM}_R}, \overline{\mathrm{PWM}_G}, \overline{\mathrm{PWM}_B})$
000	(0.3819, 0.5481)	(43, 60, 0)
001	(0.1886, 0.5785)	(4, 42, 9)
011	(0.4013, 0.4032)	(49 ,35, 16)
010	(0.2141, 0.3762)	(12, 42, 22)
110	(0.4773, 0.3091)	(65, 15, 18)
111	(0.3108, 0.2312)	(33, 16, 27)
101	(0.1736, 0.0915)	(8, 0, 42)
100	(0.6544, 0.3205)	(100, 0, 0)

Constellation. We compare our SWebCSK constellation design with the traditional CSK design specified in the IEEE 802.15.7 standard [16]. The results for $M = 4$ and $M = 8$ are shown in Table 2.1 and Table 2.2, respectively. The minimum distance among the 4-CSK and 8-CSK constellation points in the IEEE standard is always larger than the minimum distance under our 4-SWebCSK and 8-SWebCSK. However, in the angle space formed by the screen color point on the CIE 1931 diagram, the angle between different constellation points of 8-CSK to screen color point (i.e., the rays; cf. Section 2.2.2) overlap. For 4-CSK, when the screen color is yellow-green, the angles from the two constellation points to the screen also overlap. In our 4-SWebCSK and 8-SWebCSK, we not only optimize the minimum angle between different rays, but also adapt the optimal constellations based on the detected screen color. Thus, the minimum angles (important when the screen is in the ON state) among 4-SWebCSK and 8-SWebCSK constellations are higher than those of 4-CSK and 8-CSK. Additionally, we present other simple heuristic algorithms to search for CSK constellation points, as shown in Figure 2.22. For instance, in the case of bisection θ, the constellation points are selected sequentially according to the left-hand or right-hand spiral. However, the minimum distance (important when the screen is in the OFF state) between these points remains lower than in the 8-CSK design we developed. We also list the constellation points of 8-SWebCSK, as shown in Table 2.3. It also gives the corresponding PWM duty cycles to achieve these constellation points.

BER. In this experiment, three brightness of the screen are considered: 0%, 50%, and 100%. we place the transmitter and the receiver at a distance of 10 cm. For the traditional CSK modulated signals, we consider two demodulation methods: 1) use the minimum Euclidean distance to distinguish different symbols; 2) use the proposed slope-based demodulation (CSK+SD) method, cf. Section 2.3.2. For our SWebCSK modulated signals, we use our slope-based demodulation. For the modulation level $M = 4$ and $M = 8$, the BER results are shown in Figure 2.23 and Figure 2.24, respectively. We can observe that the BER of 4-CSK and 8-CSK detected by the minimum Euclidean distance are close to 0.5 when the screen is lit up. Under this BER, data transmission is not possible. Using our slope-based detection, 4-CSK reduces the BER to 0 when the screen colors are red and white; when the screen color is yellow-green, under which the angles from the two CSK symbol points to the screen color point overlap, the BER is still below 10^{-1}, showing

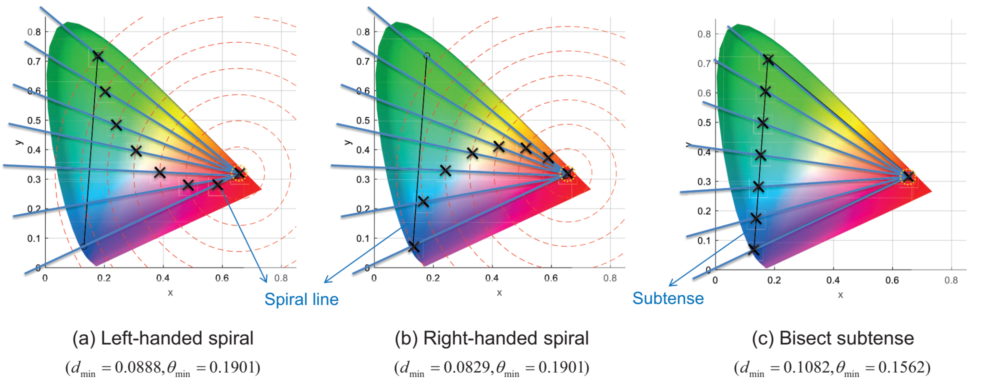

Spiral line
Subtense
(a) Left-handed spiral
($d_{\min} = 0.0888, \theta_{\min} = 0.1901$)
(b) Right-handed spiral
($d_{\min} = 0.0829, \theta_{\min} = 0.1901$)
(c) Bisect subtense
($d_{\min} = 0.1082, \theta_{\min} = 0.1562$)

Figure 2.22: Heuristic suboptimal CSK solutions: example design of 8-CSK under red screen. All performance metrics are lower than our 8-SWebCSK solution ($d_{\min} = 0.1209$, $\theta_{\min} = 0.1901$).

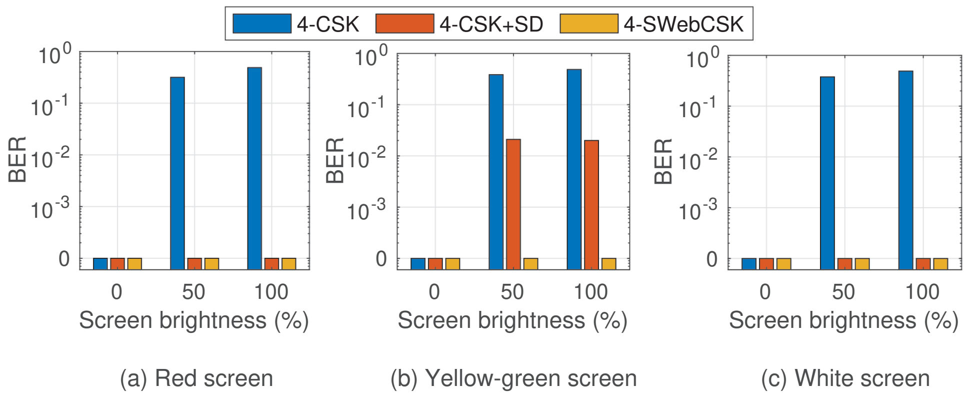


Figure 2.23: BER under different screen colors when $M = 4$.

the advantage of our proposed slope-based demodulation. In 8-CSK, the angles from several constellation points to the screen color point overlap with each other. Thus the BER falls below 10^{-1}. For our designed 4-SWebCSK and 8-SWebCSK, the BER is lower than 10^{-3} regardless of the screen color and the screen brightness, demonstrating that SWebCSK is a key enabler for through-screen VLC.

Normalized data rate. We also evaluate the date rate obtained from traditional CSK and our SWebCSK. For M-CSK, a CSK symbol represents $\log_2 M$ bits. With a larger M, the transmitter can send more bits in a unit time slot. However, a larger M often leads to a higher BER. The achievable data rate of M-CSK and our M-SWebCSK changes with different colors and brightness. Figure 2.25 shows the normalized data rate averaged over different scenarios (i.e., different screen color: red, yellow-green, and white; and 3 different screen brightness: 0%, 50%, and 100%). We can observe that system data rate is increased by at least 300%. This means SWebCSK can carry more data than the traditional CSK under dynamic screen color and brightness changes. We can observe

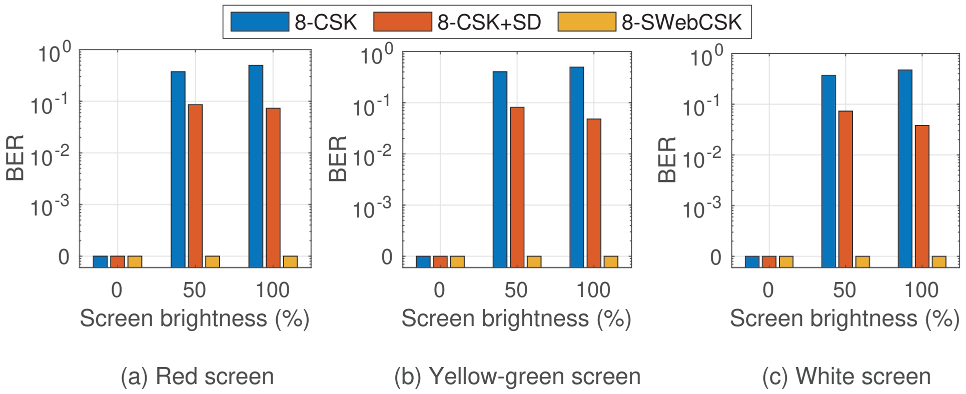


Figure 2.24: BER under different screen colors when $M = 8$.

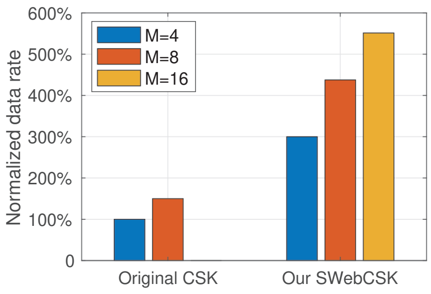


Figure 2.25: Data rate vs. schemes.

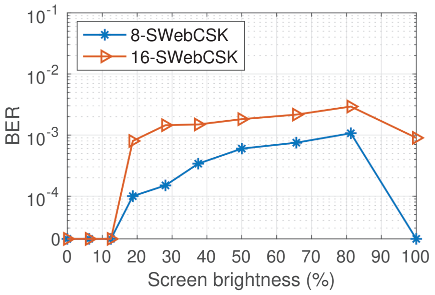


Figure 2.26: BER results vs. brightness.

that when the modulation level $M = 4$, our SWebCSK can improve the data rate by 200%. When $M = 16$, the data rate of CSK is almost zero because the BER is too high; while 16-SWebCSK achieves 5× data rate when normalized to that of the 4-CSK. On average, our proposed SWebCSK could achieve 3.4× data rate when compared to the traditional CSK.

2.5.2. SPIDERWEB EVALUATION

Now we evaluate the system performance of SpiderWeb.

Impact of screen brightness. The brightness of the transparent OLED screen determines how much the transmitted signal is interfered by the screen. Usually, the brighter the screen, the more transmitted symbols are interfered with. When the screen brightness is 100%, all the signals are interfered by the screen. The experimental BER under different screen brightness is shown in Figure 2.26. Here, 16-SWebCSK is evaluated under the screen color of yellow-green (cf. Figure 2.31(a)). Overall, the BERs of 8-SWebCSK and 16-SWebCSK are below 10^{-3} and below 10^{-2}, respectively. We observe that with both 8-SWebCSK and 16-SWebCSK, in general, the BER gradually increases with a brighter screen. An exception is that when the screen is fully lit, the BER of 8-SWebCSK is 0 and the BER of 16-SWebCSK reduces to 10^{-3}. The main reason is that the screen signal and the transmitted VLC signals are not synchronized. Since the screen brightness is not

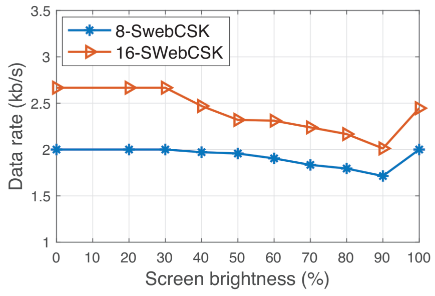


Figure 2.27: Data rate vs. brightness.

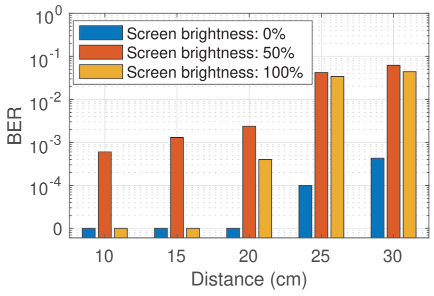


Figure 2.28: BER results vs. distance.

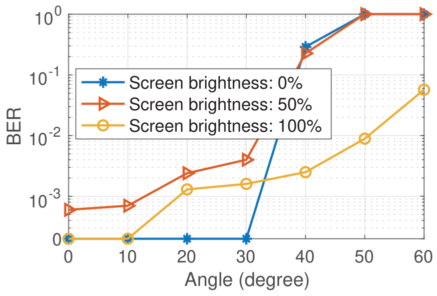


Figure 2.29: BER results vs. angle.

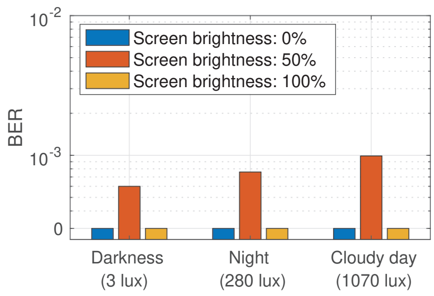


Figure 2.30: BER results vs. ambient light.

constant, with energy detection, we cannot perfectly distinguish all the VLC signals that have been interfered by the screen from those that have not, which brings inevitable misjudgments when the screen is ON but not fully lit. When the screen is fully lit, it is easy to get a priori information that all the received VLC signals are interfered by the screen. Thus, the slope-based SWebCSK detection can still achieve a significantly low BER. We also measure the achievable data rate of SpiderWeb under different screen brightness, as shown in Figure 2.27. For 8-SWebCSK, the data rate is between 1.7 kb/s and 2 kb/s; for 16-SWebCSK, the data rate changes from 2 kb/s and 2.6 kb/s. The current bottleneck is the slow response of the low-cost color sensor. The achievable data rate could be significantly improved with advanced color sensors.

Impact of distance. Next, we evaluate the system performance when the transmitter is placed at different distances from the screen/ receiver. Here, we use the maximum gain 256× of the color sensor to maximize the communication distance. Because the channel attenuation of RGB light is different, the corresponding CSK constellation point correction is performed at the transmitter for each distance. The BER results are shown in Figure 2.28. With an OFF screen, the BER is lower than 10^{-3} at 30 cm. When the screen brightness reaches 50% and 100%, the BER at 20 cm is still lower than 10^{-2}.

Impact of angle. We also evaluate the performance of SpiderWeb when the receiver is placed in different directions to the transmitter. We fix the distance between the transmitter and screen to 10 cm and vary the relative angle between them. The results are shown in Figure 2.29. We can see that the BER performance decreases severely when the angle between the transmitter and the receiver exceeds 30°. On the other hand, the BER performance is still better when the screen is fully lit, although the angle exceeds 30°. At an angle of 50°, the BER is close to 10^{-2}. The reason behind this could be that when the angle is large, the intensity of the CSK signals detected by the under-screen sensor is minimal and does not exceed its output threshold, resulting in a sharp increase in the BER when the screen brightness is OFF. However, if the screen is fully lit, the intensity of the CSK signal detected by the sensor exceeds its output threshold after adding the screen light, thereby reporting the detection value. In other words, when the angle is large, the screen light could help the signal detection in SpiderWeb.

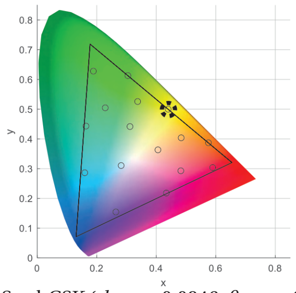
Impact of the ambient light conditions. Lastly, we study the impact of different ambient light conditions on the system performance. Three different ambient light conditions are considered: (1) Darkness (average light intensity: 3 lux); (2) Night with indoor illumination (average light intensity: 280 lux); (3) Cloudy day (average light intensity: 1070 lux). The impact of the ambient light level on the BER is shown in Figure 2.30. When the screen is either OFF or fully bright, the BER is 0 under all three conditions. With a screen brightness of 50%, the BER in all three conditions is still lower than 10^{-3}. Note that due to our over-saturation recovery method (cf. Section 2.3.2), even in the presence of ambient light, our system can still achieve a BER of 0 when the screen is fully bright.

2.6. Related Work

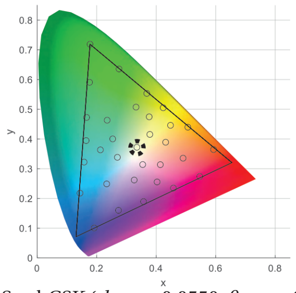
CSK modulation in VLC. It has been adopted in the IEEE 802.15.7 VLC standard [16] and studied in [30], [68], [76]. The authors in [68] optimize the CSK constellations to improve the system performance. In [76], the authors design a light-to-frequency converter as a CSK receiver to avoid using ADC, leading to a low-cost VLC system. CSK is also used in LED-to-camera communication systems to increase the data rate [30]. These studies mainly optimize and utilize CSK-based VLC with different receivers. In our SpiderWeb, we consider a more challenging and very promising scenario where the screen at VLC receivers could "block" and interfere with the communication with visible light. We observe the color-pulling effect and propose the SWebCSK modulation scheme to solve it. Accordingly, we can enable through-screen VLC with an extremely low bit error rate, which cannot be achieved by the above state-of-the-art studies.

Sensing with under-screen line-of-sight sensors. The trend toward *narrow-bezel* and *no-bezel* smartphone designs depicts future devices. Therefore, the line-of-sight sensors (e.g., front camera) are being tested under the screen for sensing. The authors in [11], [15] study how to recover the images captured by an under-screen camera. In their study, a camera is placed under a transparent and a pentile OLED screen, respectively, to obtain a degradation image dataset [15]. Two types of neural networks are proposed to tackle image degradation [11]. Compared with these studies, we focus on communication with ambient light/color sensors placed under an OLED screen, and we have identified the screen's color-pulling effect on color-based VLC signals. We further design SWebCSK modulation to eliminate this effect and thus enable through-screen VLC.

Screen-enabled VLC. Some studies use screens as multi-pixel transmitters to increase the data rate [77], [78]. They exploit standard screens and monitors to achieve screen-



(a) 16-SwebCSK ($d_{\text{min}} = 0.0846$, $\theta_{\text{min}} = 0.2244$).



(b) 32-SwebCSK ($d_{\text{min}} = 0.0550$, $\theta_{\text{min}} = 0.2027$).

Figure 2.31: Examples for high-order SwebCSK: (a) 16-SwebCSK under yellow-green screen and (b) 32-SwebCSK under white screen.

to-camera communication. There are also studies using the Liquid Crystal (LC) of LCD screens as reflectors for VLC. PassiveVLC uses a single LC cell to modulate visible light [79]. ChromaLux employs multi-layer LC cells to increase the communication distance and speed [80]. The authors in [81] leverage LCs to enable selective reception of light beams from multiple transmitters. In our system, we use a transparent OLED screen instead of an LCD screen. The OLED screen brings opportunities (being transparent) but also challenges (being an interferer) for the realization of through-screen VLC.

2.7. CONCLUSION

We have studied how to enable through-screen VLC with sensors under a transparent OLED screen that displays different colors at different pixels. We manually move the screen to various locations to change its displayed color to investigate the screen's impact on the light signal. We discovered a color-pulling effect caused by the screen on CSK signals: transmitted CSK symbols are pulled to the detected screen color. We further designed the SWebCSK modulation and proposed a slope-based demodulation to eliminate the color-pulling effect. Our proposed solutions for through-screen VLC could work under such a dynamic screen because the refresh rate of a screen is usually about 60 Hz, which is much lower than the symbol rate (several kHz) in our current implementation. The screen refresh interval can be interpreted as channel coherence time. The transmitter could know the current screen color in two ways: 1) the transmitter detects the screen color directly; 2) the receiver detects the screen color and sends the information to the transmitter through an uplink channel. Based on the color of the screen, the transmitter and the receiver can establish new SWebCSK constellations for the data transmission. We prototyped a system and validated the feasibility of achieving through-screen VLC. We envision that our work could stimulate follow-up studies on through-screen visible light communication and sensing.

Future work. Currently, the communication distance of SpiderWeb is limited due to the low power of the RGB LED used, which has a maximum power dissipation of only 120 mW. Utilizing higher-power RGB LEDs designed for indoor illumination can enhance communication distance. The different luminous efficiencies of the R/G/B channels also contribute to low light emission power. To prevent flickering, the power envelope of CSK symbols is fixed, with the transmission power determined by the highest optical output power of the color with the lowest luminous efficiency, such as the red color. Meanwhile, the highest data rate we can achieve in the current SpiderWeb is about 2.6 kb/s, though it is about 3.4× compared to that of the traditional CSK. The main bottleneck is the low speed of the color sensor. The minimum sampling time of the color sensor is 125 μs (we have configured the color sensor to its fastest speed mode). We also use oversampling to remove the sampling points affected by inter-symbol interference. When multiple transmitters send data concurrently, the received CSK symbol at the single-pixel color sensor may be a superposition of symbols from multiple transmitters, causing interference and degrading system performance without specific transceiver design enhancements. These factors collectively limit the SpiderWeb data rate. One potential solution is that when the screen color is within the modulation triangle area (with a modulation angle of 360°) we could further increase the modulation level, such as to 32-SWebCSK, as shown in Figure 2.31(b), to further improve the data rate of the system. However, these methods cannot fully overcome the rate constraints imposed by the hardware. Once hardware limitations are addressed, the aforementioned high-order modulation techniques can be applied to the new through-screen VLC system to further boost data rates through advanced signal processing.

In the next chapter, we aim to exploit another critical under-screen multi-pixel camera on commercial full-screen smartphones with fully dynamic OLED screens to overcome interference from multiple transmitters and enable advanced signal processing methods and a faster sampling rate to break the rate bottleneck at the receiver. Under-screen cameras are being adopted in today's advanced full-screen smartphones, such as ZTE AXON30 and Xiaomi MIX4. Additionally, a single transmitter can employ multiple RGB LEDs to transmit more data. When combined with an under-screen multi-pixel camera at the receiver, we can leverage light spatiality to potentially increase the system data rate. We will also use LEDs with higher visible light output power, particularly in the red color, to further extend the communication range to several meters.

3 WHEN VLC MEETS UNDER-SCREEN CAMERA

While the first-generation VLC systems mainly employ LEDs as transmitters and single-pixel sensors as receivers, recently, cameras widely available on commodity smartphones are utilized as the receiver for VLC. So in terms of hardware, both VLC transmitter (i.e., LED) and receiver (i.e., smartphone) are already pervasive in our everyday lives without incurring any extra hardware burden. Most smartphones nowadays have two cameras, i.e., a front camera and a back camera. Between them, the front camera is considered a better option to serve as the VLC receiver because during our routine use of the smartphone, the front camera faces the LEDs in our surrounding environment most of the time. The communication modality of LED-to-front-camera can be a good supplement to existing wireless technologies. Several LED-to-front-camera applications, such as museum narration, have been proposed in recent works [26], [28], [30], [82]–[84]. Another exciting application is to transfer sensitive data that needs to be confined within an area of interest, such as a room.

LED-to-front-camera communication. As a subset of VLC, LED-to-front-camera communication employs LEDs as the light emitter. To achieve a fast data rate, the rolling shutter effect is normally used in state-of-the-art solutions. A significantly faster communication can be realized compared to solutions without using the rolling shutter effect [83]. In an image sensor generally used in smartphone cameras, exposure is performed per scan-line based on the rolling shutter effect [25], [85]. When a smartphone camera photographs the rapidly blinking LED at a time interval of several milliseconds, white stripes appear when the LED is ON, and black stripes appear when the LED is OFF. In addition, the lengths of the frame time and the gap time are reflected in the pixel width of the stripe pattern; hence, temporal blink information can be converted to spatial information as stripes in the frame. Meanwhile, color information can be further leveraged to improve the throughput by several times using RGB LEDs [30].

However, the trend towards full-screen devices poses challenges for LED-to-front-camera communication, as the integration of the front camera under the screen, result-

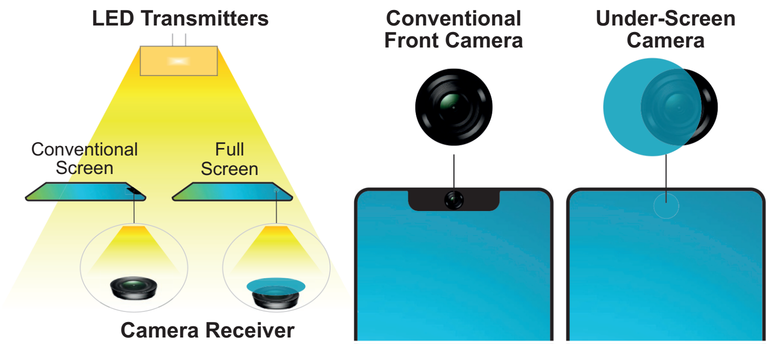


Figure 3.1: Illustration of the proposed through-screen VLC with Under-Screen Cameras (USC). USC makes VLC on full-screen devices challenging.

ing in Under-Screen Cameras (USC), critically impacts VLC performance. As shown in Figure 3.1, although the under-screen camera design has little impact on the photo and video quality due to a dedicated transparent screen layer, it severely degrades the performance of VLC: the transmission range is reduced from a few meters to merely 0.04 m, and the data rate is decreased by more than 90%.

Challenges and Contributions. We dig deep to identify the causes of this performance degradation. The first cause is signal attenuation brought in by the transparent screen. The second cause, which is more severe, is the interference induced by the transparent screen. This transparent screen region is still part of the screen, and therefore, the dynamically changing screen content can severely affect the VLC performance. In this chapter, we ask the following research question:

Can we utilize the camera under the screen for VLC without incurring a performance degradation?

To make VLC work with under-screen cameras, we must deal with the two causes of performance degradation. Note that the first issue is also the main challenge for the original function of camera, i.e., image and video taking. Therefore, a tremendous amount of effort has been devoted to addressing this issue. The state-of-the-art solution uses a small piece of transparent screen on top of the camera. However, even with the state-of-the-art solutions, we still observe a significant decrease in signal amplitude [10], [86].

To address this issue, we still use Color Shift Keying (CSK) modulation instead of conventional ON-OFF modulation. Unlike ON-OFF modulation, color-based CSK modulation utilizes the color positions, which are much less sensitive to light intensity. Also, by utilizing color modulation, more bits can be encoded to achieve a higher data rate.

To address the second issue, we need to tackle multiple challenges. As cameras on commodity devices such as smartphones usually have a low image rate, i.e., less than 200 images per second, to achieve a high data rate, a unique property of camera imaging, i.e., the rolling shutter effect [25], [85] is utilized in LED-to-front-camera communication.

The basic idea is that within a single image, all the pixels are not captured simultaneously but are recorded row by row at different timestamps. We can therefore zoom in to extract the color of light at the granularity of the row level. When there is no screen on top of the camera, the light source can be easily identified in the image as the light source area's brightness is much higher than other areas. When the camera is under the screen, a lot of other areas also have high brightness as the screen itself is also a light source. For accurate demodulation, we need to make sure we are extracting the color information of LED light but not other areas on the image. Therefore, the first challenge is

Challenge 1: *How to accurately identify the light source, i.e., the LED in the captured images in the presence of strong interference?*

To address this challenge, we propose a color compensation method leveraging an observation of the color composition of the transparent screen: when CSK-modulated images are converted into grayscale for detecting the VLC transmitter, the grayscale values of the red and blue components are lower than that of the green component. We thus increase the brightness of the blue and the red components at the captured image, which helps us detect the VLC transmitter area in the captured image of an under-screen camera. We further apply the vertical averaging scheme to distribute the same brightness to all three colors. This vertical average scheme also strengthens the low-frequency data while weakening the high-frequency noise. The VLC transmitter has the same color for pixels in a row, while the other area has random colors for pixels in a row. After this step, the VLC transmitter area is further highlighted. We can then apply a median filter to remove the background and keep the identified VLC transmitter for further processing.

Although the LED (transmitter) area is detected, the image still contains interference from the transparent screen. The second challenge is

Challenge 2: *How to remove the interference caused by the transparent screen?*

We propose two novel steps to address the interference. The first step is based on the key fact that the transparent screen (the piece of glass) on top of the camera has a unique design which is very different from ordinary screens. While an ordinary screen has a much larger pixel area for each pixel point (cf. Figure 3.3), the transparent screen has a much smaller pixel area in order to let more light go through the screen. Because of this design, the image taken with an under-screen camera has two groups of area, i.e., screen-pixel-interference area and non-screen-pixel-interference area. We thus identify the non-screen-pixel-interference area for more accurate demodulation. The good news is that the transparent screen design for each manufacturer is fixed and we just need to obtain the pixel layout design once for each model of smartphone.

The second step is to identify the optimal pixel point for demodulation. This is because the under-screen camera exposure duration can be long enough to have two different colors from LED. Note that the camera does not show both colors (one color in the first part and the other color in the second part) but just shows a mixed color for the row.[1] We, therefore, need to identify the optimal pixel point without color mixing for demodulation. To solve this problem, we propose a vertical scanning method. The basic idea is to model the color information as a transition graph. While mixed colors are on

[1]The camera can be understood as a sensor that integrates incoming light signals within a specified exposure time window, resulting in a mixed color.

the transition links, the pure colors are the transition nodes. We can thus identify those nodes with an obvious direction change for the best demodulation performance.

After we apply the methods described above to identify those non-screen-pixel-interference areas, we still find some interference leaked from those pixel-interference areas and this leakage does affect the performance. The leakage can cause the color to vary. So the third challenge we encounter is

Challenge 3: *How to address the leakage interference to improve the performance?*

To solve this challenge, we propose a series of methods. First, we observe that the color change due to the leakage is not random. For example, if a red color is interfered by blue color leakage, then the red color will be shifted to a different color between red and blue. We can therefore use a line to connect these two colors and the color only changes along the line. This property enables us to recover the original color we transmitted even if the color is shifted. A more challenging scenario is that a random screen color leakage can be on the line connecting two colors used for communication. In this case, the two colors change following the same line, confusing the method above to recover the original color. Interestingly, we find that in this case, while the colors' absolute values (locations) are not accurate, the relative information between the two colors is still reserved, i.e., if one shifted color is further away from the interfering color, the original color is also further away from the interfering color. We can thus utilize the relative information for demodulation.

With the design components, we successfully implement a proof-of-concept system for the proposed through-screen VLC system with under-screen camera using off-the-shelf smartphones. Our system is able to achieve low BER and long transmission range with under-screen camera for the first time. Below we summarize our contributions.

- We propose the concept of VLC with under-screen camera for emerging full-screen devices. We analyze the unique spatiotemporal characteristics of the rolling shutter effect on under-screen camera. We design a pixel-sweeping algorithm based on the spatiotemporal characteristics of the image/frame captured by the under-screen camera and identify the sampling points with the least amount of interference for communication. We further propose a novel slope-boosting demodulation method to deal with color shifts brought by leakage interference.

- We build a proof-of-concept testbed and thoroughly evaluate the system performance in different scenarios. The results show that our system can achieve a maximum throughput of 54.43 kb/s with four LED transmitters and one under-screen camera receiver. Compared to state-of-the-art solutions, our methods can reduce the BER by two orders of magnitude on average, and improve the throughput by 59× (914 b/s vs. 54.43 kb/s). The transmission range is extended from 0.04 m to 4.2 m, with an improvement of roughly 100×.

3.1. ROLLING-SHUTTER EFFECT OF CAMERA RECEIVER

The rolling-shutter effect, which is a fundamental property of CMOS image sensors commonly used in smartphone cameras, allows exposure to be performed on a per-row basis [25], [85]. A rolling-shutter camera controls the exposure row-by-row (or column-by-

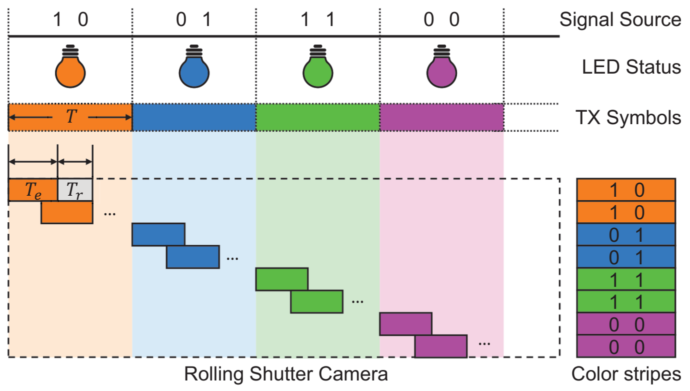


Figure 3.2: Rolling-shutter effect principle.

column), as shown in Figure 3.2. The first row is exposed for T_e and the second row starts being exposed after T_r. T_r is a measure of the time required to capture and transfer image data from the camera to the processor, which is fixed for each camera. Note that T_r is usually smaller than T_e and therefore to fully utilize the channel, the adjacent exposures are overlapped. In the traditional rolling-shutter (without the effect of the screen), all pixels in a row contain the *same modulated information*. The strip width (w) is defined as the number of pixel rows occupied by the same color strip in a captured image. Therefore, the width of the stripe is determined by $w = T/T_r$, where T donates the symbol period. Thus, the higher the frequency of the transmitted symbol, the narrower the stripes (i.e., the less number of rows) with the same color will be in the image.

3.2. SYSTEM OVERVIEW

3.2.1. ARCHITECTURE

Transmitter. We adopt the standard Color-Shift Keying (CSK) modulation [16] at the receiver. The CSK constellation symbols are chosen inside a *modulation triangle* that is formed within the standard color space (cf. Section 2.1.4 in Chapter 2). To transmit CSK symbols, the transmitter modulates the signal by varying the output power combinations of the three channels R/G/B of the LED.

Receiver. We consider a **full-screen** device (e.g., a smartphone) as the receiver. It does not have any notches/bezels on its screen to host line-of-sight sensors such as a camera and ambient light sensor. Instead, full-screen devices introduce a special region on the screen, i.e., a small **transparent screen region**. This region is composed of transparent substrates and cathodes and is usually placed at the top section of the screen. The camera is placed directly under the transparent screen region. The transparent screen region *is still part of the screen and can be lit up to display various contents.* Compared to other screen regions, the transparent screen region utilizes a transparent electrode material, and the pixel layout is also redesigned. A common transparent screen region's structure used by commercial transparent OLED screens is shown in Figure 3.3. The transparent screen region can be further divided into two parts:

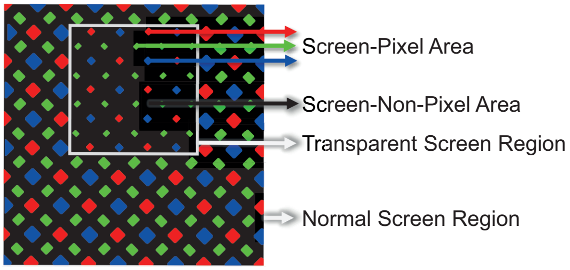


Figure 3.3: Screen-pixel layout.

- **Screen-pixel area**: It hosts specially designed R/G/B pixels that can be lit up to display various contents. A large screen-pixel area benefits the displaying function of transparent screen region but sacrifices the imaging quality of under-screen camera as well as the performance of through-screen VLC.
- **Screen-non-pixel area**: It allows visible light to pass through – but still at the cost of strong attenuation – to reach the under-screen camera. A large screen-non-pixel area brings more light to the under-screen camera, which benefits the under-screen camera's imaging quality and the performance of through-screen VLC, but reduces the transparent screen region's displaying performance.

The industrial effort has been devoted to optimizing the pixel layout to balance the performance of screen displaying and imaging quality [6], [87]. The density and size of the screen-pixels in the transparent screen region have been carefully adjusted to increase the proportion of screen-non-pixel area in the transparent screen region, as shown in Figure 3.3. However, no matter how the transparent screen region is designed, it still requires the screen-pixel area to display content, and the screen-non-pixel area cannot be made fully transparent. It thus still has large impacts on through-screen VLC. Next, we present the major impacts.

3.2.2. TRANSPARENT SCREEN'S IMPACT ON USC COMMUNICATION

As a passive 'blocker' (The corresponding solution will be presented in Section 3.3.1). The transparent screen region blocks a significant portion of the incident light. The amount of light that can travel through the transparent screen region is only 2.9% [10]. This discourages us from adopting intensity-based modulations widely used in state-of-the-art VLC [59], [61], [79]. We leverage CSK modulation to mitigate this impact since the demodulation of color-based CSK signals does not depend much on signal strength. Instead, it relies on mapping the CSK signals to the CIE color space. However, the signal strength attenuation still makes it difficult to detect the position of the VLC transmitter (i.e., region of interest) on the image captured by the under-screen camera.

As an active interferer. The pixels in the transparent screen region are light sources, inducing diffractive blur and color shifts on under-screen camera's captured images. This interference can significantly affect the whole image and lead to a low Signal-to-Interference-plus-Noise Ratio (SINR) of the captured image.

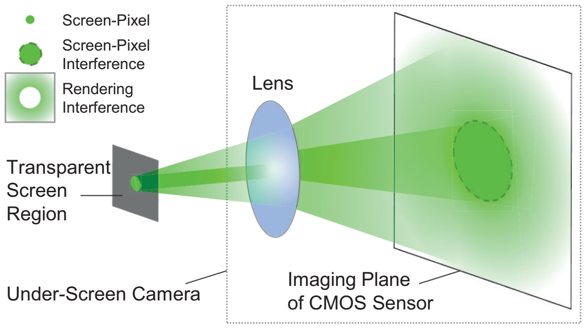

Screen-Pixel
Screen-Pixel Interference
Rendering Interference
Transparent Screen Region
Lens
Under-Screen Camera
Imaging Plane of CMOS Sensor

Figure 3.4: Illustration of screen-pixel and rendering interferences, using green pixel as an example.

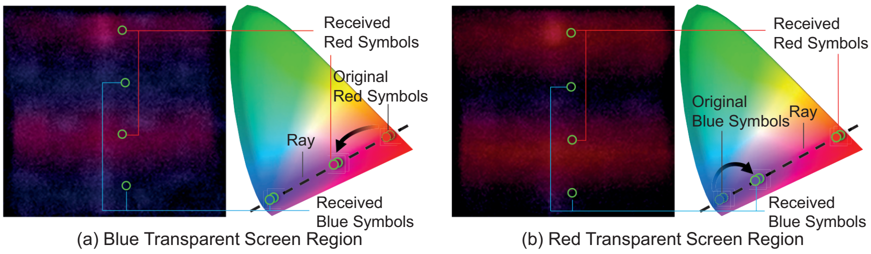


Figure 3.5: Color-pulling effect caused by the rendering interference.

- *Screen-pixel interference (The corresponding solutions will be presented in Sections 3.3.3 and 3.3.4).* An under-screen camera, like traditional cameras, is composed of a lens and a CMOS image sensor. Figure 3.4 is a micrograph illustration of a *Green* pixel in the transparent screen region and its mapping to the imaging plane of the under-screen camera. This illustration is based on the principle of pinhole imaging [17]. The *dashed circle* illustrates the interference area on the camera's imaging plane, where the interference comes from the transparent screen region's Green pixel because this area is exactly under the Green pixel. We term it as *screen-pixel interference*, which maintains the original screen-pixel shape but is larger in size on the imaging plane. Note that the area of this screen-pixel interference caused by the transparent screen region's R/G/B pixels is different, which depends on the color and size of the pixels in the transparent screen region.

- *Rendering interference.* On the camera's imaging plane, there is also leakage from the transparent screen region's illuminating pixel, causing *rendering interference* as denoted by the *solid square* in Figure 3.4. This rendering interference is induced by the diffusion effect of the screen-pixel. Compared to the screen-pixel interference, the strength of the rendering interference is weaker, but the interference area on the camera's imaging plane is larger.

This rendering interference leads to the so-called *color-pulling effect [88]* (cf. Chapter 2) on the received CSK symbols. When the screen is lit up, the original color of a

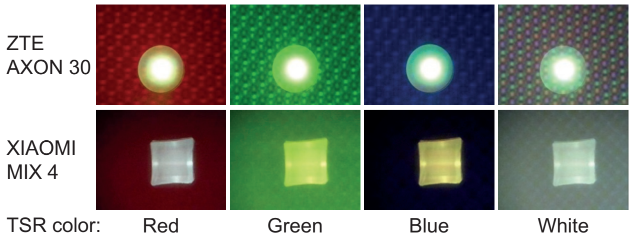


Figure 3.6: Screen diversity: The interference of different pixel layouts on images captured by the under-screen camera. (Transparent Screen Region: TSR)

CSK symbol will be pulled closer to the corresponding transparent screen region's pixel color, as shown in Figure 3.5. To give an example, let us consider when we send the red and blue symbols alternately, and the transparent screen region is displaying blue. We pick some sampling points from the area (on the camera's imaging plane) that is affected by rendering interference, and show their color coordinates in the CIE 1931 diagram in Figure 3.5(a). We can see the location of blue light is not affected when the screen color is blue. However, the received red coordinates are pulled closer to the blue screen from the original red coordinate, leading to decoding errors. When sending the same symbols, the example of a transparent screen region showing red is shown in Figure 3.5(b). *We will present our solution to this problem in Section 3.4.*

Screen diversity. The transparent screen region designs on different smartphones are not the same, bringing different levels of impact on through-screen VLC. Due to patent protection, different manufacturers adopt various pixel designs and arrangements in the transparent screen region (cf. Chapter 1). These unique transparent screen region designs bring different interference to the images captured by under-screen camera, as shown in Figure 3.6. *In Section 3.3.3 (Steps 1 & 2), we will present a solution that can work with different transparent screen region designs.*

Pipeline of our solution. Next, we present how to address these challenges. First, we propose a method to extract the Region of Interest from the captured images (Section 3.3.1). Then, we design a pixel-sweeping algorithm, including a horizontal scanning module to address screen diversity issues and remove screen-pixel interference (Section 3.3.3) and a vertical scanning module to deal with CSK's inter-symbol interference (Section 3.3.4). Finally, we propose a slope-boosting algorithm to demodulate the CSK symbols by removing the color shift caused by rendering interference (Section 3.4).

3.3. Pixel-sweeping Algorithm

3.3.1. Region of Interest (RoI) Extraction

To decode data, the prerequisite is to detect the transmitter's centroid and size – the *Region of Interest (RoI)* that contains transmitted information – on captured images. Existing methods such as grayscale threshold-based mask methods [28] and computer vision-empowered methods [82] do not work well on detecting the RoI from the CSK-

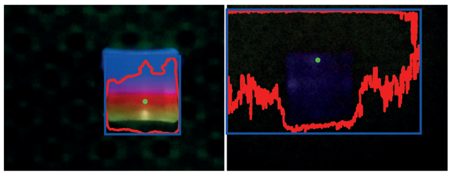


Figure 3.7: RoI is submerged in the displayed light.

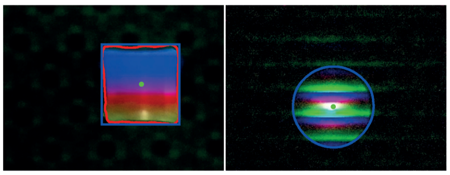


Figure 3.8: An example of shape filter working.

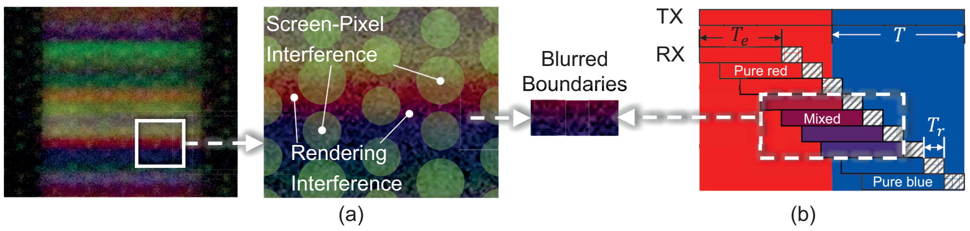


Figure 3.9: Rolling-shutter effect of under-screen camera.

modulated images captured by under-screen camera. The reasons are as follows: 1) When CSK-modulated images are converted into grayscale, the grayscale values of each color component are different. The grayscale values of the red and blue components are lower than that of the green component. Thus, the mask method does not work in through-screen VLC; 2) The transparent screen region weakens the overall intensity of the RoI and brings extra interference. As a result, the RoI's contour has more jagged edges and can be easily submerged in displayed screen light, as shown in Figure 3.7.

To extract the RoI in through-screen VLC, we propose a new method, as shown in Figure 3.10. First, we compensate the blue and red components. Then, we apply a vertical blur filter on the whole image. The kernel size of the blur filter is at least one stripe width to average the grayscale between all stripes. To cancel transparent screen region's impact, we add median and binary OTSU filters [89] to filter out the pixels of the transparent screen region, allowing us to obtain a smooth contour of the RoI. Finally, we use a shape filter with a circle area of $4\pi N_r/C_r^2$, where N_r is the number of pixels in the area and C_r is the length of the area boundary. When the output of the shape filter is greater than 0.8, the shape of the detected RoI is regarded as a circle. The illustrating results after applying shape filters as shown in Figure 3.8.

3.3.2. ROLLING SHUTTER ON USC: SPATIOTEMPORAL FEATURES

We continue to detect the positions of CSK symbols within the RoI. We first analyze the spatiotemporal characteristics of the rolling-shutter effect on under-screen camera.

Spatial characteristics. Compared to traditional rolling-shutter effect (cf. Section 3.1), for image captured by the under-screen camera, which is affected by transparent screen region, the pixels in different parts of a row are exposed to *different levels of interference* (screen-pixel interference and rendering interference). Figure 3.9(a) shows that the pixels are subject to different R/G/B interference with different intensities. The main reasons are i) the interference intensities of the screen-pixel interference and the rendering interference are different; ii) the pixel layouts are different due to screen diversity. This spatial characteristic of the sampling points of under-screen camera makes it impossible to arbitrarily pick a pixel from each row to decode the transmitted information.

Temporal characteristics. Due to the uncontrollable read-out duration (T_r), it is difficult to control the exposure duration (T_e) of under-screen camera to synchronize with the transmitted CSK symbols (T). As illustrated in Figure 3.9(b), this results in inter-symbol interference on both sides of a stripe, caused by the mix of the current stripe and the previous/next stripe. Also, due to different R/G/B screen-pixel layout designs of transparent screen region in different smartphones, the total levels of interference induced by screen light on pixels of each row are different. Thus, for the rolling shutter on under-screen camera, *pixels are subject to double interference from the transparent screen region's displayed screen light and also from the inter-symbol interference.* Also, this temporal characteristic makes the symbol width and the symbol boundary vague for detection. Therefore, it is infeasible to average the color values of the entire stripe to decode the transmitted CSK symbol represented by a stripe.

Based on the above spatiotemporal features, we design a pixel-sweeping scheme for demodulation (cf. Figure 3.12). Next, we present the details.

3.3.3. HORIZONTAL SCAN IN THE SPATIAL DOMAIN

To decode information from the stripes, we should first identify a stripe where to sample a pixel that has less transparent screen region interference in RoI. Compared to pixels in the screen-pixel area, pixels in the screen-non-pixel area are interfered by the same color (e.g., white when all R/G/B pixels are lit up) with less intensity. Note that the light transmittance from the screen-pixel area is lower than that from the screen-non-pixel area [14]. The characteristics of the transparent screen region require higher through-screen light intensity to facilitate under-screen camera photographing, and thus the ratio of the screen-non-pixel area is increased in the transparent screen region. Thus, we can always find a sampling point that is not in the screen-pixel area when we scan each row in the RoI, as detailed below:

Step 1: Obtain the placements of transparent screen region's R/G/B pixels. We first need to know the layout of the pixels in the transparent screen region. The layouts from different manufacturers mainly differ in the following aspects: *location*, *shape*, and *size*. We use a **one-time calibration** to identify the transparent screen region's R/G/B pixels to handle the transparent screen region diversity challenge: a) For *location*, since a screen must light up R/G/B screen-pixels together to display white color, we first let the transparent screen region display white color and configure a large exposure period on the under-screen camera before communication. Since different color screen-pixels have different luminous efficiency, we use a multi-level OTSU threshold algorithm [90] to obtain the exact position of each screen-pixel for each R/G/B channel. b) For *shape*, we find that the final interference formed by screen-pixels of different shapes is still approximately circular due to the isotropy of screen-pixel emission. Therefore, we unify the interference area of each screen-pixel as their minimum circumcircle, as shown in Figure 3.11, and identify the circumcircle location using Hough transform method [91].

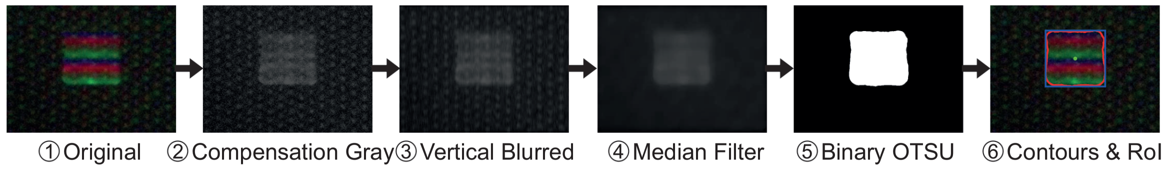


Figure 3.10: Improved RoI extraction mechanism.

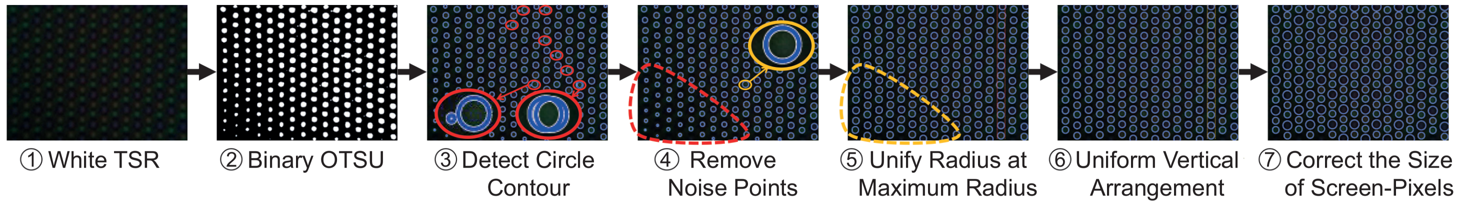

① White TSR
② Binary OTSU
③ Detect Circle Contour
④ Remove Noise Points
⑤ Unify Radius at Maximum Radius
⑥ Uniform Vertical Arrangement
⑦ Correct the Size of Screen-Pixels

Figure 3.11: Obtain placement of R/G/B screen-pixels mask. (Transparent Screen Region: TSR)

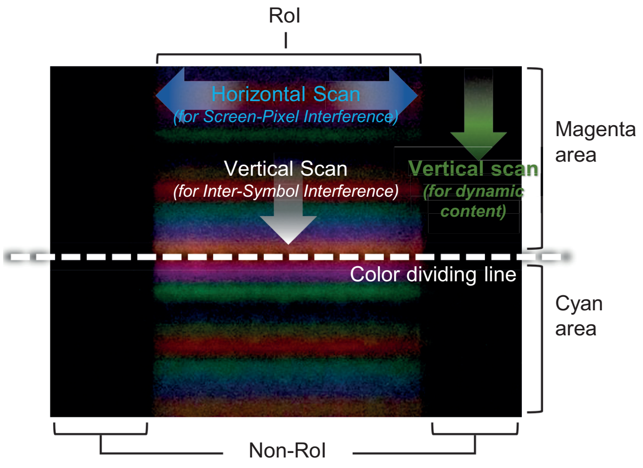


Figure 3.12: Pixel-sweeping workflow.

There are some smaller noise points after Hough transformation. We filter out the noise points by setting the mean of all contour radii as the radius threshold and the mean distance between contour centroids as the distance threshold. c) For *size*, we light up the R/G/B screen-pixels in turn, and apply procedure ②-⑥ in Figure 3.11 to each single R/G/B channel. Finally, we take the maximum radius of the detected contour as the radius of the R/G/B screen-pixel. Note that since the relative positions of the transparent screen region and the under-screen camera are fixed, each smartphone only needs to perform this calibration procedure once.

Step 2: Establish the correct screen-pixel mask. After obtaining all the R/G/B screen-pixels, we need to know the current color of the transparent screen region to establish the correct screen-pixel mask *since the screen's interference on decoding depends on screen's displayed colors*. To achieve this, we filter the pixels in the non-RoI by setting a threshold based on the average brightness level of these pixels, as shown in Figure 3.12. This step is necessary to remove the effect of black image noise. After converting the filtered pixels from the RGB color space to the standard CIE 1931 XYZ space, we obtain the screen-non-pixel area color point (x_s, y_s) corresponding to the current color of transparent screen region. We then compare it with the screen color space used in smartphones [71]. For example, if the screen color point is on the vertex of the screen color triangle, we only need to set the screen mask with one set of R/G/B pixels. The sampling points under the screen-pixels which are not lit up, can still benefit demodulation. Step 2 presents us with more space to pick up better sampling points in Horizontal scanning.

Step 3: Pick up pixels outside the R/G/B screen-pixel area. We set the middlemost pixel of the first row of RoI as the origin to start our scan. We scan each pixel in the row from left to right until one pixel located outside of the screen-pixel area is identified and this pixel is selected as the output sampling point. Then we jump to the closest pixel in the next row and repeat the above scanning process. Note that when the scanning process reaches the end of a row, we resume from the beginning of the same row.

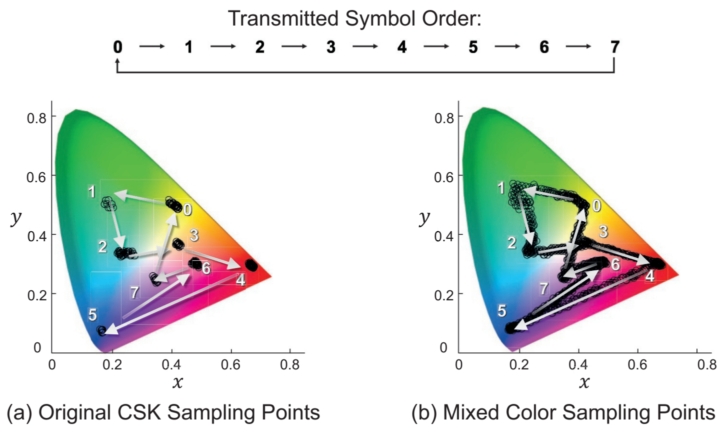


Figure 3.13: Color mixing.

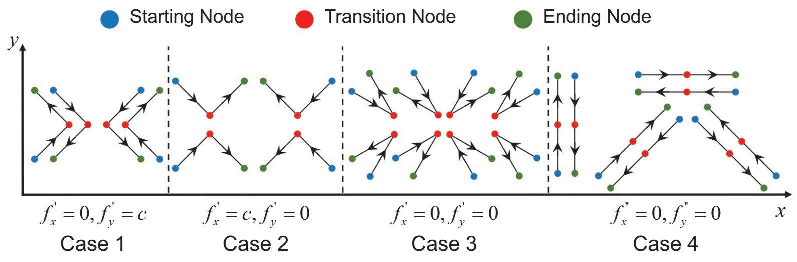


Figure 3.14: Transition graph in CIE 1931 diagram (node: pure colors; edge: mixed colors).

3.3.4. VERTICAL SCAN IN TIME DOMAIN

According to the Rolling-Shutter effect, the maximum number of rows in one stripe without inter-symbol interference can be expressed as $n = \lceil (T - T_e)/T_r \rceil$, where T is the period of the transmitted symbol. When $\lceil (T-T_e)/T_r \rceil = 3$, there are at most[2] three pure CSK symbols (i.e., red color) in a stripe without inter-symbol interference (cf. Figure 3.9(b)). However, we can also see that, even after the previous horizontal scan step, there are still CSK symbols with mixed colors (cf. Figure 3.9(b)). We therefore need to pick those CSK symbols with a pure color[3] for best communication performance.

We propose a vertical scan method to identify the pure-color CSK symbols. Specifically, we model the color information as a transition graph in Figure 3.13. The pure colors are denoted as nodes while mixed colors are denoted as lines connecting the nodes. Note that without inter-symbol interference, we can obtain the color transition graph in Figure 3.13(a) with all the nodes (i.e., pure colors) clearly separated. When there is more inter-symbol interference, we have more mixed colors and the color transition graph is shown in Figure 3.13(b). Our objective is to clearly identify the nodes (i.e., pure-color

[2]When the initial exposure timestamp of the first red stripe is not synchronized with the symbol's starting point, the number of pure red CSK symbols is less than 3.

[3]Here pure color means one of the colors used in our CSK modulation/demodulation without a mixture of other colors.

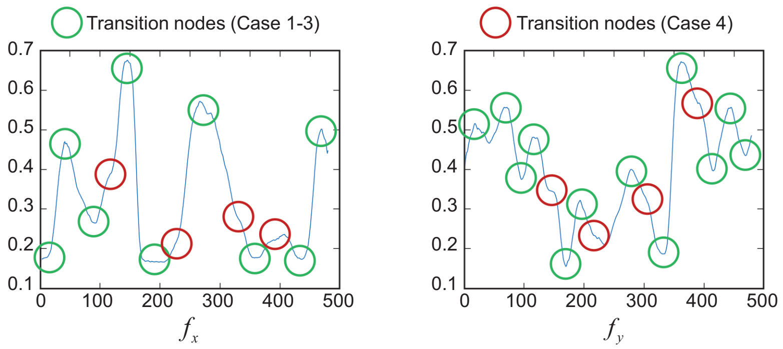


Figure 3.15: Illustrating for transition nodes found in vertical scan process.

CSK symbols) even in the presence of inter-symbol interference. We show all the possible color transition cases in Figure 3.14. We can see that while there exist clear direction changes at the nodes in the first three cases, there is no obvious direction change on the color transition graph in Case 4.

As shown in Figure 3.14, we denote the color change in the x coordinate as f_x, and the change in the y coordinate as f_y. For the first three cases, we have either $f'_x = 0$ or (and) $f'_y = 0$ at the nodes. In the challenging Case 4, the transition nodes have no clear direction change. However, due to the continuity of the color changes, we have $f''_x = 0$ and $f''_y = 0$. We thus leverage these properties to identify the transition nodes on the graph. We present an identification result using data collected in our experiment. As shown in Figure 3.15, we can accurately identify those transition nodes, circling those for the first three cases in green and those for Case 4 in red.

3.4. Demodulation

After pixel-sweeping, the original two-dimensional sampling points in the RoI are significantly reduced to a $N \times 1$ sampling points vector, where N is the number of transmitted symbols in the corresponding frame. Even if we have identified the pure-color sampling points for each stripe, the sampling points are still affected by the *color-pulling effect* of *rendering interference* (cf. Section 3.2.2).

3.4.1. Classification of CSK Symbols

The received CSK symbols (colors) can be divided into two groups, i.e., 1) one CSK symbol close to the screen color and 2) other CSK symbols. For the CSK symbol in 1), we can directly use distance detection to establish symbol mapping because they are minimally interfered by the screen. For the CSK symbols in 2), we define the line connecting the currently displayed screen color point and a CSK constellation point as a *ray*. Except for the screen color point, if there are multiple CSK symbols on a ray, we define this ray as *overlapping ray*, and these CSK symbols are regarded as overlapping symbols on this ray. If there is no overlapping, we can apply slope detection for demodulation. The slope here refers to the slope of the ray connecting the currently displayed screen color point

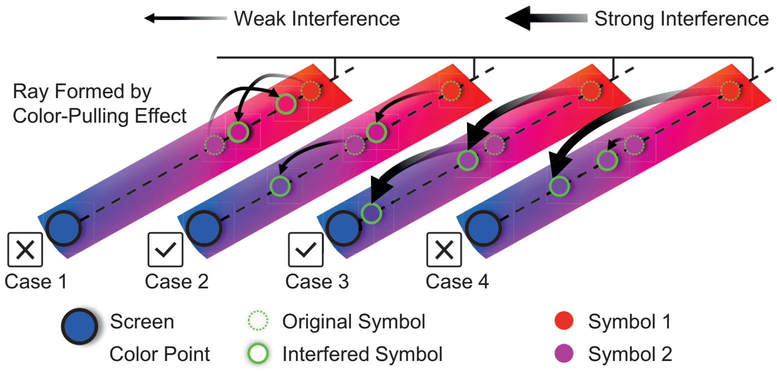


Figure 3.16: The overlapping cases.

and a CSK constellation point in the CIE 1931 diagram. We denote the screen color point and CSK symbol as $u_s = (x_s, y_s)$ and $u_m = (x_m, y_m)$, respectively. The slope of the ray is then calculated as $(y_m - y_s)/(x_m - x_s)$.

3.4.2. Analysis of Overlapping Cases

We take the case of two overlapping CSK symbols (red symbol 1 and magenta symbol 2) on a ray as an example to illustrate the concept, and the rest cases–when there are multiple overlapping CSK symbols on a ray–can be deduced by analogy. The final positions of the two CSK symbols affected by the color-pulling effect of the screen light are analyzed, as shown in Figure 3.16. We can first rule out Case 1, as the color-pulling effect can only cause unidirectional movement. Cases 2, 3, and 4, however, are all possible. In Case 2, both symbols 1 and 2 experience minor interference. In Case 3, symbols 1 and 2 are both subjected to significant screen interference. Case 4 arises when symbol 1 experiences substantial interference while symbol 2 encounters minor interference. By applying the pixel-sweeping algorithm, we filter out non-uniform and strong interference within the screen-pixel area. Consequently, all $N \times 1$ sampling points undergo uniform rendering interference, which is either the same strong or weak, depending on the screen's brightness. As a result, only Cases 2 and 3 hold true. This implies that *the CSK symbols on the same ray can always be accurately distinguished under rendering interference*. We leverage the relative positions of these CSK symbols rather than the absolute positions for precise demodulation. For instance, we can map sampling points closer to the screen color point in Cases 2 and 3 to symbol 2, and the more distant points to symbol 1.

3.4.3. Slope-boosting Demodulation

Motivated by reserved relative position information on the overlapping ray, we design a slope-boosting scheme for demodulation. With the slope of the ray, we can correctly demodulate the CSK symbols on non-overlapping rays and distinguish those CSK symbols on overlapping rays. The details are as follows:

Step 1: Color classifier. First, we need to obtain the transparent screen region's color coordinates in the captured frames. Accurate demodulation of the overlapping ray using the relative information between CSK symbols can only be achieved with rendering interference of the same color. To acquire the interference color coordinates in a frame, we refer to step 2) in Section 3.3.3 to determine the color of the transparent screen region. When the screen content changes at a high frequency, the color transitions within the transparent screen region occur frequently. Consequently, each frame captured by the under-screen camera may contain multiple color interferences.[4] However, owing to the screen's row-by-row scan mechanism [92], we can still detect color changes within the same frame/image and differentiate the sampling points in an image affected by different transparent screen region colors.

Step 2: Slope classifier. We denote all preset M-CSK symbols as $u_m = (x_m, y_m)$, $m \in \mathcal{M} \triangleq \{1, \cdots, M\}$. If the distance from the screen color point to the preset CSK symbol is less than the threshold d_{th}, we determine that this preset CSK symbol is a screen color symbol, i.e., $u_s = (x_s, y_s)$. The threshold $d_{th} = 0.05$ is determined empirically to minimize the decoding errors. All the rest of the preset CSK symbols are applied with the slope classifier defined as

$$\frac{\left|\frac{y_m - y_s}{|x_m - x_s|} - \frac{y_n - y_s}{|x_n - x_s|}\right| + \left|\frac{|y_m - y_s|}{x_m - x_s} - \frac{|y_n - y_s|}{x_n - x_s}\right|}{\left|\frac{y_m - y_s}{x_m - x_s}\right| + \left|\frac{y_n - y_s}{x_n - x_s}\right|} < s_{th}, \tag{3.1}$$

where the slope is normalized to prevent slight jitter from causing severe slope fluctuation due to different distances from each preset CSK symbol to the screen color origin, and $s_{th} = 1$ is determined empirically as the slope change tolerance. For the output of the slope classifier, we have two types of sets $\mathscr{A}$ and $\mathscr{B}$. The non-overlapping CSK symbols are stored in $\mathscr{A}_i, i = \{1, \cdots, I\}$, where $I \leq M$ is the number of non-overlapping rays, and the number of CSK symbols in $\mathscr{A}_i$ is denoted as $|\mathscr{A}_i| = 1$. Also, $\mathscr{B}_j, j = \{1, \cdots, J\}$ stores all the CSK symbols with at least one overlapping case, where $J \leq M$ is the number of the overlapping rays, and $|\mathscr{B}_j| \geq 2$ represent the number of overlapping CSK symbols on the ray j. When the slope difference between symbol m and symbol n is smaller than s_{th} in Equation (3.1), the rays formed by the color-pulling effect of symbol m and symbol n overlap at ray j. Thus, m and n are put into the set $\mathscr{B}_j$, i.e., $\mathscr{B}_j = \mathscr{B}_j \cup \{m, n\}$, otherwise, $\mathscr{A}_i = \mathscr{A}_i \cup m$ and $\mathscr{A}_{i+1} = \mathscr{A}_{i+1} \cup n$.

Step 3: Distance decoder. We denote the input sampling point sequence as $r_l = (x_l^r, y_l^r), l = \{1, \cdots, L\}$, where L is the length of sequence. The distance decoder is expressed as $\|r_l - u_s\|^2 < d_{th}$. Thus, all transmitted symbols with the least color-pulling effect are detected. Note that when the color of the transparent screen region is detected as black, all sample point sequences are input into the distance decoder for demodulation. The optimal minimum distance decoder is calculated as

$$r_l^* = \arg\min_{m \in \mathcal{M}} \|r_l - u_m\|_2, \tag{3.2}$$

where r_l^* is the decoded symbol.

Step 4: Slope-boosting decoder. We do slope-boosting detection on all remaining sample points except those processed by the distance decoder. The optimal slope decoder can

[4]Currently, due to limitations in the output frame rate of under-screen camera and refresh rate of the screen, there are at most two different transparent screen region colors in a single captured frame.

Algorithm 2 Slope-boosting demodulation

```
Input: $\mathscr{B}_j$: preset overlapping CSK symbols on ray $j$, $\mathscr{C}_j$: overlapping sampling points
    on ray $j$, $K$: the number of clusters, $D_{m,n} \triangleq \|p_m - p_n\|_2$ ;
Output: $\boldsymbol{r}^*$;
 1: Sort $\mathscr{C}$ in descending order of $|\mathscr{C}|$;
 2: for $j = 1$ to $J$ do
 3:   $K = |\mathscr{B}_j|$;
 4:   while $\mathscr{B}! = \emptyset$ do
 5:     $\boldsymbol{e}_i, i = \{1, \cdots, K\}$ with centroid $p_i$ are obtained by K-means clustering [93] on the
        set $\mathscr{C}_j$;
 6:     if $D_{m,n} > d_{th}, \forall m, n \in \{1, \cdots, K\}$ then
 7:       $r_l^* \in \boldsymbol{e}_i \leftarrow \mathscr{B}_j$, sort $\boldsymbol{e}_i$ and $\mathscr{B}_j$ in ascending order of $D_{m,n}$ ;
 8:     else if $D_{m,n} < d_{th}$ then
 9:       $K = K - 1$;
10:     end if
11:   end while
12: end for
```

be expressed as

$$r_l^* = \arg\min_{m \in \mathscr{M}} \frac{\left| \frac{y_l^r - y_s}{\left|x_l^r - x_s\right|} - \frac{y_m - y_s}{|x_m - x_s|} \right| + \left| \frac{\left|y_l^r - y_s\right|}{x_l^r - x_s} - \frac{|y_m - y_s|}{x_m - x_s} \right|}{\left| \frac{y_l^r - y_s}{x_l^r - x_s} \right| + \left| \frac{y_m - y_s}{x_m - x_s} \right|}. \tag{3.3}$$

If $r_l^* \in \mathscr{A}_i$, then it means that the ray where the demodulated symbol is located does not overlap, and the sampling point is directly mapped to r_l^*. Otherwise, if $r_l^* \in \mathscr{B}_j$, we put this sample point into set $\mathscr{C}_j$, i.e., $\mathscr{C}_j = \mathscr{C}_j \cup r_l^*$. The sampling points in set $\mathscr{C}_j$ correspond to those sample points falling on the overlap ray represented by set $\mathscr{B}_j$.

The slope-boosting procedure is shown in Algorithm 2. We can group CSK symbols interfered by the same screen color to increase the number of symbols used for clustering in Step 1. A larger number of sample points can improve the decoding accuracy.

3.5. IMPLEMENTATION

In this section, we present the implementation details of our practical through-screen VLC with under-screen camera system.

LED transmitter. A snapshot of our implemented transmitter is presented in Figure 3.17(a). It includes a full-color LED chip, a lampshade, an LED driver board, and a control unit. The full-color LED chip has a maximum power consumption of 5 Watts and a maximum brightness of 450 lumens, powered by a 24V DC voltage. We use a long-strip lampshade with a dimension of 35 cm × 160 cm in our experiment. The shape and length of the lampshades are similar to the commonly seen fluorescent luminaires in offices. An Arduino DUE–a low-cost embedded platform–is used as the control unit at the LED transmitter. We leverage three independent PWM ports of the Arduino DUE to control the full-color

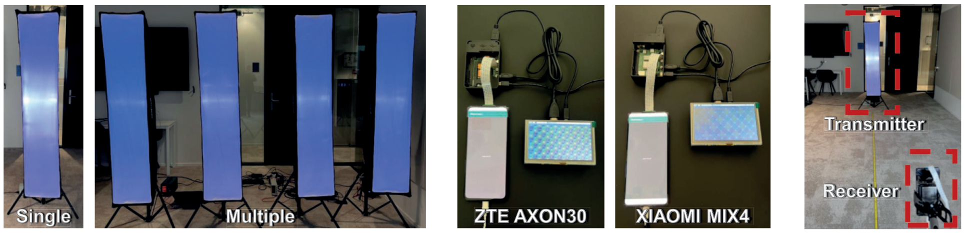


Figure 3.17: The evaluation setup.

LED separately for generating different CSK symbols. The LED driver has three transistors (ON MOSFET 20N06L) and three transistor drivers (TC4420). The PWM ports of the Arduino DUE trigger the transistors to modulate the full-color LED.

Receiver. The receivers we implement for the through-screen VLC are presented in Figure 3.17(b). We use two different models of full-screen smartphones that are equipped with under-screen cameras, i.e., ZTE AXON30 and Xiaomi MIX4, both are available in the market since late 2021. AXON30 employs an AMOLED screen that can reach a maximum brightness of 475 cd/m^2. MIX4 also has an AMOLED screen but with a maximum brightness of 903 cd/m^2. The screen refresh rate of the two smartphones both supports 60 Hz and 120 Hz, and uses the color gamut of DCI-P3. For both smartphones, the pixel density of the transparent screen region is 400 pixels per inch, but *the shape, size, and layout of pixels in the transparent screen regions are different*. We develop an APP to control the displayed color in the transparent screen region to evaluate the performance of our system under different screen colors. The APP can run with Android 4.4.2 and above. For the under-screen camera, a 16-megapixel front camera and a 20-mega-pixel front camera are originally used in the AXON30 and the MIX4, respectively. The recorded frames are transferred to a laptop for further processing.

3.6. Performance Evaluation

We evaluate the performance of our proposed through-screen VLC with under-screen cameras in various scenarios. We use Bit Error Rate (BER), data rate, and transmission range as the metrics for performance evaluation.

3.6.1. Preliminary Evaluation

The default experiment setup is shown in Figure 3.17(c). We place the transmitter and receiver with a distance of 1 m in between. At the transmitter, we adopt the traditional CSK design specified in the IEEE 802.15.7 standard [16]. The frequency of the PWM signals is set to 1 MHz to support 5 kHz CSK symbol rate. At the receiver, the screens are set to 100% full brightness. We turn off automatic exposure/gain control of the under-screen camera to capture frames with a size of 640×480 at the highest frame rate of 200 Frames Per Second (FPS). We set the ISO (sensitivity of the camera) to the highest value of 1600. We fix the under-screen camera exposure time to 1/5400 s, slightly lower than the period of the transmitted CSK symbols, which is 200 μs. The read-out duration of

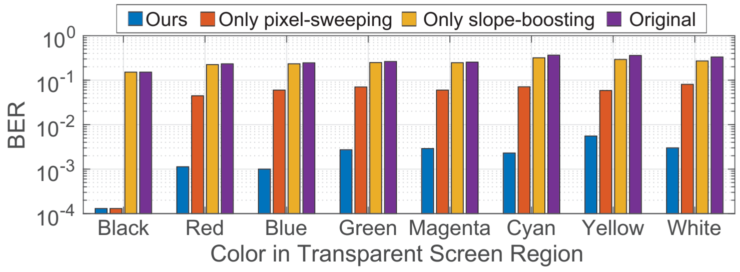


Figure 3.18: BER results vs. screen color (8-CSK).

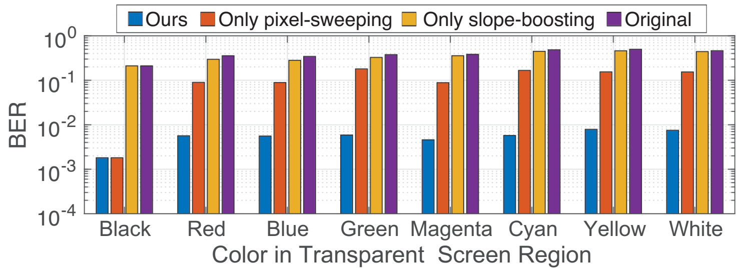


Figure 3.19: BER results vs. screen color (16-CSK).

the under-screen camera is estimated to be around 10 μs, using the method presented in [26]. We mainly use AXON30 for evaluation (except in Section 3.6.4). To evaluate the performance of our system, we compare the performance of four solutions:

- *Ours:* As presented in the previous sections, our solution leverages i) the proposed pixel-sweeping algorithm to obtain the pixels that represent each color stripe (cf. Section 3.3), and ii) the proposed slope-boosting detection algorithm to decode the CSK symbols (cf. Section 3.4).

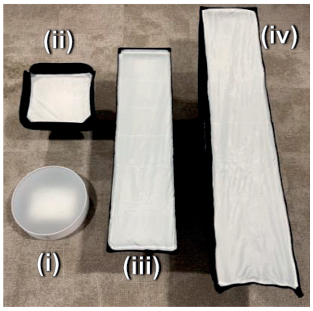
- *Original:* It uses a conventional method to sample pixels at a fixed width instead of our pixel-sweeping algorithm; It also only adopts conventional minimum Euclidean distance detection for CSK symbol decoding [30], [94].

- *Only pixel-sweeping:* It only adopts our pixel-sweeping algorithm to obtain the pixels; For decoding the CSK symbols, it uses the conventional minimum Euclidean distance detection method [30], [94] instead of our proposed slope-boosting detection algorithm.

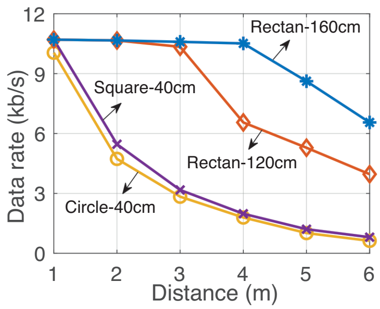
- *Only slope-boosting:* It samples the pixels at a fixed width instead of our proposed pixel-sweeping algorithm; It adopts our proposed slope-boosting detection algorithm to decode the CSK symbols.

BER results versus screen color. We evaluate the system performance under eight different colors of the screen: *black,*[5] *red, blue, green, magenta, cyan, yellow, white.* We

[5]This is achieved when the smartphone's screen is not lit up.



(a) Different lampshade setup



(b) Data rate vs. distance

Figure 3.20: Impact of size and shape of the light source.

use our designed APP to switch the smartphone's screen color among these eight colors. The BER results under 8-CSK are shown in Figure 3.18. We first observe that without our proposed algorithms, i.e., with the 'Original' solution, the BER under all the eight screen colors goes beyond 10^{-1}, which fails the through-screen VLC link. However, with our proposed solution ('Ours'), the BER can be as low as 10^{-4} when the smartphone's screen is not lit up ('black'), and around 10^{-3} when the smartphone displays other seven colors. An interesting observation is that among the three colors: red, green and blue, the BER under the green screen is the highest. This is because in OLED, the number of green pixels is much larger than that of red and blue pixels. Therefore, more interference is generated when the smartphone displays green color. We observe that the BER under 'Only pixel-sweeping' is lower than the BER under 'Only slope-boosting'. Also for both of them, the BERs under different screen colors (except 'black') are always higher than 10^{-2}. We further evaluate the system performance when 16-CSK is used at the transmitter. The results are presented in Figure 3.19. We can observe that under all eight different screening colors, our proposed solution can still achieve a BER below 10^{-2}.

In the rest of the evaluations in this section, unless otherwise specified, we use 8-CSK at the transmitter and consider the most challenging scenario, i.e., the smartphone displays white color (all the R/G/B pixels of the screen are lit up).

Impact of transmitter size/shape. We next investigate the impact of transmitter shape and size on the achieved data rate. As shown in Figure 3.20(a), we use four lampshades with distinct shapes and sizes at the transmitter: (i) a circular lampshade with a diameter of 40 cm (Circle-40cm); (ii) a square lampshade with a side length of 40 cm (Square-40cm); (iii) a rectangular lampshade with a size of 30 × 120 cm (Rectan-120cm); and (iv) a rectangular lampshade with a size of 35 × 160 cm (Rectan-160cm). Figure 3.20(b) shows the impact of these lampshade shapes and sizes on the achieved system data rate. Rectangular lampshades of 160 cm and 120 cm exhibit a data rate drop when the communication distance exceeds 4 m and 3 m, respectively, while square and circular lampshades show a data rate drop beyond 1 m. This is due to the rolling-shutter effect, as the system's data rate is significantly influenced by the number of pixel rows of the light source in the image captured by the under-screen camera. The smaller the captured

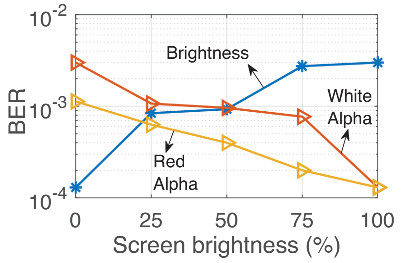


Figure 3.21: BER results vs. brightness/alpha

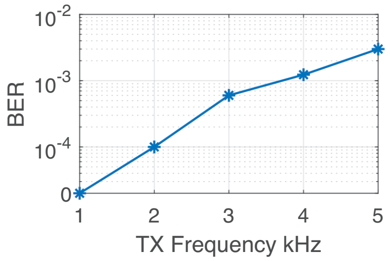


Figure 3.22: BER results vs. TX frequency

light source in the image, the less information can be decoded at the receiver. The circular lampshade achieves a slightly lower data rate compared to the square lampshade due to a smaller light source area. Note that even when small lampshades (e.g., a circular lampshade with a diameter of 40 cm) are employed, a data rate larger than 4 kb/s can still be achieved within a range of 2 m.

Impact of screen brightness and transparency. To avoid possible damage to human eyes, smartphones' screens adopt DC dimming instead of PWM dimming [95]. Therefore, the brightness of the screen determines how much interference it causes on the transmitted CSK signals. We carry out experiments to evaluate the effect of screen brightness on through-screen VLC. The BER results are shown in Figure 3.21. Overall, as the brightness increases, the BER gradually increases. When the brightness is lower than 50%, the BER is lower than 10^{-3}. When the brightness is higher than 50%, the BER is higher but still lower than 10^{-2}. On the other hand, as a well-known concept in computer graphics, the *alpha channel* is widely used to form a composite image with partial or full transparency [96]. It stores a value between 0 and 1 to indicate pixel translucency: 0 means that the pixel is fully opaque and 1 means that it is fully transparent. We also evaluate the impact of screen transparency on the performance of our system by changing the alpha value. The results are shown in Figure 3.21. We observe that changing the transparency of a pixel does not affect the color coordinates. The BERs under red and white screens both gradually decrease when the transparency increases, while the screen color coordinates detected do not change. This result is interesting because it means changing pixel transparency can be equivalent to adjusting screen's brightness.

Impact of transmitter frequency. To capture the effect of transmitter frequency (CSK symbols transmitted per second), we vary it from 1 kHz to 5 kHz at a step size of 1 kHz. Figure 3.22 shows the BER results when 8-CSK is used. We observe that as we increase the transmitter frequency, the BER increases. At 1 kHz, the BER is 0; at 2 kHz, the BER is around 10^{-4}; and at 4 kHz, the BER increases to 10^{-3}. This is because the width of the color stripe decreases with a higher transmitter frequency. This increases the inter-symbol interference as it becomes more difficult to distinguish colors with fewer pixels.

3.6.2. Robustness Evaluation

Impact of angle. We evaluate the robustness of our system when the receiver is placed in different directions with respect to the LED transmitter. We fix the distance between

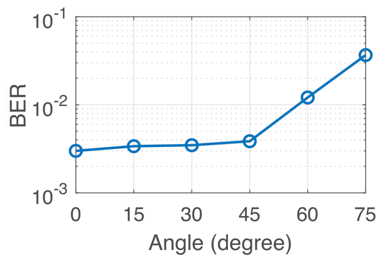


Figure 3.23: BER results vs. TX-camera angle.

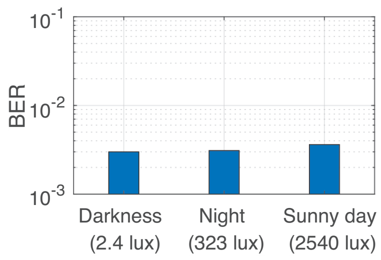


Figure 3.24: BER results vs. ambient light.

the transmitter and screen to 1 m and vary the relative angle between them. The results are shown in Figure 3.23. We can observe that the BER maintains almost at the same level when the angle between the transmitter and the receiver does not exceed 45°. At the angle of 60°, the BER increases to around 10^{-2}. The results show that our system can work well when the transmitter-receiver angle is below 60°.

Impact of ambient light conditions. Next, we evaluate the system's robustness under different ambient light conditions. Three ambient light conditions are considered: (1) Darkness (average light intensity: 2.4 lux); (2) Night with indoor illumination (average light intensity: 323 lux); and (3) Sunny day (average light intensity: 2540 lux). The impact of ambient light conditions on BER is presented in Figure 3.24. First, we observe that the BER under all three conditions is lower than 10^{-2}. Thus, the performance of the system is not affected much by the ambient light level. This is because the exposure time of the camera is usually set very small to exploit the rolling-shutter effect of the camera to detect the rapidly changing stripes. This makes the impact of interference from ambient light very limited on our system. In addition, even in a bright indoor environment (sunny day), with our proposed RoI detection algorithm, we can still accurately detect the light source, which further alleviates the impact on the BER performance.

Impact of distance. Lastly, we evaluate the system robustness when the transmitter is placed at different distances from the receiver. The BER results are shown in Figure 3.25. As the incident light attenuates with distance, a longer distance causes a lower signal strength at the receiver. The variance of the cluster formed by received CSK symbols on the CIE 1931 diagram becomes larger due to the decrease in SINR. Therefore, the larger the distance, the higher the BER. However, even at a distance of 4 m, the achieved BER is still lower than 10^{-2}. Following the state-of-the-art studies [62], [97], we define the *transmission range* as the maximal communication distance between the transmitter and receiver with a BER below 10^{-2}. The transmission ranges under different solutions as shown in Figure 3.26. We can observe that in the presence of the transparent screen that covers the under-screen camera, the 'Original' solution can only achieve a transmission range of 4 cm. On the other hand, by adopting the proposed pixel-sweeping and slope-boosting detection algorithms, the transmission range is extended to 4.24 m, bringing a gain of over 100×.

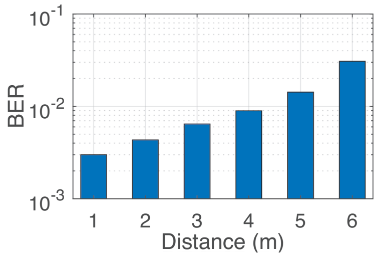


Figure 3.25: BER results vs. distance.

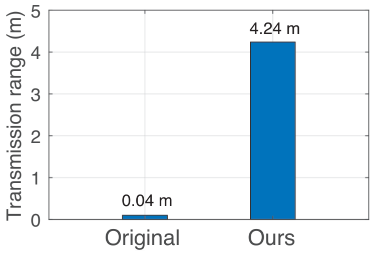


Figure 3.26: Communication range comparison.

3.6.3. DYNAMIC CONTENTS ON SCREEN

Next, we display dynamic contents on the transparent screen region and evaluate the impact on through-screen VLC. We change the transparent screen region's displayed color periodically following the order of *black, red, blue, green, magenta, cyan, yellow, and white*. We test four content update rates: 1 Hz, 25 Hz, 50 Hz, and 100 Hz, all of which are within the content update rate range of commonly used APPs.[6] The achieved BERs under different content update rates and different screen refresh rates are shown in Figure 3.27. We can observe that at all the content update rates, the BERs are always below 10^{-2}. We note that when the content update rate is 25 Hz, the BER is slightly higher. This is because the frame rate of the screen is extremely unstable at 25 Hz.

3.6.4. SCREEN DIVERSITY

We also evaluate the system performance with different commercial smartphones as the receiver. We test two commercial smartphones, ZTE AXON30 and Xiaomi MIX4 (cf. Figure 3.17(b)). The transparent screen regions of the two smartphones have dramatically different pixel layouts, sizes, and shapes. Also, their screens differ in maximum brightness. We fix the distance between the transmitter and receiver as 1 m and measure the BER of the through-screen VLC link when different smartphone screens are adopted. The results are shown in Figure 3.28. We can observe that although there are some performance variations under different smartphone screens, the BERs are always below 10^{-2}, demonstrating the effectiveness of our proposed pixel-sweeping algorithm. Interestingly, we find that the BER under the screen of MIX4 is higher than that of AXON30, even when the screen is not lit up (i.e., 'black'). We believe this is because MIX4 has a lower light transmittance compared to AXON30 due to intrinsic screen material and thickness diversity. When smartphones display white color at full brightness, MIX4 causes higher interference to the captured frames because it has a higher brightness.

3.6.5. MULTIPLE TRANSMITTERS

Finally, we carry out experiments to evaluate the system performance under multiple transmitters. The under-screen camera is a multi-pixel receiver. Therefore, an under-screen camera can detect the CSK signals from multiple LED transmitters by splitting a captured frame into several slices and detecting the RoI in each slice independently.

[6]In the experiment, we find that even when we set the screen refresh rate to 60 Hz and the content update rate to 100 Hz, due to the screen optimization mechanism in OS [98], the phone still displays contents at 60 Hz.

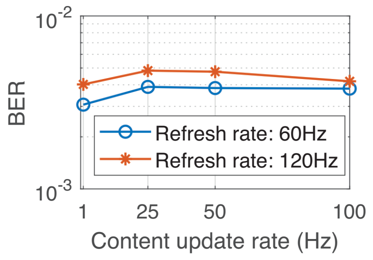


Figure 3.27: BER results vs. dynamic contents.

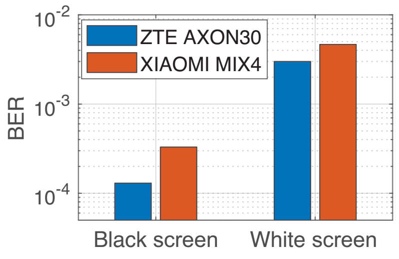
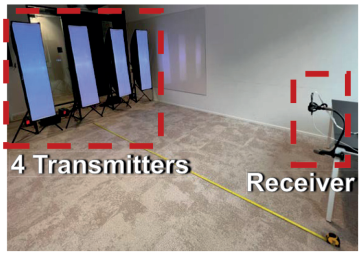
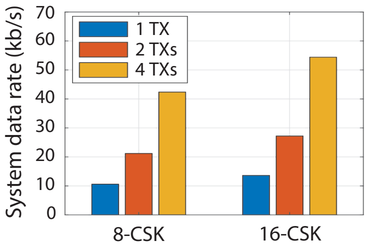


Figure 3.28: BER results vs. different screens.



(a) Multiple transmitters' setup



(b) System data rate vs. No. of TXs

Figure 3.29: Multiple transmitters results.

With multiple transmitters, the system data rate can be significantly increased. In our evaluation, we test up to four transmitters. We place the transmitters in a row at a distance of 3 meters from the receiver. A snapshot of the experiment setup with four transmitters is shown in Figure 3.29(a). We test both 8-CSK and 16-CSK. The evaluation results are shown in Figure 3.29(b). We can observe that under 8-CSK, the date rates achieved in our through-screen VLC system can reach 10.59 kb/s, 21.19 kb/s, and 42.38 kb/s with one transmitter, two transmitters, and four transmitters, respectively. Under 16-CSK, the maximum system data rate can go up to 54.43 kb/s with four transmitters.

3.7. Related Work

Image restoration on under-screen cameras. Under-screen cameras for smartphones have spurred interest in techniques that can restore images captured by them. A recent work [99] presented a method to recover dimmed and blurred images by restoring angular frequencies in the scene. Other works [9]–[13] also applied deep neural networks to handle the blur and low SNR issues associated with under-screen images. However, these works mainly addressed the passive interference on the camera induced by the screen when the screen is OFF. Instead, our work aims to remove the impact of active interference from the screen on VLC when the screen is on, which has never been considered in previous works.

LED-to-camera communication. It utilizes the existing lighting infrastructure as transmitters and pervasively available rolling shutter cameras as the receivers [28]–[30], [100].

Table 3.1: Summary of LED-to-Camera communication systems (in the one-transmitter and one-receiver setup).

Name	Modulation	Throughput	Range
VLandmark[101]	Binary FSK	10 b/s	1.2 m
RollingLight [26]	FSK	90.56 b/s	Not Specified
CamCom [102]	OOK	15 b/s	Not Specified
Seminal[25]	OOK	148 b/s	9 cm
ReliableVLC[27]	OOK	700 b/s	3 m
IoTorch [103]	PWM	2.92 kb/s	1 m
CeilingTalk [28]	OOK-PWM	1 kb/s	5 m
ReflexCode [29]	GSK	1.07 kb/s	3 m
Martian [100]	Prefix code	1.6 kb/s	25.4 cm
ColorBars [30]	CSK	5.2 kb/s	3 cm
Our work with USC	**CSK**	**13.61 kb/s**	**4.2 m**

The work in [28], [29] investigated a high-order intensity modulation by encoding data into different luminance levels. ColorBars exploited CSK to improve the data rates [30]. The data rate using pulse-based optical camera communication modulation methods such as OOK and FSK does not exceed 4.2 kb/s as shown in Table 3.1. ColorBars uses CSK modulation to reach a data rate of 5.3 kb/s, but the communication distance is only 3 cm. In our work, the dynamic contents on screens bring unique challenges for realizing through-screen VLC. Still a higher data rate and a longer communication range can both be achieved with our designed algorithms.

Screen-to-camera communication. This communication modality employs images or videos on a standard monitor to transmit data information [104]–[108]. However, these coded images are typically visible to users, comprising the confidentiality of the data content. Thus, hidden screen-to-camera communication was proposed to achieve communication without comprising the confidentiality [77], [78], [109]–[112]. In through-screen VLC, the screens become receivers with cameras under the screen as antennas. Transmitters are those ceiling lamps which are used for both illumination and communication. We utilize the color information and apply dedicated signal processing methods to achieve a throughput of 54.43 kb/s and a range of 4.2 m with four transmitters.

3.8. CONCLUSION

In this chapter, we studied how to enable through-screen VLC with under-screen cameras on full-screen devices such as smartphones. Unlike the SpiderWeb system discussed in Chapter 2, which discovered the color-pulling effect and proposed the SWebCSK modulation scheme to address screen-induced challenges, our design does not require any modifications to LED transmitters. SpiderWeb necessitated transmitters to be aware of the receiving device's current screen color and to optimize its modulation constellation points accordingly, introducing significant overhead for transmitter-receiver synchronization and constellation optimization. In contrast, proposed approach in this chapter allows for a higher data rate without such modifications. We identified key challenges associated with the transparent OLED screen covering the under-screen camera and proposed solutions such as the pixel-sweeping algorithm and slope-boosting demodulation method to address these challenges. Our comprehensive experiments demonstrated the feasibility of through-screen VLC, and we anticipate that our work will inspire further research on communication and sensing using under-screen cameras.

Future work. Our system works well when the entire transparent screen region displays a single color (the color can change over time), as shown in our evaluation. The transparent screen region is located at the top of the smartphone and this part usually hosts the status bar. Thus, the transparent screen region usually only displays a single color at a time, such as white in many APPs. However, for APPs running on a full screen such as games, complex contents with different colors could be displayed in the region. In this case, every single frame captured by the under-screen camera is disturbed by a different color, making decoding more challenging. For this scenario with multi-color screen contents, one potential solution is to cancel the effect of the screen color at each single screen-pixel. This approach, however, introduces new challenges, such as accurately determining color interference at the pixel level. In high-order CSK modulation, an inability to accurately decode color interference can lead to higher bit error rates and significant information loss. Beyond communication, full-screen devices also hold promise for sensing applications. For example, the screen can serve as a light source and under-screen sensors as receivers to enable through-screen air-writing recognition, using reflections from a finger close to the screen. This design benefits from having a fixed screen light source, eliminating the need for additional external light sources, for instance, fingertip air-writing [113] could be a typical application for through-screen visible light sensing system.

Up till now, we have demonstrated how to overcome screen interferences and achieve through-screen light communication using Commercial Off-The-Shelf (COTS) components on mobile devices. However, we must also consider the original utility of these COTS components, such as the primary visual imaging function of the under-screen camera. In the next chapter, we will discuss our findings on how the screen impacts the imaging function of under-screen cameras and explore the implications for user security and privacy on mobile devices.

4 Screen Perturbation: Adversarial Attack and Defense on USC

To allow visible light to reach the Under-Screen Camera (USC) and thereby preserve its photographic capabilities [8], a special region, *transparent screen region*, is positioned above the USC. Unlike pressure or fingerprint sensors that can be easily integrated into a screen, maintaining the full functionality of an original front camera after mounting it behind a transparent screen is relatively challenging. The imaging quality of an USC is severely degraded due to lower light transmittance and diffraction effects, resulting in noisy and blurry images. Consequently, while enhancing user experience and interaction, the USC may compromise the quality of photography, face processing, and other downstream vision tasks. Numerous recent research and industry efforts have focused on restoring and enhancing images captured by USC systems [10]–[12], [14], [114]. However, a more critical issue that has not been adequately addressed is the security concern associated with USCs. In this chapter, we explore the following research question:

What are the key security impacts of the screen on the visual imaging of the under-screen camera?

Motivated by this new feature of full-screen mobile devices, we investigate how the transparent screen region lighting significantly impacts the images captured by USC. This phenomenon is corroborated by the findings and conclusions in Chapter 3. However, in this chapter, we explore it from a security perspective, considering how it can be exploited for both attacking and safeguarding purposes.

In this chapter, we propose the concept of *Screen Perturbation*, which modifies the pixels displayed on the transparent screen region to nullify deep learning models such

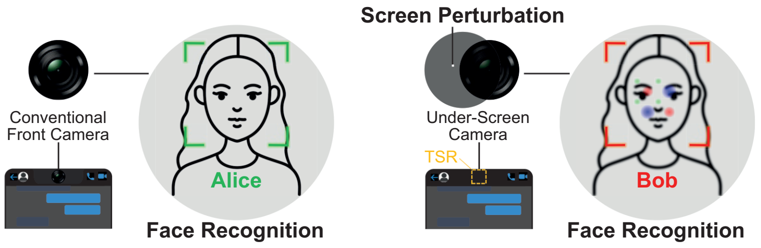


Figure 4.1: Illustration of screen perturbation on an example of in defending unauthorized face recognition: carefully-selected pixels on the transparent screen are lit up, which are 'imperceptible' to human eyes, but could cause an unauthorized system to misrecognize the user Alice as Bob. (Transparent Screen Region: TSR)

as image classification and face recognition models. The proposed screen perturbation is a *double-edged sword*. *On one hand*, it can serve as a defensive mechanism. Like traditional front cameras, hackers can exploit USCs through camfecting [44] to capture images and record videos. With the rise of deep learning models in security-critical applications like face authentication [115], these illegally captured images can be used to train extensive models capable of recognizing millions of individuals without their consent. To tackle this security risk, users can leverage the transparent screen region to embed adversarial perturbations into captured images, thereby protecting themselves from unauthorized deep-learning models. For instance, an unauthorized face recognition system would fail to correctly identify users based on these perturbed images, as depicted in Figure 4.1. Users can proactively activate specific screen pixels during their interactions with the smartphone, ensuring that any images covertly captured by USC are not correctly identified by unauthorized facial recognition models. Importantly, this does not interfere with other functionalities, such as the motion detection capability of USC. *On the other hand*, the transparent screen region can be exploited by malicious attackers. They can embed adversarial perturbations into captured images to launch adversarial attacks, aiming to disrupt legitimate face recognition systems or fool image classification models. To fully harness the potential of screen perturbation and mitigate its risks, we must address several unique challenges. For simplicity of presentation, we describe these challenges and our solutions and contributions *from the attacker's perspective* in the rest of this section.

Challenges and Contributions. To launch adversarial attacks that can fool applications such as face recognition and image classification, the attacker needs to manipulate the content displayed in the transparent screen region to embed adversarial perturbations in the formed images. This is challenging because the modification in the transparent screen region needs to be carefully designed such that the changes are imperceptible to the users, while the generated image perturbations are powerful enough to fool the deep learning models. To achieve this goal, the first challenge is

Challenge 1: *How to understand the impact of the transparent screen region's perturbation on the formed image at the USC?*

Current studies [10], [14], [15] have discovered that when the transparent screen is not illuminated, it can still scatter and absorb light, leading to a lower signal-to-noise ratio and a color shift in the formed image; we refer to this as *passive perturbation*. However, the effect of the screen's illuminated pixels on the images captured by the USC has never been explored. Our research reveals that when the transparent screen region is lit for displaying content, the illuminated screen pixels embed various translucent speckled color blocks and color shifts in the formed image. We term this *active perturbation*. Factors such as whether the pixels in the transparent screen region are illuminated, their colors, and their brightness levels strongly affect these perturbations in the formed image. To address this, we develop a comprehensive model to study the impact of both passive perturbations and active perturbations on USC's image formation. Building upon this model, we successfully create an USC image simulator that generates both passive and active perturbations on captured images simultaneously, allowing us to quantitatively understand the impact of screen perturbation on under-screen image formation.

After analyzing the impact of screen perturbations on images captured by the USC, caused by activated and inactivated screen pixels on the transparent screen region, it is essential to carefully select the pixels on the transparent screen region to produce screen perturbations that achieve the desired attack effect. Although the size of the transparent screen region is small (only about 0.1 cm^2), it contains *several thousand to twenty thousand pixels*. For example, the smartphone Fold4, AXON30, and MIX4 have 58×58, 108×60, and 108×60 pixels within their transparent screen region, respectively. Each pixel has multiple R/G/B sub-pixels, making it challenging to identify the most suitable pixels for creating the screen perturbation and how to light them up (using optimal colors and brightness). The simplest way could be to light up all the pixels in the transparent screen region at a proper brightness level to create the screen perturbation. However, this is inefficient and the created screen perturbation will be striking, and therefore, users can perceive it. Hence, the second challenge is

Challenge 2: *How to determine which transparent screen region pixels to manipulate and how, in order to ensure that the created screen perturbation can successfully fool deep learning models?*

To address this challenge, we present the chromaticity destruction and morphology destruction modules. Specifically, the chromaticity destruction determines the optimal color of the screen perturbation and maximizes its attacking region and energy intensity. The morphology destruction module optimizes the position and brightness of the screen-pixel perturbations. Leveraging these two modules, we design a one-pixel perturbation approach, wherein modifying only a single pixel on the screen *(less than 1‰ of the pixels in the transparent screen region)* we can reduce the average image classification accuracy of six deep learning models from 85% to 14%, and reduce the average face recognition accuracy of two deep learning models from 91% to 1.8%. To enhance attack efficiency, we propose a multiple-pixel perturbation that modifies only a few screen pixels *(less than* 1% *of the pixels in the transparent screen region)* to generate perturbation with higher adversarial strength, thereby further decreasing the average accuracy of image classification and face recognition to as low as 5.5% and 0.25%, respectively. We summarize our contributions as follows:

- To our best knowledge, we are the first to discover this critical security phenomenon at USC.

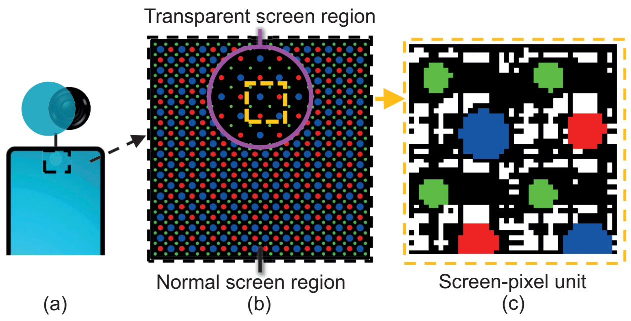


Figure 4.2: Illustration of (a) smartphone with a under-screen camera; (b) micrograph of the transparent screen region and the normal screen region; and (c) a typical structure of the screen-pixel unit.

- We analyze the imaging formation of USC both theoretically and experimentally. Theoretical models are successfully established, and we build an image captured on the USC simulator based on those models.
- We design and implement one-pixel and multiple-pixel perturbations including chromaticity and morphology destruction modules, which could generate imperceptible but powerful screen perturbations to fool deep learning models.
- We thoroughly evaluate the system on a dataset collected by three smartphones equipped with USC and synthesized datasets generated using our USC image formation simulator.

4.1. PRIMER ON SCREEN PERTURBATION

4.1.1. STRUCTURE OF TRANSPARENT SCREEN REGION

In Under-Screen Camera (USC) smartphones [88], [113], [116]–[118], the front-facing camera is placed under a small 'transparent' screen area known as the transparent screen region built on transparent electrode material. As shown in Figure 4.2(b), the transparent screen region has a different pixel layout from the normal screen region, and can be considered as an RGBG array that consists of multiple *screen-pixel units*. Figure 4.2(c) shows a typical structure of the screen-pixel unit, which is made up of three R/G/B subpixel sets. Depending on the manufacturer, a R/G/B subpixel set can contain multiple subpixels of the same color. For example, in the design shown in Figure 4.2(c), the R/G/B subpixel set is made up of two red subpixels, four green subpixels, and two blue subpixels. We can further divide the transparent screen region into three functional areas:

- **Screen-non-pixel area** (the white area) allows light to pass through the screen.
- **Control-circuit area** (the black area) is mainly used to control the circuit wiring of the screen, and has a lower light transmittance than the screen-non-pixel area.
- **Screen-pixel area** (the RGB color matrix) consists of R/G/B subpixel sets that can be lit up to display contents.

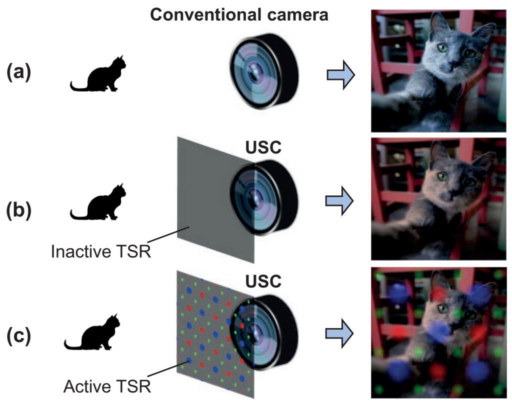


Figure 4.3: Images formed in different scenarios: (a) pristine image is captured by a traditional front-facing camera; (b) the transparent screen region is inactive but still introduces *passive screen perturbation* to the captured image, i.e., color shift and low signal-to-noise ratio; (c) the transparent screen region is active and introduces both *passive and active screen perturbations* to the final image, i.e., translucent speckled color blocks and color shifts. (Transparent Screen Region: TSR)

4.1.2. UNDER-SCREEN IMAGE PERTURBATIONS

The unique structure of the transparent screen region provides an intriguing way to maximize the display area. However, it inevitably introduces two types of perturbation during image formation. First, as the transparent screen region is placed above the camera, the screen-non-pixel and the control-circuit areas introduce **passive perturbations** on the formed images. As shown in Figure 4.3(b), passive perturbations are embedded into the image even when the transparent screen region is inactive and the screen is completely turned off due to light scattering and absorption from the fine pixel pitch [10]. The passive perturbation causes a lower signal-to-noise ratio and color shifts in the image [10], [14], [15]. In addition, as shown in Figure 4.3(c), when the screen is lit up for display, the screen-pixel area introduces **active perturbations** to the image. The lighting of different R/G/B subpixel sets embeds various translucent speckled color blocks and color shifts in the final image captured by the under-screen camera.

4.2. SYSTEM OVERVIEW

4.2.1. THREAT MODEL

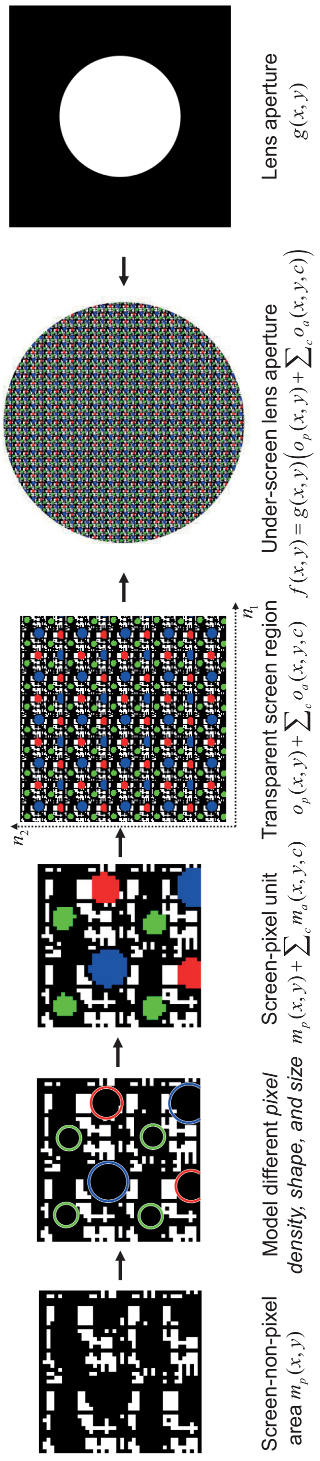
In this chapter, we investigate and demonstrate that the new type of screen perturbation is a double-edged sword for users. On one hand, malicious attackers can exploit it to embed adversarial perturbations into captured images, launching adversarial attacks aimed at disrupting the performance of legitimate face recognition systems [119] or fooling image classification models [120]. On the other hand, benign defenders, e.g., users, can leverage screen perturbation to protect themselves from unauthorized deep learning models. For instance, users can proactively activate specific screen pixels in use. Thus, any images covertly captured by the under-screen camera would not be correctly identified by unauthorized facial recognition models [115], [121], while other functionalities, such as the motion detection capability of the under-screen camera, would remain unaffected. Next, we motivate our system design from the perspective of the attacker/defender's goal, capability, and background knowledge.

Attacker/defender's goal. We consider an attacker/defender aims to manipulate the content displayed in the transparent screen region to generate and embed adversarial perturbations to images captured by the under-screen camera. As a result, a machine learning model (referred as **target model**) trained on images captured by a standard camera should exhibit low accuracy for those corrupted images. The alterations to the displayed content should be imperceptible and not interfere with the normal content displayed in the transparent screen region.

Attacker/defender's capability and background knowledge. We begin by assuming that a benign or malicious application is installed by the attacker or defender. This application has the ability to control all screen-pixel units in the transparent screen region and can modify the content displayed on the screen. Android platforms do not restrict UI settings, allowing any applications to set their own UI displays and perturb the camera input. For devices such as Samsung Galaxy Z Fold 4, ZTE AXON 30, and Xiaomi MIX 4 which we investigate in this chapter, the content displayed on the transparent screen region can be easily modified using the Android Canvas Drawing API [122]. We also assume that these screen-pixel modifications are not confined to a single application but can be implemented across different applications, as Android systems support UI modifications between various applications, including multi-window settings [123] and notification pop-ups [124]. Next, we assume that the attacker or defender has access to a pristine image for obtaining object information. Lastly, the attacker/defender can generate different screen perturbations by modifying the content displayed on the screen for various usage scenarios. These modified pixels on the screen are invisible to the naked eye, ensuring that they do not affect the user's normal use of the smartphone in benign applications and do not alert the user to the embedded screen perturbation in malicious applications. To consider a more realistic scenario, we assume the attacker/defender lacks knowledge about the target model's implementation details used to classify captured images, but they have access to a **surrogate model** for the task, which could have different architecture and hyper-parameters from target models. For simplicity, we describe our design **from the attacker's perspective** in the rest of the chapter.

4.2.2. Overall Design

In a nutshell, we propose a suite of algorithms that manipulate only a few screen-pixel units with optimized pixel location, color, and brightness in the transparent screen region to embed adversarial perturbations in the formed images. The screen modifications are imperceptible to the human eyes (cf. Figure 4.10), but are powerful enough to fool machine learning models. To this end, we first establish a body of image formation models that serve as the theoretical basis of our design. Specifically, we devise the active perturbation function to model the image perturbation introduced by the screen-pixel area (in Section 4.3.2), and simulate how different color, brightness, and position of the subpixel set affect the active perturbation. Further, we introduce the under-screen aperture function to model the passive perturbation caused by the screen-non-pixel area in the



Screen-non-pixel area $m_p(x,y)$
Model different *pixel density, shape, and size*
Screen-pixel unit $m_p(x,y)+\sum_c m_a(x,y,c)$
n_2
n_1
Transparent screen region $o_p(x,y)+\sum_c o_a(x,y,c)$
Under-screen lens aperture $f(x,y)=g(x,y)\left(o_p(x,y)+\sum_c o_a(x,y,c)\right)$
Lens aperture $g(x,y)$

Figure 4.4: Modeling the effective under-screen lens aperture.

transparent screen region (in Section 4.3.3). Finally, we design an under-screen image formation pipeline to model and synthesize images that are corrupted by both passive and active perturbations in various under-screen imaging settings. Building on our theoretical models, we introduce two novel mechanisms to determine the configuration of the transparent screen region to generate effective but imperceptible screen perturbations. Specifically, we propose the chromaticity destruction module (in Section 4.4.2) to determine the optimal color, and the morphology destruction module (in Section 4.4.3) to optimize the position and brightness of the screen-pixel unit in generating active perturbations. To improve the attack efficiency, we introduce the multiple-pixel perturbation (in Section 4.4.4) that maximizes the adversarial strength of the active screen perturbation while being imperceptible to human eyes.

4.3. Modeling USC's Image Formation

Below, we introduce the modeling of image formation for the under-screen camera. Firstly, we discuss the concept of the blur-kernel, and then specialize it to model the active and passive perturbation introduced in Section 4.1.2.

4.3.1. Blur Kernel

Light diffraction through the transparent screen region in a camera system can cause image degradation due to the comparable size of the screen-non-pixel area and the visible light wavelength [10]. Following Fourier optics principles [125], we model this diffraction phenomenon using the blur-kernel, which is also known as the point spread function [14]. The blur kernel mathematically depicts the spread of light around a point light source in an image, characterizing the blurring effect of an imaging system or physical medium on the image. The blur kernel is the squared magnitude of the scaled Fourier transform of the under-screen aperture function $f(x, y)$, derived from the product of the camera lens aperture and the transparent screen region:

$$f(x, y) = g(x, y)o(x, y), \tag{4.1}$$

where (x, y) represents the coordinates of the screen-pixel unit. The camera lens is approximated as a thin lens with an aperture function $g(x, y)$, while the transparent screen region is modeled as a function $o(x, y)$ that maps the light transmittance properties of coordinates (x, y) in the transparent screen region to a range between 0 and 1. A value of 1 indicates that light can fully pass through the transparent screen region, whereas a value of 0 indicates that light cannot pass through the transparent screen region. The blur kernel is defined as follows:

$$k(\lambda, F) = \left| \frac{1}{\lambda r_0} F\left(\frac{x}{\lambda r_0}, \frac{y}{\lambda r_0} \right) \right|^2, \tag{4.2}$$

where λ is the wavelength of light, r_0 is the focal length of the camera lens, and $F(u, v)$ is the Fourier transform of the under-screen aperture function $f(x, y)$. The (u, v) is the mapping of coordinates (x, y) in the frequency domain.

Based on the blur-kernel model, obtaining the under-screen aperture function of active and passive perturbations for the transparent screen region is necessary to generate the corresponding blur kernels. The under-screen aperture functions for both types of perturbations will be presented below, as illustrated in Figure 4.4.

4.3.2. Modeling Screen's Active Perturbation

USCs in mobile devices like smartphones are placed near the screen for a slim profile, leading to the transparent screen region co-locating with the thin lens aperture. This proximity allows the R/G/B subpixel sets on the transparent screen region to be approximated as point light sources per the Huygens-Fresnel principle [125]. The transparent screen region's screen-pixel area is periodic, with identical screen-pixel units, enabling modulation of the entire area via a single unit. Let D μm be the inter-pixel distance of the pixels in the transparent screen region. D actually determines the screen resolution as $25,400/D$ pixels per inch (= $25,400$ μm). The light intensity of screen-pixel area within a screen-pixel unit is denoted as $m_a(x, y, c)$, where $c \in \{R, G, B\}$ represents the R/G/B subpixel sets. This $m_a(x, y, c)$ repeats at a periodicity of D along both axes to form all screen-pixel areas on the transparent screen region. Here, we can set R/G/B subpixel sets *shape*, *size*, and *brightness* in the screen-pixel unit to impact the active screen-pixel perturbation. However, the shape and size of each R/G/B subpixel set are typically predetermined during manufacturing. The active perturbation can still be manipulated by selectively lighting up different R/G/B subpixel sets on the transparent screen region to form different patterns. Using n_1 and n_2 to represent the horizontal and vertical indices in the spatial domain and N_1 and N_2 for the frequency domain, we can express the light intensity of (x, y) of different R/G/B subpixel sets on the transparent screen region as:

$$o_a(x, y, c) = m_a(x, y, c) * \sum_{n_1} \sum_{n_2} \delta(x - n_1 D)\delta(y - n_2 D), \tag{4.3}$$

where $\delta(\cdot)$ is the Dirac delta function, also known as the unit impulse [125]. Multiplication in the space domain leads to convolution in the frequency domain, which allows us to obtain $F_a(u, v, c)$, the Fourier transform of the effective aperture at different wavelengths (R/G/B) for active perturbation:

$$F_a(u, v, c) = \sum_{N_1} \sum_{N_2} M_a\left(\frac{N_1}{D}, \frac{N_2}{D}, c\right) G\left(u - \frac{N_1}{D}, v - \frac{N_2}{D}\right), \tag{4.4}$$

where $M_a(u, v, c)$ represents the Fourier transform of R/G/B subpixel set in the screen-pixel area $m_a(x, y, c)$, $G(u, v)$ is the Fourier transform of the camera lens aperture $g(x, y)$, and N_1 and N_2 are the indexes corresponding to n_1 and n_2 in the frequency domain.

4.3.3. Modeling Screen's Passive Perturbation

We present a specialized under-screen aperture function expression, focusing on passive perturbation from the screen-non-pixel area. Given its higher light transmittance compared to the control circuit and screen-pixel areas, we can neglect the latter [14]. The screen-non-pixel area within a screen-pixel unit, denoted as $m_p(x, y)$ (cf. the white area in Figure 4.2), also exhibits periodicity. This allows us to mathematically express the light transmittance properties of (x, y) on transparent screen region's screen-non-pixel area, $o_p(x, y)$, and the Fourier transform of the effective aperture for passive perturbation:

$$o_p(x, y) = m_p(x, y) * \sum_{n_1} \sum_{n_2} \delta(x - n_1 D)\delta(y - n_2 D), \tag{4.5}$$

$$F_p(u,v) = \sum_{N_1}\sum_{N_2} M_p\left(\frac{N_1}{D},\frac{N_2}{D}\right) G\left(u-\frac{N_1}{D}, v-\frac{N_2}{D}\right). \tag{4.6}$$

4.3.4. Under-Screen Image Formation Pipeline

Based on the analysis in the previous sections, and given a calibrated screen-pixel unit as shown in Figure 4.4, we can model the degraded images from a scene with both passive and active perturbation of transparent screen. For the scene $\boldsymbol{I}$, the degraded observation $\boldsymbol{I} \oplus \boldsymbol{z}$ formed on the sensor can be modeled as a convolution process, given by:

$$\boldsymbol{I} \oplus \boldsymbol{z} = \underbrace{(\gamma\hat{\boldsymbol{z}}) \otimes \sum_c k(\lambda, F_a(c))}_{\text{Active Perturbation}} + \underbrace{(\gamma\boldsymbol{I}) \otimes k(\lambda, F_p)}_{\text{Passive Perturbation}} + \underbrace{n}_{\text{Noise}}, \tag{4.7}$$

where γ is the intensity scaling factor considering camera gain and screen attenuation. The blur kernels for active and passive screen perturbations are $\sum_c k(\lambda, F_a(c))$ and $k(\lambda, F_p)$, respectively. The noise is represented by n, which includes both shot noise and read-out noise. $\hat{\boldsymbol{z}}$ (in pixels) represents an active perturbation image on USC projected from the R/G/B subpixel sets on the transparent screen region, represented by $\boldsymbol{z}$ (in mm). The camera's imaging process projects three-dimensional objects onto a two-dimensional plane (cf. Figure 4.3). Given the screen's proximity to the USC, the projection process can be approximated as a conversion from the camera to the pixel coordinate system. Following the perspective camera model [96], we denote this projection as $\hat{\boldsymbol{z}} = \text{Resize}(\boldsymbol{z}, r_0)$, where r_0 is the USC's focal length.

4.4. Screen-Pixel Perturbation

In this section, we first introduce how to identify the potential attacking region in images captured by USC (Section 4.4.1). We then present the chromaticity destruction module that selects the optimal color for the screen-pixel perturbation (Section 4.4.2), followed by the morphology destruction module that determines the optimal position and brightness of the screen-pixel perturbation (Section 4.4.3). Combining the above two modules, we can activate a single screen-pixel unit on a transparent screen region and add corresponding perturbation in images captured by USC, and we call it **one-pixel perturbation**. Lastly, we introduce the **multiple-pixel perturbation** to improve the attacking efficiency by illuminating multiple screen-pixel units simultaneously (Section 4.4.4).

Design space. Different from the theoretical model, the design space when generating screen-pixel perturbation on practical under-screen camera smartphones is smaller. Specifically, hardware-related parameters, such as shape, size, and the number of pixels on the screen, are fixed by the manufacturer. Instead, we have access to each screen-pixel unit in the transparent screen region, and can program the color and brightness of the R/G/B subpixel set in the screen-pixel unit.

4.4.1. Calculating the Attacking Region

Our goal is to generate adversarial perturbations by modifying as few screen-pixel units as possible (to be less perceptible), while greatly degrading the classification accuracy of the victim deep learning model. To achieve this, we first need to localize the region in the targeted image that has the highest influence on the decision making of the deep

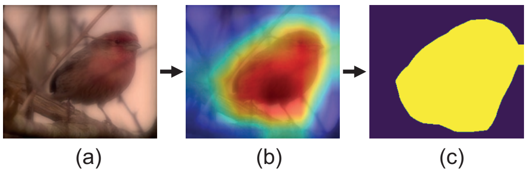


Figure 4.5: Example of using the Grad-CAM for attacking region estimation: (a) taking an image captured by the USC as input, (b) we leverage the Grad-CAM to generate the heatmap of the most significant region in the image, and then (c) obtain the attacking region.

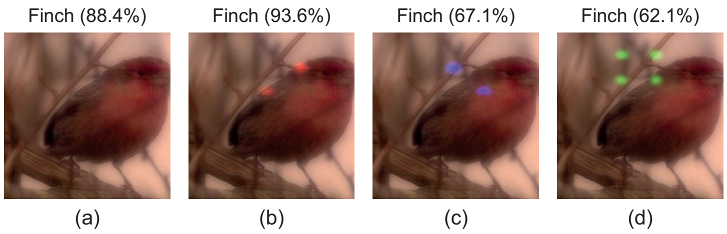


Figure 4.6: The adversarial strength of the screen-pixel perturbation with different colors: (a) image without active perturbation; by contrast, in (b), (c), and (d), the red, blue, and green subpixel set is used to create the screen-pixel perturbation, respectively.

learning model. We leverage the Gradient-weighted Class Activation Mapping (Grad-CAM) [126] to estimate the attacking region where the screen perturbation will be added.

Figure 4.5 shows the pipeline in calculating the attacking region. Taking an image captured by the USC as input, we first leverage the Grad-CAM to calculate a heatmap of the image based on the surrogate model. As shown in Figure 4.5(b), the heatmap highlights the most significant region in the image that drives the surrogate model to its final prediction. Then, by calculating the average value of the heatmap and thresholding it, we can calculate the attacking region that indicates the potential image area where adversarial perturbations should be added to. Note that for a given image, the heatmap generated by the Grad-CAM varies when different architectures and training datasets are used by the surrogate model. However, because of the transferability effect (cf. Section 4.5), different models trained for similar tasks will share highly similar properties and vulnerabilities, even when they have different architectures and are trained on different datasets [127]–[131]. Thus, we can leverage one representative model as the surrogate model for attacking region estimation, but still be able to apply and transfer the resulting adversarial perturbations to different models in the attacking stage.

4.4.2. Chromaticity Destruction

The ability of an adversarial perturbation to fool the classifier is known as the adversarial strength. In general, for colorization-based perturbation [132], its adversarial strength is proportional to the number of perturbed pixels [133], [134] and the energy change in the pixels [135]. Based on these facts, we introduce the chromaticity destruction to generate the screen-pixel perturbation in three steps.

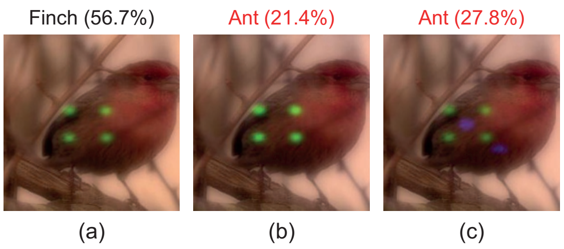


Figure 4.7: Attacking with different brightness levels and number of subpixel sets used: (a) lighting up the green subpixel set; (c) lighting up the green subpixel set with a high brightness; (d) lighting up both green and blue subpixel sets but with a lower brightness.

Step 1: Identifying the dominant color channel. First, we identify the dominant channel of the attacking region. The dominant channel is one of the R/G/B channels with the highest color intensity. Next, for a given screen-pixel unit, we light up one or two of its R/G/B subpixel sets that are distinct from the dominant channel to generate corresponding screen-pixel perturbations. By doing so, the color of the resulting screen-pixel perturbation will be distinct from the dominant channel of the attacking region, which causes the 'color shift' effect on the targeted image with maximized energy change in the perturbed region. Figure 4.6 shows the adversarial strength of the screen-pixel perturbations generated by lighting up different subpixel sets. Figure 4.6(a) exhibits the baseline scenario where no active perturbation has been added. The surrogate classifier utilizes ResNet-50 and is trained on miniImageNet dataset. As shown, the surrogate classifier can correctly recognize the object as 'finch' with a probability score of 88.4%. In Figure 4.6(b), because R channel is the dominant channel of the attacking region, lighting up the red subpixel set to generate the screen-pixel perturbation does not lead to the expected mis-classification. Instead, the 'finch' can still be correctly recognized with a high probability score of 93.6%. By contrast, as shown in Figures 4.6(c) and (d), a screen-pixel perturbation generated by lighting up the blue or the green subpixel set can dramatically degrade the probability score of the surrogate model to 67.1% and 62.1%, respectively.

Step 2: Prioritizing R/G/B subpixel sets. As a larger image perturbation has a higher adversarial strength [133], [134], [136], we prioritize the three color subpixel sets in the order of green, blue, and red, according to their actual sizes in the screen-pixel unit. Specifically, the green subpixel set has the highest preference, as the screen-pixel unit in most OLED screens utilizes an RGBG structure, resulting in twice as many green subpixel sets as red and blue subpixel sets (cf. Figure 4.2). Thus, green subpixel set can perturb more image pixels and has the largest perturbation size. The blue subpixel set is the second preference, as it has the second-large area. This is because the blue subpixel material has the shortest lifespan, and the area of the blue subpixel set is always maximized to extend the lifespan of the screen [86].

Step 3: Lighting up two subpixel sets. Figure 4.6 shows that lighting a single subpixel set cannot ensure the perturbation is strong enough to fool the classifier. To improve the adversarial strength, we can either increase the number of perturbed subpixels (by lighting up two subpixel sets simultaneously) or enhance the energy change in the subpixels (by increasing the brightness of the subpixel set). However, as the human visual system is more sensitive to luminance than chrominance, we opt to light up two subpixel sets simultaneously while minimizing their brightness. Figures 4.7(a) and (b) show that by increasing the brightness of the green subpixel, the 'Finch' is misclassified as 'Ant', but the perturbation becomes more perceptible. By contrast, Figure 4.7(c) shows that lighting up both green and blue subpixels at a lower brightness can still fool the classifier.

4.4.3. MORPHOLOGY DESTRUCTION

Below, we introduce the morphology destruction, which optimizes the location and brightness of the screen-pixel perturbation. We use the following notations in our design.

- $\boldsymbol{I}$: is the targeted pristine image with true label l.
- $\boldsymbol{z}$: is the original configuration of the unperturbed screen-pixel units in the transparent screen region.
- $\boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}}$: the configuration of the screen-pixel units when generating the adversarial perturbation, which is configured by screen-pixel index j, color vector $\boldsymbol{c}$, and subpixel brightness vector $\boldsymbol{b}$. Specifically, j indicates that the jth screen-pixel unit in the transparent screen region will be manipulated; $\boldsymbol{c}$ is a three-dimensional vector that indicates which of the three R/G/B subpixel sets will be light up; similarly, $\boldsymbol{b}$ is a three-dimensional vector that indicates the brightness of the three subpixel sets.
- $\oplus$: denotes the image formation process of the under-screen camera, which is defined in Equation (4.7).
- Φ: denotes the surrogate classifier.
- $\Phi(\boldsymbol{I}, l)$: denotes the probability of surrogate classifier Φ in classifying the image $\boldsymbol{I}$ with label l.

The goal of the morphology destruction is to search for a configuration $\boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}}$ that can fool the surrogate classifier Φ. As we only change a single screen-pixel, we aim to find the screen-pixel index j such that the surrogate classifier is most likely to misclassify the object of interest. Assuming a perturbation at the jth screen-pixel, we need the probability of an incorrect label surpassing the true label l to trick Φ. To reach the goal, we aim to find the most susceptible label when the perturbation is added at jth screen-pixel. In particular, we use the growth rate of the output probability of the surrogate classifier Φ to measure the susceptibility for each incorrect label $\hat{l}$ ($\hat{l} \neq l$). Our intuition is that the output probability of the surrogate classifier Φ for an incorrect label is more likely to increase more when it has a larger growth rate. Formally, the growth rate for $\hat{l}$ can be computed as follows:

$$s(\hat{l}, j) = \frac{\Phi(\boldsymbol{I} \oplus \boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}}, \hat{l}) - \Phi(\boldsymbol{I} \oplus \boldsymbol{z}, \hat{l})}{\Phi(\boldsymbol{I} \oplus \boldsymbol{z}, \hat{l})}, \tag{4.8}$$

where $s(\hat{l}, j)$ is the growth rate; term $\boldsymbol{I} \oplus \boldsymbol{z}$ is the image captured by the under-screen camera with unperturbed screen configuration $\boldsymbol{z}$; term $\boldsymbol{I} \oplus \boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}}$ is the image captured with the perturbed configuration $\boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}}$; $\boldsymbol{c}$ is calculated by the chromaticity destruction, which indicates the lighting of the R/G/B subpixel sets given input $\boldsymbol{I} \oplus \boldsymbol{z}$; $\boldsymbol{b}$ is set to a low brightness level of $\boldsymbol{b}_0 = 0.01$. Given the growth rate for each incorrect label, we view the label whose growth rate is the largest as the most susceptible label, i.e., $\tilde{l}_j =_{\hat{l} \neq l} s(\hat{l}, j)$.

Recall that our goal is to find the screen-pixel index j that is most likely to cause the misclassification of the surrogate classifier. We reach the goal by finding the j whose most susceptible label has the largest growth rate. Formally, we can find j by solving the following optimization problem:

$$j =_{\hat{j}} s(\tilde{l}_{\hat{j}}, \hat{j}), \tag{4.9}$$

where $s(\tilde{l}_{\hat{j}}, \hat{j})$ is the growth rate of the most susceptible label $\tilde{l}_{\hat{j}}$ when the perturbation is added at $\hat{j}$th screen-pixel.

After finding the optimal index j, we calculate the minimum required brightness $\boldsymbol{b}$ by a one-directional search process. Initially, we set the brightness of the screen-pixel unit to a small value $\boldsymbol{b} = \boldsymbol{b}_0$. We then gradually increase $\boldsymbol{b}$ with a small step size $\Delta\tau$. The search is terminated when:

$$\Phi(\boldsymbol{I} \oplus \boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}}, l) < \Phi(\boldsymbol{I} \oplus \boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}}, \hat{l}), \exists \hat{l} \neq l, \quad \text{or} \quad \boldsymbol{b} > \varepsilon, \tag{4.10}$$

which means either the perturbation generated by $\boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}}$ is able to fool the surrogate classifier with brightness $\boldsymbol{b}$ or the maximum screen-pixel intensity budget ε is reached. This process repeats for each subpixel set of the chosen colors until the optimal set requiring the least brightness is found.

4.4.4. Multiple-Pixel Perturbation

We are motivated by the fact that when using the smartphone and looking at the screen at a certain distance, the human visual system has a high tolerance for screen pixel changes. Current research has shown that a change of 120 screen-pixels is indistinguishable [137] per degree of viewing angle when viewing at a normal distance. Thus, instead of lighting up single screen-pixel unit, we can potentially manipulate multiple screen-pixel units to enable a larger perturbation area. This can greatly improve the adversarial strength of the generated screen-pixel perturbation while ensuring the changes on the transparent screen area are unnoticeable.

First, we introduce the process of selecting multiple screen-pixel units to generate the perturbation. As shown in Figure 4.8, we apply the morphology destruction method (cf. Section 4.4.3) to determine the position and brightness of multiple screen-pixel units for perturbation generation when the one-pixel perturbation fails. The added screen-pixel units are adjacent to the screen-pixel unit selected for the one-pixel perturbation due to the regional aggregation effect [138].

In one-pixel perturbation, the color of subpixel sets is selected based on the dominant color channel of the captured image. However, this strategy may not always be applicable, as the object of interest can be obscured by the color of the screen. For instance, when the screen displays a red status bar, the object of interest will be concealed by red screen-pixels, and makes it difficult to discern the original dominant color channel of the object. To solve this problem, we propose the *differential incremental module*, which estimates the impact of each individual R/G/B subpixel set on the adversarial strength of the generated perturbation.

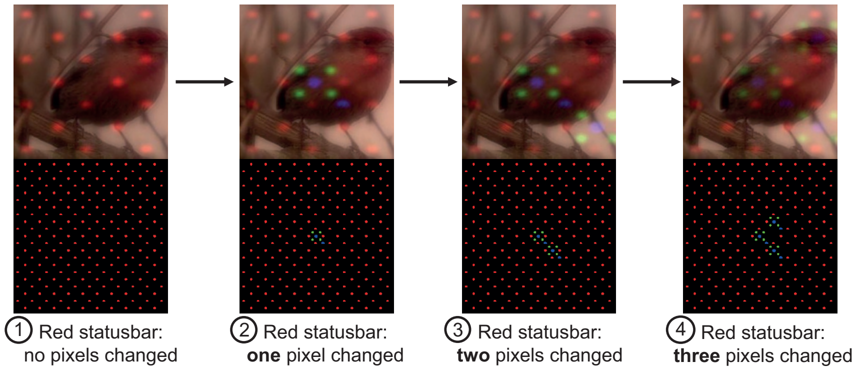

① Red statusbar: no pixels changed
② Red statusbar: **one** pixel changed
③ Red statusbar: **two** pixels changed
④ Red statusbar: **three** pixels changed

Figure 4.8: Example of pixel spread process: transitioning from no manipulated screen-pixel units to activate three screen-pixel units.

Specifically, we add a small increment to the brightness vector by $\boldsymbol{b}' = \boldsymbol{b} + \Delta\tau$ for each R/G/B subpixel set. Then, we select the colors of subpixel sets perturbation (c_1/c_2) by:

$$c_1/c_2 = \begin{cases} G/B,\ R = \arg\min_{c\in\{R,G,B\}}\left(\frac{\Phi(\boldsymbol{I}\oplus\boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}'},\hat{l})}{\Phi(\boldsymbol{I}\oplus\boldsymbol{z},\hat{t})}\right), \\ G/R,\ B = \arg\min_{c\in\{R,G,B\}}\left(\frac{\Phi(\boldsymbol{I}\oplus\boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}'},\hat{l})}{\Phi(\boldsymbol{I}\oplus\boldsymbol{z},\hat{t})}\right), \\ B/R,\ G = \arg\min_{c\in\{R,G,B\}}\left(\frac{\Phi(\boldsymbol{I}\oplus\boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}'},\hat{l})}{\Phi(\boldsymbol{I}\oplus\boldsymbol{z},\hat{t})}\right). \end{cases} \tag{4.11}$$

By adding the increment $\Delta\tau$ to the corresponding screen pixel color $\boldsymbol{c}$, we can identify the two subpixel sets that have the highest impact on the adversarial strength. Additionally, we use a second-order difference for brightness calculation, instead of the linear search used for the one-pixel perturbation. Specifically, we use the second-order difference of the probability score as the searching condition:

$$\Phi(\boldsymbol{I}\oplus\boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}''},\hat{l}) - \Phi(\boldsymbol{I}\oplus\boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}'},\hat{l}) < \Phi(\boldsymbol{I}\oplus\boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}'},\hat{l}) - \Phi(\boldsymbol{I}\oplus\boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}},\hat{l}), \tag{4.12}$$

where $\boldsymbol{b}' = \boldsymbol{b}+\Delta\tau$ and $\boldsymbol{b}'' = \boldsymbol{b}+2\Delta\tau$. If the second-order difference satisfies the above condition, we stop increasing the current screen-pixel unit's brightness and start calculating the next selected screen-pixel unit's position and brightness.

Finally, to ensure the changes in the multiple screen-pixel units are less perceptible by human visual system, we add a constraint on Equation (4.12) when searching $\boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}}$:

$$\text{DSSIM}(\text{Resize}(L(\boldsymbol{z}),\alpha),\text{Resize}(L(\boldsymbol{z}_{j,\boldsymbol{c},\boldsymbol{b}}),\alpha)) < \rho, \tag{4.13}$$

where DSSIM$(\cdot)$ calculates the structural dis-similarity index, a common measure of user-perceived image distortion [121], [139]. Resize$(\cdot)$ and $L(\cdot)$ represent image resizing and low-pass filtering functions, used to simulate the pixel density and viewing distance changes during smartphone use, and to determine the visual impact of screen pixel changes on human perception. Lastly, ρ denotes the perceptual perturbation budget. We leverage the downsampling scale α to simulate human visual perception, influenced by the viewing angle, screen resolution, and viewing distance [140], [141].

4.5. PERFORMANCE EVALUATION

4.5.1. METHODOLOGY

Goal and metrics. For simplicity, we evaluate our method from the attacker's perspective: the aim is to manipulate transparent screen region's screen-pixel units, generating perturbed images to fool different **target models**. The screen-pixel perturbation's performance is measured by the target model's accuracy in classifying perturbed images.

Applications & Target Models. We consider two widely used image-based applications.

- **Image classification**. We consider three deep learning architectures with six models: (1) **ResNet** [142], which includes ResNet-18 and ResNet-50; (2) **MobileNet** [143], which includes MobileNet V3 Large (MobileNet$_L$) and Small (MobileNet$_S$); and (3) **ShuffleNet V2** [144], which has two variants ShuffleNet V2 with half of the network parameters (ShuffleNet$_S$) and with all network parameters (ShuffleNet$_L$). All models are pre-trained on the ImageNet dataset [145].

- **Face recognition**. We consider two representative backbone models **IncepResNet V1** [146] (IncepResNet) and **MobileNet V2** [147] (MobileNet) that are pre-trained on the VGGFace2 dataset [148] and WebFace dataset [149], respectively.

Synthesized Datasets. We leverage the theoretical model to embed screen perturbations to images from two datasets. Specifically, we use the **miniImageNet** dataset [150], with 60,000 images across 100 classes, to evaluate the image classification task. We use the **FaceScrub** dataset [151], with 38,202 images from 530 people, to evaluate the face recognition task. We randomly select 500 and 530 images from miniImageNet and FaceScrub, respectively, to add screen perturbations. In simulation, our model employs a transparent screen region on OLED screens with a pixel density of 400 Pixels Per Inch (PPI), utilizing a circular R/G/B subpixel shape that completely fills the screen-pixel area. To ensure consistency with the USC in Commercial Off-The-Shelf (COTS) smartphones, we utilize camera parameters that are identical to those employed in commercial devices. We also incorporate peak wavelength values for the R, G, and B channels based on prior research [10], [14] with values of 0.61 μm, 0.53 μm, and 0.47 μm, respectively.

Dataset collected by testbed. We setup a testbed to acquire practical USC images. The setup is shown in Figure 4.9, which consists of a 4K LCD monitor displaying pristine images, and three COTS USC smartphones, i.e., Samsung Fold4, ZTE AXON30, and Xiaomi MIX4. The pixel density of the transparent screen region in these devices is 400 PPI. The size of the transparent screen region is 58×58 pixels, 108×60 pixels, and 108×60 pixels for the Fold4, AXON30, and MIX4, respectively. A custom Android application is developed to control the screen-pixel units for perturbation generation. The smartphone is positioned at the center of the monitor and is adjusted to cover the monitor's full range.

We then leverage high-resolution full-face images from the XGaze dataset [152] to generate perturbed images. Specifically, we select a subset of 12,720 images from 110 subjects, and randomly sample a total 549 images to add screen perturbations. The selected images are displayed on the 4K monitor in full-screen mode and adjusted to maintain the aspect ratio through rotation or resizing. As an example, Figures 4.9(b-d) shows the perturbed images captured by the three smartphones with one-pixel perturbation

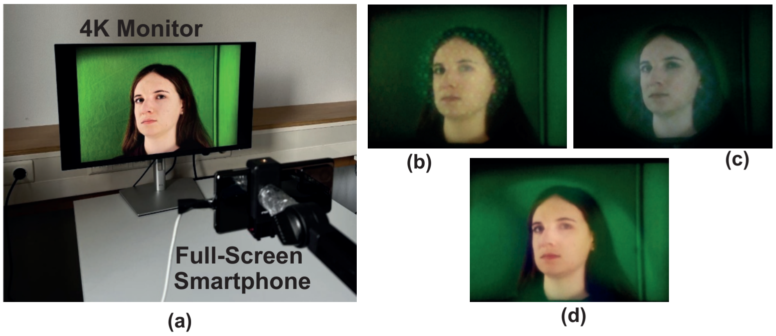


Figure 4.9: The testbed setup: we use COTS smartphones to generate images containing screen perturbations, and images captured by (b) ZTX AXON30, (c) Xiaomi MIX4, and (d) Samsung Fold4, respectively.

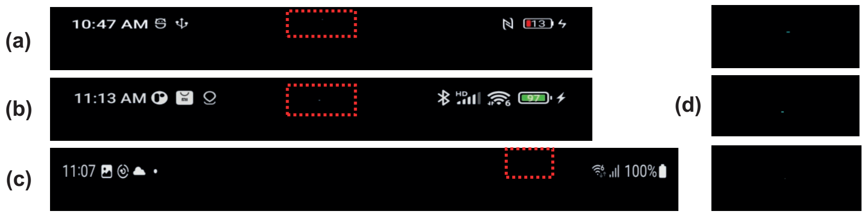


Figure 4.10: Screenshots of the smartphone status bar when running the one-pixel perturbation on (a) ZTE AXON30, (b) Xiaomi MIX4, and (c) Samsung Fold4. The transparent screen region is highlighted by the red dotted rectangle, and magnified in (d). These changes are imperceptible.

added. The resulting screen-pixel perturbations differ as different screen layouts. Moreover, Figure 4.10 shows the screenshots of the status bar when the three smartphones are generating one-pixel perturbation. As highlighted in the red dotted rectangle, the changes of screen-pixel units in the transparent screen region are imperceptible.

4.5.2. PERFORMANCE ON SYNTHESIZED DATASETS

Performance on image classification. We first investigate the effectiveness of screen perturbation in disrupting image classification on the miniImageNet dataset. We consider five different scenarios, including (1) pristine image with no screen perturbation, (2) passive perturbation when the status bar is black, (3) passive perturbation when the status bar is white, (4) passive perturbation with the additional active perturbation generated by one-pixel perturbation, or by (5) multiple-pixel perturbation. The results are shown in Table 4.1. Even with a black status bar, the accuracies of all models are decreased by 20% due to the passive perturbation. Furthermore, with a white status bar, the accuracy of all models decreased by 30%, due to the uncontrolled active perturbation. In comparison, when applying the one-pixel and multiple-pixel perturbation, the accuracies of all examined models are degraded below 20% and 10%, respectively. The results demonstrate the effectiveness of the proposed screen-pixel perturbation in disrupting image classification.

Table 4.1: Performance of both one-pixel and multiple-pixel perturbations on miniImageNet.

Target model	Pristine image	Black screen Blurred/Deblurred	White screen Blurred/Deblurred	One-pixel Blurred/Deblurred	Multiple-pixel Blurred/Deblurred
ResNet-18	87.4%	62.4% / 86.8%	56.8% / 75.2%	**16.6%** / **20.6%**	**5.0%** / **7.6%**
ResNet-50	94.2%	74.4% / 94.6%	66.8% / 86.2%	**20.8%** / **26.8%**	**5.8%** / **9.2%**
MobileNet$_S$	82.4%	58.8% / 81.6%	50.6% / 70.2%	**10.2%** / **14.8%**	**5.4%** / **10.0%**
MobileNet$_L$	91.6%	69.8% / 90.8%	66.2% / 79.2%	**13.4%** / **17.4%**	**6.4%** / **11.4%**
ShuffleNet$_S$	71.0%	49.8% / 70.2%	29.6% / 46.0%	**7.2%** / **9.0%**	**4.6%** / **5.4%**
ShuffleNet$_L$	85.4%	64.4% / 85.2%	47.8% / 62.2%	**16.0%** / **21.0%**	**5.8%** / **6.2%**

Performance against deblurring algorithms. We also evaluate the performance when state-of-the-art deblurring method is applied. Specifically, we use the unsupervised Wiener filter to alleviate the blurring effect caused by screen passive perturbation [10], [14]. The results are shown in Table 4.1. The deblurring method can effectively eliminate the impact from the passive perturbations, i.e., the accuracy of all examined models is increased after deblurring when there is only passive perturbation on the image. However, the deblurring method does not help when one-pixel or multiple-pixel perturbations have been applied, i.e., only a modest 5% accuracy improvement after applying the deblurring method.

Transferability. The perturbation should be effective even when the target model is different from the surrogate model. Current work [131] suggests that the transferability of perturbation between models depends on the robustness of the surrogate model used to create it, and more robust surrogate models are less reactive to small perturbations. Thus, to ensure the transferability of the screen perturbations, we first retrain the surrogate model using perturbed images generated from a white status bar. This provides the surrogate model with exposure to fixed screen perturbation. We then use the updated surrogate models to generate multiple-pixel perturbations on the miniImageNet dataset. The results shown in Table 4.2 demonstrate that the multiple-pixel perturbations transfer almost perfectly across different models.

Ablation Study. First, we investigate the impact of the maximum screen-pixel intensity budget (ε) on the performance of the one-pixel perturbation attack (cf. Equation (4.10)). The initial intensity of the screen-pixel is 0.01 and the step size of each iteration is 0.01. Then, we vary the maximum screen-pixel intensity constraint from 0 to 1. The results are shown in Figure 4.11. the accuracy gradually decreases with the increase of the maximum screen-pixel intensity constraint. Even when the maximum screen-pixel intensity is limited to 0.2, the accuracy still drops by more than 30%.

Next, we examine the step size ($\Delta\tau$) effect on the screen-pixel intensity search (cf. Equation (4.10)), determining the optimal intensity for one-pixel perturbation. We fix the maximum screen-pixel intensity to 1 and the initial screen-pixel intensity to 0.01, and then vary the step size. Results in Figure 4.12 show image classification accuracy gradually increases with the step size, as a larger step size complicates finding the optimal intensity. To reduce the computational overhead of calculating one-pixel and multiple-pixel perturbations, we employ a step size of 0.01 in the following experiments.

We also investigate the impact of initial screen-pixel intensity (b_0) (cf. Equation (4.8))

Table 4.2: Performance (attack transferability) of multiple-pixel perturbations on miniImageNet.

Robust Surrogate Model	Target Model					
	ResNet-18	ResNet-50	MobileNet$_S$	MobileNet$_L$	ShuffleNet$_S$	ShuffleNet$_L$
	Blurred/*Deblurred*	Blurred/Deblurred	Blurred/Deblurred	Blurred/Deblurred	Blurred/Deblurred	Blurred/Deblurred
ResNet-18	**4.8%** / **7.4%**	6.8% / 12.2%	5.6% / 7.8%	7.8% / 12.2%	4.2% / 4.2%	7.4% / 8.8%
ResNet-50	5.0% / 7.6%	**5.8%** / **9.2%**	5.4% / 10.0%	6.4% / 11.4%	4.6% / 5.4%	5.8% / 6.2%
MobileNet$_S$	6.4% / 11.6%	7.8% / 16.4%	**4.0%** / **9.2%**	8.4% / 16.2%	4.6% / 7.4%	6.8% / 11.0%
MobileNet$_L$	6.4% / 7.6%	5.0% / 10.6%	3.4% / 6.8%	**6.8%** / **10.2%**	4.4% / 5.0%	6.4% / 7.8%
ShuffleNet$_S$	8.2% / 12.8%	11.2% / 23.2%	7.4% / 14.6%	11.6% / 22.6%	**7.4%** / **8.8%**	9.0% / 15.6%
ShuffleNet$_L$	7.4% / 9.8%	9.4% / 16.0%	6.4% / 11.2%	8.2% / 14.4%	6.2% / 7.4%	**6.8%** / **9.8%**

Table 4.3: Performance of both one-pixel and multiple-pixel perturbations on Facescrub dataset.

Model	Pristine image		One-pixel			Multiple-pixel	
		Black screen	IncepResNet	MobileNet	White screen	IncepResNet	MobileNet
		Blurred/Deblurred	Blurred/Deblurred	Blurred/Deblurred	Blurred/Deblurred	Blurred/Deblurred	Blurred/Deblurred
IncepResNet	90.57%	50.75% / 85.09%	**0.00%** / **0.76%**	0.38% / 0.38%	38.30% / 72.64%	**0.00%** / **0.00%**	0.00% / 0.00%
MobileNet	84.53%	26.04% / 81.89%	0.19% / 0.76%	**0.57%** / **0.76%**	19.25% / 58.11%	0.00% / 0.00%	**0.00%** / **0.00%**

Table 4.4: Performance of both one-pixel and multiple-pixel perturbations on XGaze dataset.

Model	Pristine image	Black screen	One-pixel		White screen	Multiple-pixel	
			IncepResNet	MobileNet		IncepResNet	MobileNet
IncepResNet	94.72%	81.06%	**4.01%**	2.37%	41.53%	**0.36%**	0.91%
MobileNet	94.54%	70.13%	2.37%	**1.28%**	34.43%	0.18%	**0.55%**

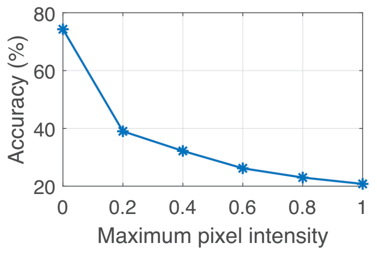


Figure 4.11: Accuracy vs. max intensity (ε).

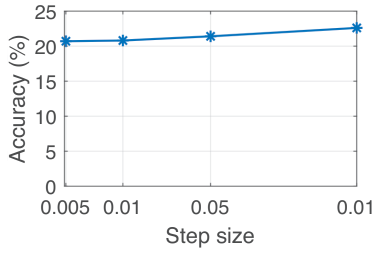


Figure 4.12: Accuracy vs. step size ($\Delta\tau$).

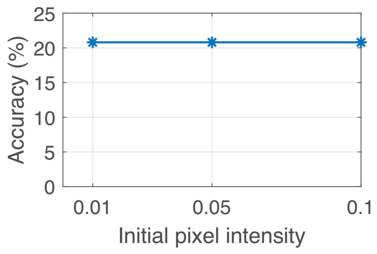


Figure 4.13: Accuracy vs. initial intensity (b_0).

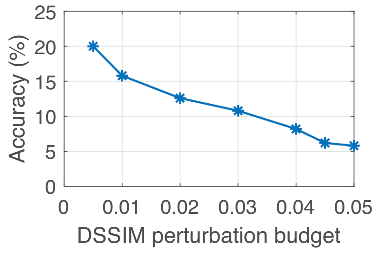


Figure 4.14: Accuracy vs. DSSIM budget (ρ).

on the results of one-pixel perturbation, by fixing the maximum pixel intensity and step size to 1 and 0.01, respectively, and varying the initial intensity of the screen-pixel. As shown in Figure 4.13, the variation in initial intensity has a negligible impact on the final performance. We chose to use a smaller initial intensity of 0.01.

Finally, we evaluate the impact of the DSSIM in multiple-pixel perturbation (cf. Equation (5.4)). The results are shown in Figure 4.14. As the DSSIM perturbation budget (ρ) increases, the accuracy decreases. Specifically, when ρ is set to 0.05, the accuracy drops to the same level as that of one-pixel perturbation, at around 20%. Furthermore, when ρ exceeds 0.03, the accuracy of multiple-pixel perturbation falls below 10%. Existing work [153] suggests that higher DSSIM values (up to 0.2) are imperceptible to human eyes.

Performance on face recognition using FaceScrub. We also evaluate our method in the face classification task using the FaceScrub dataset. The results are shown in Table 4.3. The passive screen perturbation leads to a substantial accuracy degradation of up to 40% and 60% for IncepResNet and MobileNet, respectively. Both one-pixel and multiple-pixel perturbation can decrease the accuracy of all models to less than 1%. Moreover, we also apply the deblurring method to mitigate the adverse impact of the screen perturbations. The deblurring method can mitigate the impact of passive screen perturbations, but fails to remove the impact of the proposed one-pixel and multiple-pixel perturbations.

4.5.3. Performance on Testbed Dataset

Overall performance. Table 4.4 shows that the proposed one-pixel perturbation can reduce the accuracy of both IncepResNet and MobileNet to less than 5%. The multiple-pixel perturbation further reduces the accuracy of both IncepResNet and MobileNet to under 1%, demonstrating the effectiveness of our proposed screen perturbation meth-

Table 4.5: The PSNR and SSIM when one-pixel and multiple-pixel perturbations are embedded in the image.

Metrics	One-pixel	Multiple-pixel
PSNR	23.72 dB	21.50 dB
SSIM	0.84	0.80

ods in real-world scenarios. The results in Section 4.5.2 and Section 4.5.3 align with each other, as shown in Table 4.3 and Table 4.4.

Different color in status bar. We also evaluate the multiple-pixel perturbation with four different status bar colors: *red, blue, green, white*. Note that when the status bar is displayed in red, green, or blue, only the corresponding primary color subpixel sets are illuminated. When the status bar is white, all three primary color subpixel sets are illuminated. The results are presented in Figure 4.15. Without active perturbation, we can achieve 40% to 50% of accuracy in all four background colors. With the multiple-pixel perturbation, the face recognition accuracy drops to 1% for all four colors.

Screen Diversity. We evaluate one-pixel perturbation with different smartphones. The results are shown in Figure 4.16. In the absence of screen-pixel perturbation and relying solely on passive perturbation, the performance of images captured by under-screen cameras from different smartphone manufacturers is relatively similar, hovering around 80%. However, after adding one-pixel perturbation, the classification performance of under-screen camera captured images dropped below 20%. Specifically, the classification accuracy of ZTE AXON30 declined the most, falling below 5%, while that of Samsung Fold4 decreased the least. This is because the pixel size of the Samsung transparent screen region is too large, causing the screen-pixel perturbation on the under-screen camera captured image to be too diffuse, thereby reducing the perturbation intensity of the attacking region in the USC-captured images.

Image Quality. We further evaluate the impact of screen-pixel perturbation on image quality by comparing a set of unperturbed images with the corresponding perturbed images, all captured using the ZTE AXON30. We use two widely adopted metrics: Peak Signal-to-Noise Ratio (PSNR) [96] and Structural Similarity Index Measure (SSIM) [154]. Specifically, PSNR quantifies the variations in individual pixel intensity levels, while SSIM assesses the structural distortions within an image, e.g., stretching, banding, and twisting. The PSNR typically ranges from 0 to 50 dB, where a higher PSNR indicates greater similarity between the unperturbed and perturbed images. The value of SSIM ranges from 0 to 1, with 1 denoting minimal quality loss between the unperturbed and perturbed images. The results are shown in Table 4.5. For images containing one-pixel perturbations, the PSNR and SSIM are 23.72 dB and 0.84, respectively. For images added with multiple-pixel perturbations, the PSNR and SSIM are 21.5 dB and 0.8, respectively, as more screen-pixel perturbations have been added to the captured images. Note that for images with acceptable viewing quality, the minimum PSNR and SSIM are in the range of 20~40 and 0.8~0.9, respectively [155], [156]. Thus, the perturbed images have acceptable image quality.

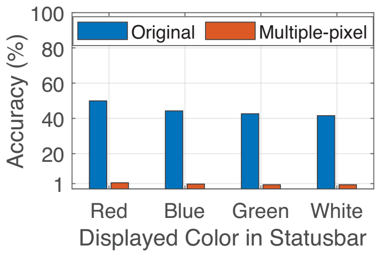


Figure 4.15: Accuracy vs. status bar color.

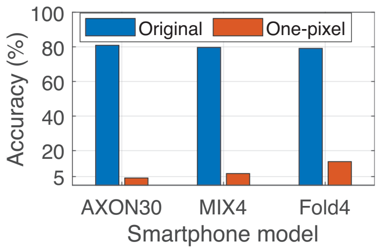


Figure 4.16: Accuracy vs. smartphone model.

4.5.4. User Study

Below, we conduct a user study to investigate if the added perturbation is visible in images captured by the under-screen cameras. We also examine if and how changes in the smartphone transparent screen region, i.e., added single-pixel and multi-pixel screen perturbations, affect the user's experience with smartphone use. We recruit 30 participants from our university (13 female and 17 male, aged between 20 and 45) to conduct a user study. We advertised our study through university mailing lists, social networks, and advertising boards in our department building. All participants had little or no experience with USC smartphones. Ethical approval for the user study has been granted by our organization. The entire study session took 30 minutes on average, and we provided each participant with a 10 euro gift card as a token of appreciation for participating.

Study design. After arriving at the lab, participants are told a brief introduction to the study session. Participants sit down at a fixed position and the study session begins. We design two tasks in the study.

- **Perturbed images task**. We investigate if users perceive the screen perturbation added to images. Using the three USC smartphones (cf. Section 4.5.3), Samsung Fold4, ZTE AXON30, and Xiaomi MIX4, we prepare two image sets: one-pixel and multiple-pixel paired sets. We randomly select ten different subjects from the XGaze [152] dataset each time. In the one-pixel paired set, each smartphone captures ten *unperturbed images* without adding any screen perturbation and ten *perturbed images* with one-pixel perturbation from these subjects. The multiple-pixel paired set consists of ten *unperturbed images* captured by AXON30 when the status bar is in red, green, blue, or white, and ten *perturbed images* captured by AXON30 with added multiple-pixel perturbation. Images in two image sets are displayed on a 27-inch monitor in a random order, and each image is played for 10 seconds.

  Results. Participants answer two 5-point Likert scale questions (1: strongly disagree; 5: strongly agree) after viewing each displayed image:

  - Q1: *Can you see a person in this image?*
  - Q2: *Do you think there is a perturbation in this image?*

  A standard two-sample t-test [157] is conducted to compare user visual perception for *unperturbed* and *perturbed* image sets. The null hypothesis (H_0) posits no true difference between the means of the two sets, while the alternate hypothesis (H_a)

Table 4.6: T-test results of the one-pixel paired set.

Device	AXON30		MIX4		Fold4	
Question	Q1	Q2	Q1	Q2	Q1	Q2
p	0.8822	2.83×10^{-7}	0.3416	2.67×10^{-5}	0.3367	0.0545
Results	H_0	H_a	H_0	H_a	H_0	H_0

suggests a non-zero difference. Setting a significance level $\alpha = 0.05$, we calculate p to get the t-test result, where p is the likelihood that the observed difference is occurred by chance. If $p > \alpha$, we accept H_0 and conclude that participants have similar responses on both sets. Otherwise, we reject H_0 at the significance level of 0.05, and conclude that participants have different responses on the two sets. The t-test results are shown in Tables 4.6 and 4.7. H_0 is all accepted in Q1, showing similar user perception for seeing a person in all unperturbed and perturbed images. In Q2, images captured by Fold 4 accept H_0, suggesting less perceptible one-pixel perturbation, while images from AXON30 and MIX4 reject it, indicating more visible perturbations due to their different transparent screen region designs. Recall that different transparent screen region designs of smartphones result in diverse one-pixel perturbations (cf. Figure 4.9). However, images captured with different status-bar colors accept H_0, suggesting participants cannot perceive the added multiple-pixel perturbation since the color of the status bar has added a fixed perturbation to captured images.

- **Smartphone usage task**. Next, we investigate the perceptibility of screen-pixel changes in the transparent screen region and their impact on user experience during typical smartphone usage. We adopt the three USC smartphones used (cf. Section 4.5.3), and randomly assign ten participants to use each of the smartphones. We developed two smartphone apps, i.e., *image app* and *text app*, for image and text viewing. To reduce individual differences in visual perception of screen perturbations, we conduct a within-subject study [158] and consider two screen settings: (1) *with screen perturbation*, and (2) *without screen perturbation*. We ask the participant to use the two developed apps on the assigned smartphone, and each app is used for two minutes. We configure the app with both screen settings and load them in random order, thus ensuring one-minute usage for each screen setting. We consider both one-pixel and multiple-pixel perturbations in the experiment and modify the screen pixel in the transparent screen region accordingly.

  Results. We provide a short questionnaire to the participants, and ask them if they have noticed any screen perturbation during the use of the apps. The results show that none of the 30 participants notice the added one-pixel or multiple-pixel screen perturbation. We also conduct a standard two-sample t-test [157] to investigate the visual perception of the user in the two screen settings, i.e., with and without screen perturbation added. The result shows that H_0 is accepted with $p > 0.99$, which indicates that two screen settings on *image app* and *text app* have the same visual perception.

Table 4.7: T-test results of the multiple-pixel paired set.

Color	Red		Green		Blue		White	
Question	Q1	Q2	Q1	Q2	Q1	Q2	Q1	Q2
p	0.1326	0.4063	0.7063	0.8051	0.6367	0.9484	0.3007	0.2298
Results	H_0	H_0	H_0	H_0	H_0	H_0	H_0	H_0

Table 4.8: Accuracy of one-pixel and multiple-pixel perturbations with adversarial training on XGaze.

Model	Pristine image	One-pixel		Multiple-pixel	
		IncepResNet	MobileNet	IncepResNet	MobileNet
IncepResNet	85.79%	**7.29%**	4.19%	**3.83%**	1.82%
MobileNet	81.42%	8.20%	**4.92%**	4.00%	**1.64%**

4.5.5. COUNTERMEASURES

Below, we investigate possible countermeasures against the proposed method. Many defenses [159]–[164] have been proposed to defend against adversarial perturbations. Depending on whether those defenses have formal robustness guarantees, we can categorize them into *empirical defenses* [159], [161], [162] and *certified defenses* [163]–[165]. We consider state-of-the-art defenses from both categories. For empirical defenses, we consider adversarial training [161] as it is viewed as one of the most effective empirical defenses. For certified defenses, we consider randomized smoothing [164] as it is applicable to any classifier and scalable to large deep learning models.

Adversarial training [159], [161] leverages adversarially perturbed training examples to train a model to enhance its robustness. First, we generate adversarial examples using the Project Gradient Descent (PGD) attack [161]. We employ an L_∞-based PGD attack on the XGaze dataset, with a maximum distortion of 0.1 and an untargeted attack mode. We run the PGD attack for 50 steps with a step size of 0.01. We train IncepResNet-V1 and MobileNet-V2 for 200 epochs with a learning rate of 0.1, decayed by 0.1 after 100 and 150 epochs, respectively [161], [166]. Subsequently, we evaluate the performance of the resulting models against one-pixel and multiple-pixel perturbations using the XGaze dataset [152]. The results are shown in Table 4.8, with adversarial training, the classification accuracy of the two deep learning models has still degraded below 8% and 4% when one-pixel and multiple-pixel perturbations have been added.

Moreover, we also perform adversarial training with a dataset that comprises pristine images paired with corresponding perturbed images containing one-pixel perturbation. The results are presented in Table 4.9. When comparing with the results reported in Table 4.4, we can observe a decline in classification accuracy when perturbed images are utilized for training. However, the accuracy is below 10% and 5%, respectively, when the captured images contain one-pixel or multiple-pixel perturbations. Overall, the results indicate that adversarial training is not sufficient to mitigate the perturbations added by the proposed method. While this doesn't conclusively negate the possibility of developing robust models, it indicates the inherent challenges in such pursuits.

Randomized smoothing. Given an arbitrary classifier H (called *base classifier*) and a testing input $\mathbf{x}$, randomized smoothing builds a certifiably robust smoothed classifier G

Table 4.9: Accuracy of proposed perturbations when adversarial training is enhanced with perturbed images.

Model	Pristine image	One-pixel		Multiple-pixel	
		IncepResNet	MobileNet	IncepResNet	MobileNet
IncepResNet	87.07%	**10.56%**	5.83%	**4.37%**	2.00%
MobileNet	86.52%	10.20%	**5.28%**	1.82%	**2.37%**

Table 4.10: Accuracy of one-pixel and multiple-pixel perturbations under randomized smoothing on XGaze.

Model	Pristine image	One-pixel		Multiple-pixel	
		IncepResNet	MobileNet	IncepResNet	MobileNet
IncepResNet	93.99%	**6.01%**	3.46%	**0.73%**	0.36%
MobileNet	94.90%	6.19%	**3.64%**	1.09%	**0.91%**

by adding zero-mean isotropic Gaussian noise $\mathcal{N}(0,\sigma^2\mathbf{I})$ to the testing input $\mathbf{x}$, where σ is the standard deviation and $\mathbf{I}$ is the identity matrix. Formally, the predicted label of the smoothed classifier G for the testing input $\mathbf{x}$ is $G(\mathbf{x}) =_{c=1,2,\cdots,C} \Pr(H(\mathbf{x}+\mathcal{N}(0,\sigma^2 I)) = c)$, where C is the total number of classes. Existing work [164] shows that the predicted label of the smoothed classifier G for the testing input $\mathbf{x}$ does not change when the adversarial perturbation added to $\mathbf{x}$ is bounded. To compute $G(\mathbf{x})$ in practice, randomized smoothing first adds random Gaussian noise to the testing input $\mathbf{x}$ to create M noisy versions of the testing input, then use the base classifier H to predict labels for those M noisy inputs, and finally take a majority vote over the M predicted labels as the final prediction for the testing input. Following previous work [164], we set $\sigma = 0.5$ and $M = 10^5$. Moreover, we train the base classifier on training inputs augmented with Gaussian noise to improve the robustness of the smoothed classifier. Table 4.10 shows our experimental results, which show that the classification accuracy of the smoothed classifier built by randomized smoothing is still very low. The reason is that randomized smoothing can only certify a very small perturbation. Our results demonstrate that randomized smoothing is insufficient to mitigate our proposed one-pixel and multiple-pixel perturbations.

4.6. RELATED WORK

Image processing for the under-screen camera. There are efforts in modeling and restoring images that are affected by passive perturbations caused by the transparent screen [10]–[12], [14], [114]. Specifically, Zhou et al. [10] and Yang et al. [14] modeled the passive perturbation of the screen and proposed an unsupervised Wiener deconvolution method. Moreover, they employed deep neural networks to address the issues of large blur and low signal-to-noise ratio in under-screen images [10]–[12]. Feng et al. [167] and Gao et al. [168] explored simulation pipelines to generate synthetic datasets with real-captured passive perturbation. While existing studies focused solely on passive perturbations, we are the first to consider the active perturbations caused by the screen displays. We apply the Huygens-Fresnel principle to model the impact of different color pixels and screen parameters on image formation. This allows us to analyze the impact of active screen perturbations on deep neural networks.

Programmable aperture. Our work is related to the concept of programmable aperture that has been implemented in some new types of cameras such as DiffuserCam [169] and FlatCam [170]. Existing works of programmable aperture leverage amplitude or phase masks to code the aperture of a camera lens, and operate at scales that are larger than screen pixels. This is different from the under-screen camera apertures we consider, which coexist with R/G/B OLED arrays and can be dynamically manipulated to generate screen-pixel perturbation when the transparent screen region is illuminated.

Adversarial attacks can be categorized into digital space attacks [161], [171], [172] or physical space attacks [119], [173]–[175]. Digital space attacks directly change pixel values in the digital pixel domain, typically using methods such as PGD [161] and C&W [172]. However, they may not generalize to real-world scenarios due to the constraints present in the physical environment. Physical space attacks, such as those utilizing graffiti [173], rectangular stickers [174], eyeglasses [119], or LED lamps [176], [177], have been explored for privacy protection but are often limited by artificial settings or hardware modifications. By contrast, ours is the first to generate adversarial perturbation on USC-captured images utilizing both passive and active screen perturbation of full-screen devices. Our threat model is unique in that we inject a perturbation into the optical path between the camera and the object, disrupting any object without physically tampering with the object itself.

4.7. CONCLUSION

We show for the first time the screen can be used for adversarial attack and defense on Under-Screen Camera (USC). Directly modifying captured images is challenging for attackers, as it requires write access from smartphone OS to a set of media files [178]. However, attacking the optical path (camera input) is relatively easier, as Android does not have permission restrictions on UI settings, allowing any app to set the screen content and potentially perturb camera's input. We identify the activated screen-pixel that can be exploited for applying perturbation on images captured by USC. In typical smartphone usage scenarios, successive frames captured by the camera are similar. Thus, we only need to compute the proposed screen-pixel perturbation once, rather than for each image separately, making the attack more feasible. We derive an imaging formation model for the USC, which facilitates the generation of screen-pixel perturbations on synthesized datasets. We design and implement a method to successfully fool different deep learning models. We believe this is a pioneer work which can stimulate a lot of follow-up works either to attack or safeguard current USCs on full-screen mobile devices.

Future work. A potential defensive strategy is to prohibit a single application from rendering content on the screen and accessing the camera simultaneously. However, this approach is inconvenient for users, especially when using video communication applications that require camera access alongside other applications. In fact, many modern smartphones, e.g., Samsung Fold 3/4, ZTE AXON 20/30/40, Xiaomi MIX4, and K50S, are designed to support simultaneous operation of one app's camera usage while another app utilizes the screen. The current system design in this chapter requires a pristine image to compute the attacking region for the screen perturbation. This image can be obtained from many sources, e.g., selfies of the subject published on social media, and does not need to be taken in run-time. For instance, with the aim to defend against unauthorized face recognition systems, users can take a selfie in run-time or upload a previously taken selfie to compute the attacking region. Since the face of the subject is always located in the center of the selfie, it leads to similar attacking regions as long as the image contains the same subject. Similarly, in the attacking scenario, malicious parties can find and leverage the selfie of the victim published on social media to achieve the same purpose. In the next chapter, we can release the requirement of a pristine image by designing a universal and scene-independent perturbation.

5 UNICORN: SECURING USC WITH UNIVERSAL SCREEN PERTURBATION

George Orwell's novel *1984* introduced the Telescreen, a surveillance device transmitting visuals and sounds to monitor people's actions and thoughts. This fiction mirrors reality today as front-facing cameras on mobile devices pose privacy risks, i.e., *Camfecting* [44]. A lot of sensitive information is often at risk of exposure in typical camfecting scenarios, as illustrated in Figure 5.1. While manually analyzing a large volume of captured images to find needed sensitive information in the past is challenging, hackers can now employ advanced deep neural networks for efficient extraction of critical and sensitive information from these images [48], [49]. A significant risk is facial data exposure through front-facing cameras [45]. Users' regular interactions with devices make facial data vulnerable to misuse for identification, location tracking, and severe threats like stalking or extortion. Another incident of such sensitivity included a password note captured in a photograph [46]. One instance saw hackers extracting a password from an employee's office selfie at TV station TV5Monde, leading to signal disruptions in 11 stations [47]. Furthermore, a survey by Keeper Security [179] also highlights the severity of this issue, showing that 57% of American employees record work-related passwords on sticky notes, a trend that has escalated in the post-COVID-19 remote work environment.

Traditional physical isolation countermeasures, like sticking tape over the camera, compromise user experience. The emergence of full-screen devices with transparent screen regions presents an opportunity to develop innovative, non-intrusive privacy safeguards. In this chapter, we aim to explore new privacy protection mechanisms - 'covering your camera with a transparent screen, instead of tape' - that are user-friendly and effective, redefining privacy standards in the modern age without diminishing the user experience. Hence, we ask the following research question:

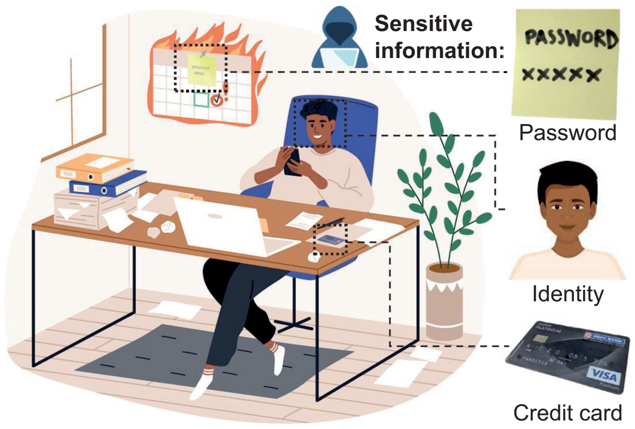


Figure 5.1: Illustration of a typical office scenario highlighting sensitive information to exposure through camfecting, such as facial data, password notes, and credit card information.

How to deploy a visually sensitive information protection tool on full-screen devices without affecting the normal use of users?

In this chapter, we propose *Unicorn*, which utilizes the transparent screen region above the Under-Screen Camera (USC) as a dynamic shield. The design of Unicorn is based on our key observation that the translucent screen region can carefully activate specific *screen patterns* and generate a kind of image-agnostic perturbations (which we refer to as “*universal screen perturbations*”) on USC captured images that can fool hackers’ machine learning models from accurately recognizing user identity through any facial images and mining password notes or other sensitive content. This technique provides an imperceptible, yet effective defense against camfecting, preserving user privacy without affecting the user experience.

Why universal perturbations? Universal perturbation is able to fool deep neural networks on any image with high probability. Universality refers to the property of a perturbation being image-agnostic, as opposed to having good transferability. A fundamental property of existing image perturbations is their intrinsic dependence on a pristine input image: the perturbations are specifically crafted for each input image independently. Consequently, generating a perturbation for a new input image requires solving a data-dependent optimization problem from scratch. Therefore, the proposed screen perturbation method in Chapter 4 is a typical image-specific method and is not suitable for practical privacy protection scenarios. It requires a pristine image to generate the screen perturbation for each scene and object, making it ineffective for real-world dynamic scenarios. This is different from the universal perturbation considered in this chapter, as we seek a single perturbation vector that fools the network on most unseen images in a sensitive class or multiple sensitive classes. Perturbing a new input image then only involves the mere addition of the universal perturbation to the image, without solving an optimization problem or computing gradients again. We focus on developing a universal, scene-independent screen perturbation without the need for a pristine image. Universal screen perturbations offer greater scalability and generalizability compared to image-specific screen perturbations, making them more suitable for real-world physical perturbations and deceiving classifiers in diverse contexts.

Challenges and Contributions. The universal perturbation can be deployed as a background process on devices to effectively and consistently preserve user privacy at all times. It remains imperceptible to the user while successfully preventing hackers' neural networks from accurately recognizing unseen images containing diverse sensitive information. The image-specific screen perturbation method introduced in Chapter 4 employs a heuristic search algorithm to empirically test which screen-pixel has the most impact on neural networks' final decisions for a given input image [180]. However, without the real-time pristine image's input, such an empirical trial-and-error method is impractical for generating universal screen perturbations. Thus, the first challenge is

Challenge 1: *How to design universal perturbations without a pristine image input?*

The core of Unicorn is a novel optimization framework that learns latent features from the data distributions of multiple sensitive classes that need protection, enabling the perturbations to work universally across unseen images belonging to these sensitive information classes. Our design incorporates three novel loss functions to meet three key requirements: (1) the *post-processing loss* to counter potential image manipulations by hackers; (2) the *visual loss* to minimize the impact on user experience and ensure the perturbations remain imperceptible; and (3) the *pixel-level energy loss* to reduce power consumption, making the solution energy-efficient. A naive approach would involve optimizing these losses to create a digital perturbation and letting screens reproduce it on USC images. However, realizing the digital perturbation on a physical screen is close to impossible, as *the digitally optimized perturbations require perfect pixel-level precision and changes on the captured image, e.g., altering one specific pixel in the image from* $[200,200,200]$ *to* $[201,201,201]$*)*, which, in practice, state-of-the-art smartphone screens cannot reliably produce at such precision. This limitation arises from the complex interplay between the optical synthesis of *information in the target scene*, the *screen configurations*, and the *USC's response*.

To make screens generate universal perturbations in a physically realizable way and meet the optimization requirements as well, we need to tackle multiple challenges. Screens in different commodity devices usually have different configurations, e.g., layouts, shapes, and positions of screen-pixels. These specialized designs introduce unique perturbations to USC images. Such variations pose a severe challenge in accurately representing the perturbations induced by various screens. Hence, the second challenge is:

Challenge 2: *How to represent screens from different manufacturers?*

We observe that screens from different manufacturers are tiled by a *minimum screen pixel unit*. Hence, we use reverse engineering to learn this unit and then compose the screen model with the learned unit. By simply substituting the learned unit, the screen model can adapt to different manufacturers, enhancing its versatility. While this screen model can accurately recreate practical screen configurations on USC devices, we notice a significant discrepancy between the modeled screen perturbations and the actual perturbations observed in USC images. To uncover the root causes of this gap, we conducted an in-depth analysis and identified two key factors. First, each USC employs a unique Bayer filter [17], resulting in distinct color responses. Second, variations in the screen's finite pixel openings and the USC's lens aperture jointly affect how screen light diffracts and scatters. These factors introduce complex optical variations that are not fully captured by the initial screen model. This leads to our third challenge:

Challenge 3: *How to bridge the gap between the screen model and the practical universal screen perturbations on the USC image?*

To develop a deep learning-based computational architecture to bridge the gap between the screen model and the final USC image, encapsulating both active and passive perturbations from screens. The active generator simulates the effects of different intensities, colors, and positions of screen pixels on USC images. The passive generator models the screen's impact on USC imaging, including scenarios with inactive screen pixels. This comprehensive, differentiable model facilitates gradient propagation during perturbation crafting. To protect sensitive information in diverse environmental conditions, we propose a training approach to strengthen Unicorn's real-world robustness. Experimental results show that Unicorn can achieve over 90% protection against image classification under various hacker controls and shooting scenarios. We further achieve 100% success against advanced image classification services from Google (Google Vision API [50]) and OpenAI (ChatGPT Vision API [51]). We summarize key contributions:

- We propose Unicorn, a novel USC defense vector in the physical world using a transparent screen, offering flexibility, unobtrusiveness, and energy efficiency.
- We propose a neural USC architecture to craft universal screen perturbation. The method models a tripartite relationship between sensitive information, screen configurations, and USC response. We enhance the robustness by accounting for varying environmental conditions during the optimization process.
- We evaluate the Unicorn in real world with two smartphones, demonstrating its robust performance across diverse USC settings and environmental conditions.
- Our user study demonstrates that the screen-pixel changes are imperceptible during normal device usage, ensuring a imperceptible experience. Besides, Unicorn is also energy-efficient, only increasing screen power consumption by 0.14 mW (i.e., 0.024%) of typical smartphone usage.
- Finally, we demonstrate that Unicorn maintains 90+% success rates against a variety of perturbation disruptions from both empirical and certified perspectives.

5.1. CHALLENGES ON PRACTICAL PERTURBATION DEPLOYMENT

5.1.1. FROM DIGITALITY TO PHYSICALITY

To deceive the Deep Neural Networks (DNNs) used by hackers, we need to introduce an image perturbation technique. Recent studies have shown that DNNs are susceptible to carefully constructed small noise called image perturbations, which, when added to an input image, cause the network output to change drastically [181], [182]. These perturbations, crafted by exploiting the gradient information of DNNs, are added to digital images to deceive DNNs. However, their real-world effectiveness is limited since digital perturbations are often too subtle to be detected by USCs. For instance, printed images with perturbations yield the same recognition results as those without perturbations [181]. Moreover, even if an image is digitally perturbed, it retains *traces* (as a pristine version exists somewhere), posing a potential privacy leakage risk. Therefore, it is crucial to ensure that images containing the sensing information are safeguarded *from the moment they are created*. Consequently, we need a more robust physical perturbation to secure sensitive information before the digital image is output. Screens in USC devices can introduce the powerful physical perturbations necessary during image acquisition. To protect the sensitive information of the user, our Unicorn techniques not only need powerful *physically-realizable* perturbations necessary during image acquisition, but also need to meet the requirements of *universality*.

5.1.2. FROM IMAGE-SPECIFIC TO IMAGE-AGNOSTIC

For existing image perturbation techniques [175], [180], [182]–[185], the perturbations generated are different for each image, meaning a separate optimization must be performed for each image to generate a perturbed version. This generation of *image-specific perturbations* is hence also called *per-instance generation*. In our case, image-specific perturbations vary for different samples in a dataset and do not apply to practical scenarios [174], [175], [180], [183], [185], [186] since they need access to the pristine image to compute the screen perturbation. However, at that point, the image has been taken already and it would be too late for the computed perturbation to be realized on screen. Hence, we need to design *image-agnostic perturbations* called *universal screen perturbations* that even can fool advanced DNNs on many unseen images with high probability. Since only a single perturbation vector is needed to fool all images, they are much more efficient in terms of computation time and cost compared to image-specific perturbations. Unicorn focuses on developing a scene-independent, universal screen perturbation without the need for a pristine image. Unicorn is optimized for robustness and effectiveness in various conditions, including different shooting distances, angles, and USC settings. Universal screen perturbations offer greater scalability and generalizability compared to image-specific perturbations, making them more suitable for real-world physical perturbations and deceiving classifiers in diverse contexts.

5.2. SYSTEM OVERVIEW

5.2.1. THREAT MODEL

As shown in Figure 5.2, we consider the camfecting scenario [44], where a hacker exploits some loopholes [187], [188] to take full control of the USCs. This allows the hacker to capture a large collection of images. Given that most of the captured images may not contain sensitive information, it could be extremely inefficient for the hacker to manually inspect each captured image. For instance, the chance of capturing a password note is very low, occurring only when the USC is oriented toward the location of the password note. Recent reports [48], [49] indicate that attackers have employed deep learning classification models to automatically sift through large volumes of data gathered online for sensitive information. Such models have also been integrated into malware, notably

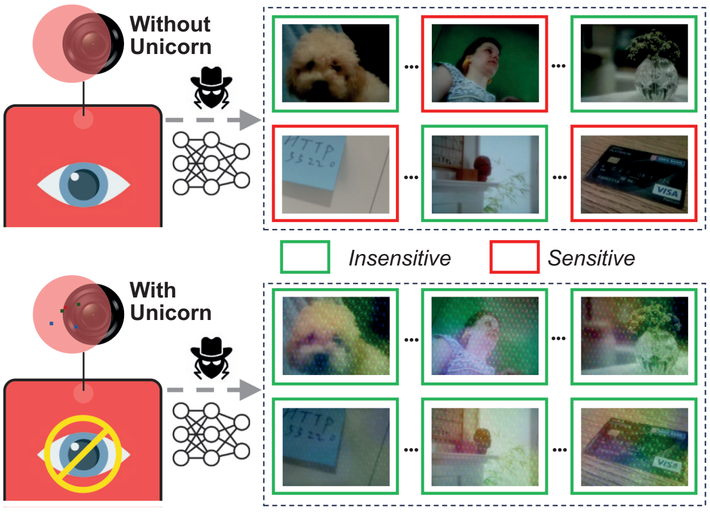


Figure 5.2: Illustration of the threat model & defense scenario. When an USC without Unicorn is hacked to gather a large collection of images, the hacker can leverage DNNs to identify sensitive user information. However, if an USC with Unicorn is hacked, the Unicorn's protection prevents the recognition.

Emotet [189], to filter through victims' data efficiently. Thus, we consider that the hacker utilizes a deep learning classification model to automatically identify sensitive information from those captured images.

Hacker's goal. The hacker leverages a classification model (referred to as **target model**) to infer sensitive information from a large collection of captured images. For instance, the hacker could utilize the target model to recognize identities of users from captured images. The hacker could also identify whether the captured images contain sensitive information such as password notes and credit cards. We note that obtaining those sensitive information could be the first step in conducting advanced and targeted cyber-attacks. According to [190], the majority of illicit camera access is related to gathering information for financial gain. With access to credit card details or passwords, a hacker can execute financial fraud or gain unauthorized access to secured systems [190].

Hacker's capability and knowledge. We consider a hacker with strong capabilities and knowledge. The hacker has full control over the USC parameters, e.g., exposure time, ISO, and can apply any post-processing to the captured images. The hacker knows all sensitive information classes of interest. The hacker has a dataset that consists of people's facial images and other sensitive objects sourced via web scraping. They also have sufficient computational resources to train powerful target models (e.g., IncepResNet [146], MoileNet [147], and ResNet [142]) by themselves or utilize other advanced, publicly available image classification services (e.g., Google Vision API [50] and ChatGPT Vision API [51]).

5.2.2. PROBLEM FORMULATION

Design goals. In this chapter, we aim to design a protection method that could be deployed by a user to its device with USC. In particular, the protection method adds carefully designed "*screen pattern*" on the transparent screen region to embed a universal screen perturbation to the captured images. For convenience of expression, the term "screen" will be used from now on to specifically refer to the transparent screen region. Specifically, we have the following goals when designing the protection method:

- The perturbation should be unnoticeable to the user when using the device, i.e., it will not disrupt the user's normal use of the device.
- When the perturbation is embedded into captured images, target models have low recognition accuracy on the captured images that contain sensitive information.
- It will be energy-efficient, enabling it to operate constantly in the background of mobile devices with minimal energy consumption.

Assumptions for the user. We assume the user has access to some computing resources to calculate the screen pattern to apply for universal screen perturbation. Moreover, the user has some sensitive information classes of interest to be protected. We also assume that the user has no knowledge about the target model utilized by the hacker for image classification. However, the user can train a **surrogate model** for the task, which may differ in architecture and hyper-parameters from the target models.

5.3. UNIVERSAL SCREEN PERTURBATION

5.3.1. DESIGN SPACE

The pixels on the screen are arranged in an RGB array, which is composed of multiple screen-pixel units, each with three R/G/B subpixel sets. A single R/G/B subpixel set can include multiple identical-color subpixels due to the control circuitry designs (cf. Figure 4.2 in Chapter 4). We use θ to denote the parameters that affect the screen perturbation generation and divide into the following two categories:

- **Manufacturing-fixed parameters** (θ_{fixed}): correspond to hardware-level parameters set by the screen manufacturer including density, shape, size, and the number of R/G/B subpixels. They vary across different full-screen models.
- **Dynamic optimization parameters** (θ_{free}): correspond to color, position, and brightness of R/G/B subpixel sets that we aim to optimize during the perturbation generation. Unicorn modulates these screen-pixels via software-level controls in full-screen devices.

5.3.2. SCREEN PERTURBATION GENERATION

We aim to generate a universal screen perturbation U that by configuring a set of screen parameters $\{\theta_{\text{fixed}}, \theta_{\text{free}}\}$, i.e., $U(\theta_{\text{fixed}}, \theta_{\text{free}})$. When embedding U with any USC-captured image I drawn from the sensitive scene distribution $\mathscr{I}_s$, the resulting perturbed image can fool the target model Φ in recognizing the sensitive information containing in I. This objective can be formulated as:

$$\Phi(I) \neq \Phi(I \oplus U), \forall I \in \mathscr{I}_s, \tag{5.1}$$

where $\oplus$ is the image formation process of the USC that will be described later in §5.4.4. To achieve our objective, we start by optimizing the following loss function:

$$\min_U J(f(I \oplus U), l^*), \tag{5.2}$$

where f is a surrogate model that approximates the behavior of Φ used by the hacker. We rely on the transferability effect between Φ and f, as models trained for similar tasks often share properties and vulnerabilities, even when trained on different architectures and datasets [127], [129]–[131]. $J(\cdot,\cdot)$ is the loss function measuring the difference between the prediction of f and the target insensitive label l^*. In our current design, l^* is a set of insensitive labels selected by the user, which are different from the sensitive ones.

Post-processing loss. Images captured by USC also contain passive screen perturbation that will affect the recognition accuracy of the target model Φ. To improve recognition accuracy, hackers can use advanced deblurring algorithms to obtain sharper and more detailed images. We incorporate the deblurring algorithm into our loss function as:

$$J(f(\mathscr{P}_d(I \oplus U)), l^*), \tag{5.3}$$

where $\mathscr{P}_d$ is a surrogate deblurring algorithm (detailed in §5.4.3).

Visual loss. Changes made by Unicorn on the transparent screen should not affect the user's visual experience when using the mobile device. Thus, we impose a visual constraint on U such that the changes in multiple screen-pixel units are not perceptible to the human visual system. Specifically, the visual constraint ensures that the modifications made by Unicorn, in screen-pixel position, color, and intensity, will not deviate significantly from the original screen display. The visual loss is defined as:

$$\mathscr{L}_{\text{vis}}: \ |\text{DSSIM}(\text{Resize}(L(S_{\text{ori}}), \kappa), \text{Resize}(L(U^{-1})), \kappa)) - \rho|, \tag{5.4}$$

where S_{ori} is the original unaltered screen, while U^{-1} is the screen pattern for generating universal screen perturbation. DSSIM$(\cdot)$ calculates the structural dissimilarity index that measures the user-perceived image distortion [121], [139]. Resize$(\cdot)$ and $L(\cdot)$ are image resizing and low-pass filtering functions. We use them to simulate changes in screen-pixel density and viewing distance during smartphone use and assess the visual impact of screen-pixel changes. Lastly, ρ denotes the perceptual perturbation budget. We leverage the downsampling scale κ to simulate human visual perception, influenced by screen resolution, the viewing angle, and distance [140], [141].

Energy loss. Lastly, to ensure energy efficiency, we incorporate an energy loss into the optimization. First, the power consumption of the OLED screen pixel linearly correlates with its RGB values, specifically the gamma decoding from its sRGB values [191]. Meanwhile, the number of screen pixels that have been light up also affects the overall power consumption. Therefore, we define the energy loss as:

$$\mathscr{L}_{\text{en}}: \sum_{i}^{N(U)} P_{\text{pixel}}(c_i(U)), \tag{5.5}$$

where $c_i = [R, G, B]$ represents the color combinations of the activated screen-pixel i, while N denotes the number of activated color screen-pixel units. Since the activated screen pixels are used to generate universal screen perturbation, they are all determined by U. The power model for each screen-pixel unit can be derived from [191] as:

$$P_{\text{pixel}}(\hat{R}, \hat{G}, \hat{B}) = h_r(\hat{R}) + h_g(\hat{G}) + h_b(\hat{B}), \tag{5.6}$$

where $h_r(\cdot)$, $h_g(\cdot)$, and $h_b(\cdot)$ represent power consumption of red, green and blue subpixel sets, respectively. Note that $\hat{R}$, $\hat{G}$, and $\hat{B}$ are the values of R, G, and B after gamma decoding in the color space, i.e., $R^\gamma = \hat{R}$, $G^\gamma = \hat{G}$, $B^\gamma = \hat{B}$, where γ denotes gamma correction coefficient. This is because the human visual system perceives brightness logarithmically, prompting screens to use gamma correction to enhance display quality.

Total loss function. After integrating all loss terms, our final optimization problem is formulated as:

$$\min_U \lambda_{\text{bl}} J(f(I \oplus U), l^*) + \lambda_{\text{debl}} J(f(\mathscr{P}_d(I \oplus U)), l^*) + \lambda_{\text{vis}} \mathscr{L}_{\text{vis}} + \lambda_{\text{en}} \mathscr{L}_{\text{en}}, \tag{5.7}$$

5

where λ_{bl}, λ_{debl}, λ_{vis} and λ_{en} are Lagrangian multipliers used to balance these loss terms.

To optimize the above loss function, a naive solution is to employ the gradient descent method. However, digital perturbations obtained by gradient descent are hard to practically reproduce using USC devices. This difficulty arises because the real universal screen perturbation emerges from a complex interplay between the optical synthesis of sensitive scene light, screen light, and USC's response. When a screen pixel is activated, the emitted color captured by an USC depends on multiple factors including (i) different pixel layouts on various device screens; (ii) varying colors and brightness of screen pixels at different positions, which affect the active perturbations in the captured images; (iii) the Bayer RGB filter's [17] unique color response in the USC; (iv) different USC parameter settings, such as the exposure time and ISO. Consequently, it is challenging to ensure that, by displaying a specific screen pattern, the transparent screen can accurately generate the effective perturbation, i.e., $U(\theta_{\text{fixed}}, \theta_{\text{free}})$, to fool the target model used by the hacker. Thus, rather than directly using gradient descent to optimize Equation (5.7), we must first understand the gradient relationship between the changes of screen-pixel parameters in the design space and the changes of the target model in classification. This requires U being a differentiable function for the parameter set $\{\theta_{\text{fixed}}, \theta_{\text{free}}\}$. In Section 5.4 we design a deep learning-based under-screen camera model to capture the gradient relationship of parameter changes in $I \oplus U(\theta_{\text{fixed}}, \theta_{\text{free}})$.

5.4. NEURAL UNDER-SCREEN CAMERA ARCHITECTURE

The overview of the neural USC architecture is shown in Figure 5.3, which consists of the screen model (Section 5.4.1; § 5.4.1), the active screen perturbation generator (Section 5.4.2; § 5.4.2), the passive screen perturbation generator (Section 5.4.3; § 5.4.3), the exposure module (Section 5.4.4; § 5.4.4), and the USC noise generator (Section 5.4.5; § 5.4.5). Lastly, we also propose a data augmentation technique (Section 5.4.6) in the optimization process to enhance the robustness of the generated screen perturbations. Below, we introduce each of the system components in detail.

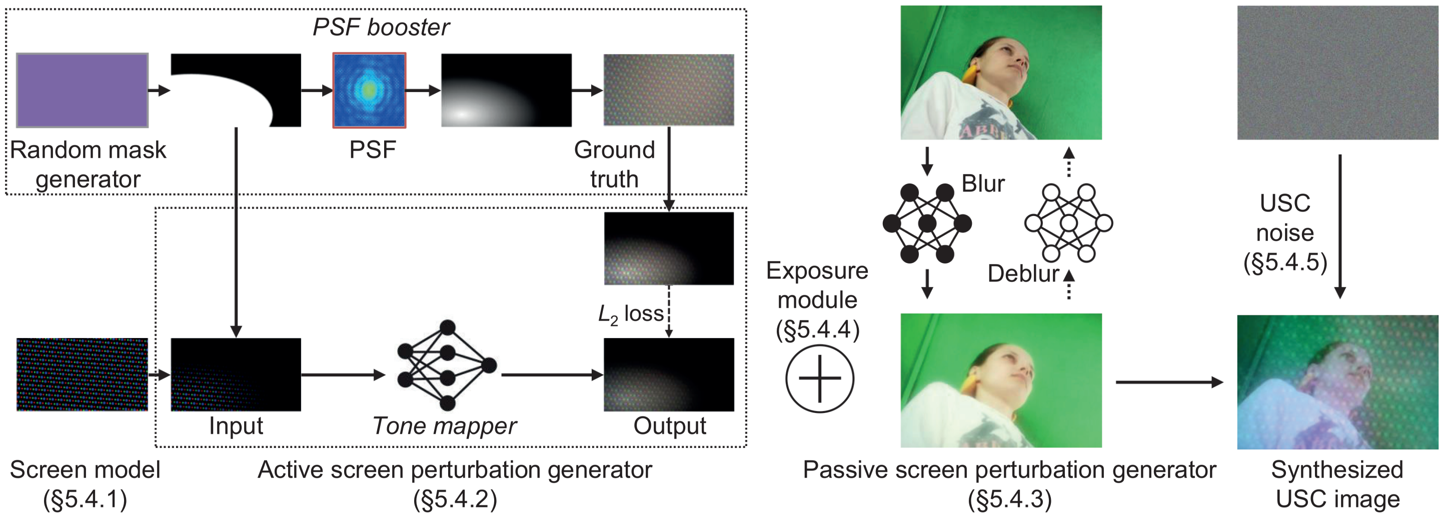


Figure 5.3: Overall developed neural USC architecture: screen pixels are activated in the screen model (Section 5.4.1; §5.4.1) and fed into the active screen perturbation generator (Section 5.4.2; §5.4.2) to generate active perturbations. The passive screen perturbation generator (Section 5.4.3; §5.4.3) overlays passive perturbations on the original image through the blur module. Both active and passive perturbations undergo optical synthesis controlled by the exposure module (Section 5.4.4; §5.4.4). USC noise (Section 5.4.5; §5.4.5) is added to synthesize the final USC image.

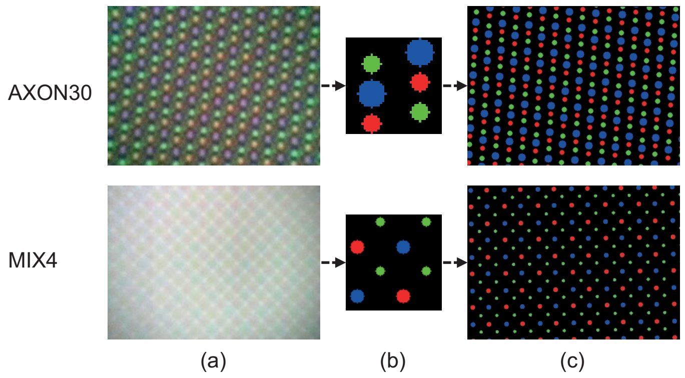


Figure 5.4: Reverse engineering the screen models of two COTS full-screen smartphones, i.e., ZTE AXON30 and Xiaomi MIX4. (a) different screen-pixel layout images captured by the USC are taken as inputs; (b) screen-pixel units obtained by the proposed minimum unit search method; (c) the final screen model generated by composing the obtained screen-pixel units.

5.4.1. SCREEN MODEL

Camera optics [17] indicate that even tiny illuminated screen pixels can result in an array of translucent, speckled color variations and shifts on the captured image. The generation of U is determined by the configuration of the screen, i.e., $\theta_{\text{fixed}}, \theta_{\text{free}}$, making U a function of the screen model $S(\theta_{\text{fixed}}, \theta_{\text{free}})$. Thus, our first step is to obtain the screen model of the smartphone. However, as shown in Figure 5.4, screen models vary across devices (θ_{fixed}) and exhibit diverse screen-pixel shapes and layouts. These factors affect the images captured by the USC, as shown in Figure 5.4(a). Such variations make accurately representing screen perturbation U challenging. Below, we introduce how to learn screen models from different manufacturers in detail.

A. SCREEN-PIXEL TILING

Motivated by [14], we propose a *minimum unit search* method. This method is used to find the screen-pixel unit, as screen-pixels on the screen are tiled into numerous identical units (see Figure 5.4(a)). This tiling arrangement is characterized by a periodic pattern where each screen-pixel unit is periodically positioned at a constant distance $D\ \mu m$. Given that the screen resolution, denoted as N screen-pixels per inch, can be obtained from the smartphone's specifications, we can calculate the tiling period D from $D = 1\ \text{inch}/N$. Therefore, knowing the layout of R, G, and B subpixels in a screen-pixel unit allows us to infer the complete screen-pixel layouts on the screen. Specifically, the spatial distribution of screen-pixels across the entire screen, denoted as S, can be mathematically represented by the following equation:

$$S = m * \sum_{n_x} \sum_{n_y} \delta(x - n_x D)\delta(y - n_y D), \tag{5.8}$$

where m denotes the layout of a screen-pixel unit. $\delta(\cdot)$ represents the Dirac delta function, which serves as the unit impulse function [192]. n_x and n_y are indices for the horizontal and vertical positions of screen-pixels, respectively, in the spatial domain.

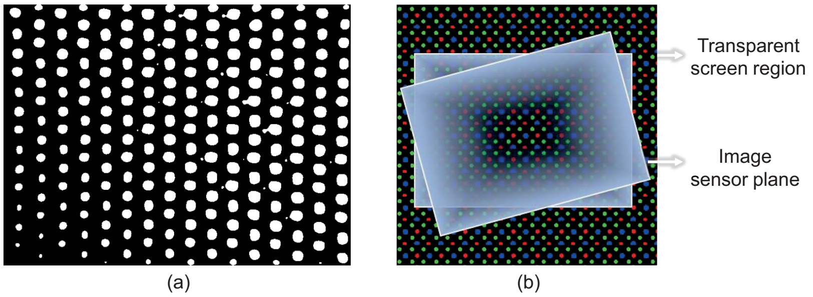


Figure 5.5: (a) *Relative illumination*: the relative illumination phenomenon causes light intensity in the corners of images captured by the USC to be lower than in the center of the image. Therefore, the OTSU filter cannot accurately identify screen-pixels in the corners of the images. (b) *Relative position*: slight variations in the relative positions between the image sensor in the USC and the transparent screen can lead to significant changes in the screen-pixel layout of the final obtained screen model.

Color space theory states that a screen emitting light from all its screen-pixels is perceived as white due to the combined emission [71]. In our study, we display white light on the screen and capture ground truth images of screen-pixel positions with an USC. The captured images show both the direct emissions from screen-pixels and the diffused light emissions from adjacent ones. To mitigate the effect of this diffused light and accurately localize each screen-pixel, we employ the OTSU thresholding method [89]. This technique segments the image into foreground and background based on the light intensity, utilizing the fact that light intensity at screen-pixel positions is inherently higher than that of the diffused light. By applying the OTSU method, we effectively isolate and eliminate the diffused light, preserving only the essential screen-pixel position information in the foreground. However, this intensity-based segmentation faces challenges from the phenomenon of relative illumination.

B. RELATIVE ILLUMINATION

Relative illumination refers to the brightness variability in USC-captured images, with central areas typically appearing brighter than corners. This relative illumination effect is evident in screen photographs captured by USCs, as illustrated in Figure 5.5(a). In USC-captured image, the screen-pixels in the lower left corner exhibit lower brightness after grayscale thresholding. To overcome this challenge, we implement the mask-correction technique [118] proposed in Chapter 3. This technique involves generating a mask M that keeps only the active screen perturbations using a combination of filtering and contour detection methods, as depicted in Figure 5.6. This mask is crucial in uniforming the size of the speckle patterns produced by lit screen-pixels and in mitigating unneeded noise in the image. This preprocessing step facilitates a more accurate learning of the screen-pixel unit layout. Consequently, the final ground truth image can be mathematically represented as follows:

$$\hat{G} = G * M. \tag{5.9}$$

Due to the camera lens's optical properties, illuminating even a tiny screen-pixel can

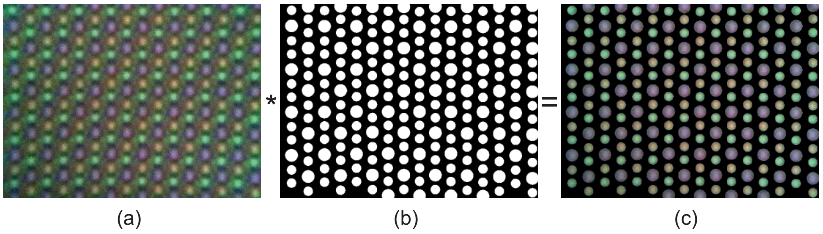


Figure 5.6: (a) Captured lit-up screen-pixels; (b) A foreground mask focused solely on the screen-pixel area using a filtering method; (c) The final ground truth image $\hat{G}$.

create slightly larger colored speckles in the captured image. The projection relationship between the lit screen-pixel size and speckle size is fundamentally influenced by the USC's focal length [96].

C. FOCAL LENGTH

Focal length plays a vital role in the imaging process: Increasing it narrows the USC's field of view and magnifies the scene, reducing the number of screen-pixels in the captured image. Conversely, decreasing focal length broadens the view, capturing a wider, more distant scene, and thus including a greater number of screen-pixels in the captured image. Consequently, the screen-pixels of the entire screen are projected onto the ground truth image captured by the USC. This projection's transformation can be approximated as follows:

$$G_s = \text{Resize}(S, r_0) \tag{5.10}$$

where G_s is the synthesized image projected by the screen-pixels, S is the lit-up screen-pixels on screen, and r_0 represents the scaling factor associated with the focal length. To accurately reconstruct the screen model via periodic tiling of screen-pixel units, the USC correct focal length can be determined through a one-dimensional search algorithm. This ensures screen model construction through periodic tiling of screen-pixel units.

Besides understanding the focal length's influence, accurately modeling the screen also requires considering the relative position between the screen and the USC.

D. RELATIVE POSITION

The spatial relationship between the screen and the imaging sensor of an USC varies across different smartphone models, as illustrated in Figure 5.5(b). Therefore, calibrating the screen model to align precisely with the actual positioning of the USC is crucial. To address the misalignment of the screen-pixel patterns, we apply two-dimensional affine transformations [193], which enable accurate control of the position and orientation of each screen-pixel group. These transformations are expressed mathematically as:

$$\hat{G}_s = R * G_s, \tag{5.11}$$

$$R = \begin{pmatrix} \cos\varphi & -\sin\varphi & -x_0\cos\varphi + y_0\sin\varphi + x_0 \\ \sin\varphi & \cos\varphi & -x_0\sin\varphi - y_0\cos\varphi + y_0 \end{pmatrix}, \tag{5.12}$$

where R denotes the affine transformation matrix, incorporating rotation through angle φ and centroid (x_0, y_0). Since the USC's position is defined by discrete coordinates, using gradient-based optimization with continuous parameters is challenging. Affine transformations offer a continuous, differentiable approximation of the USC's relative position, facilitating precise, tunable screen model construction.

E. SCREEN MODEL GENERATION

To better understand screen-pixel behavior, we set all screen-pixels to emit light, displaying a pure white screen. We captured 100 successive frames utilizing the USC to form the training dataset, ensuring the USC remained stationary throughout the data collection process. This method helps mitigate the impact of USC noise in the optimization process. For the optimization phase, we treat the screen-pixel unit as a variable. Employing the gradient descent technique, we optimized a loss function aimed at deriving the screen-pixel unit configuration for the specific smartphone model. Our optimization goal is to minimize the differences between the captured ground-truth images and the synthesized images composed of the screen-pixel units. We also find that image areas with consistent colors are easier to recreate. To ensure that these easier areas help improve the overall accuracy of the loss function, we implement a technique known as hard-sample mining [194]. This strategy involved calculating the residuals, $\Delta G = \hat{G} - \hat{G}_s$, converting them into vector form, and sorting the absolute values of these residuals, $|\Delta\vec{G}_t|$, in descending order. We then select the top 10 percent of these residuals for L_2 loss computation. The loss function is expressed as:

$$\mathscr{L}_{screen} = \sum_{t=1}^{T} |\Delta\vec{G}_t|^2, \tag{5.13}$$

where T represents the number of elements within the top 10 percent tier of total loss.

To this end, we employ reverse engineering to learn the screen-pixel layout of different smartphone models. An example is shown in Figure 5.4, in which we obtain the screen models of two COTS full-screen smartphones, i.e., ZTE AXON30 and Xiaomi MIX4. First, as shown in Figure 5.4(a), we repeatedly display a white screen on the smartphone screen and capture a series of images with the USC. They will be used as the inputs for reverse engineering. Then, we apply the reverse engineering method to determine the screen-pixel unit layouts shown in Figure 5.4(b). Finally, we approximate the locations of the output screen pixels and compose the screen model S_{AXON30} and S_{MIX4} for both the AXON30 and MIX4, as shown in Figure 5.4(c). By obtaining the screen model, we can manipulate screen pixels with varying brightness and colors at different positions on the screen model to create distinct active screen perturbations.

5.4.2. ACTIVE SCREEN PERTURBATION GENERATOR

Constructing the screen model S alone is not enough to replicate the screen perturbations appearing in practical images captured by USCs when the transparent screen is active. This is because each smartphone USC has a different Bayer filter [17] that leads to a different color response [195]. Moreover, the variation in screen's finite openings and USC's lens will influence how screen light diffracts and scatters after passing through this combined USC aperture. To fill the gap between the screen pixels in the screen model to the final captured image, we develop a deep learning-based computational model $\mathscr{P}_a$ to find the universal screen perturbation U, which is defined as:

$$U = \mathscr{P}_a(S(\theta_{\text{fixed}}, \theta_{\text{free}})), \tag{5.14}$$

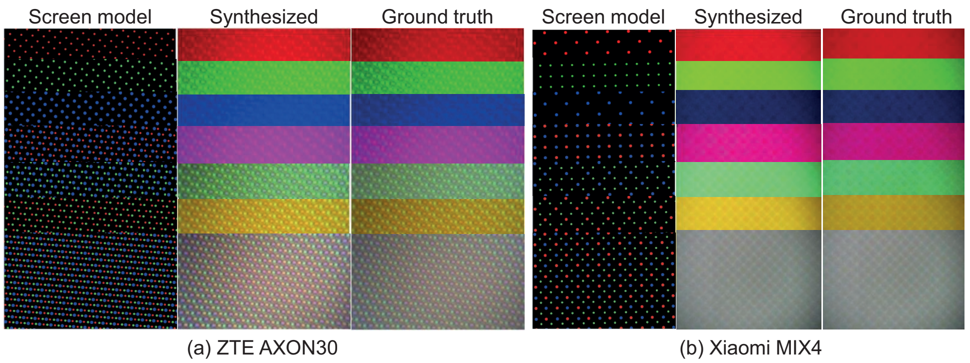


Figure 5.7: Comparison of output results of the tone mapper on USCs of ZTE AXON30 and Xiaomi MIX4. The figure shows synthesized active screen perturbation and the captured ground truth when different R/G/B subpixel sets on the screen model are activated.

where $\mathscr{P}_a$ is a neural network with a UNet architecture [196], [197], consisting of the tone mapper and the PSF booster. Below, we introduce these two components in detail.

A. TONE MAPPER

Different USCs exhibit distinct color responses due to their unique Bayer filters. Thus, the tone mapper aims to learn the tone-mapping function [195] for each USC. In our design, the UNet structure maps the illuminated screen pixels of the screen model to the active screen perturbations in USC-captured images. The input for the tone mapper is the lit screen pixels generated by screen model S, as shown in Figure 5.7, and its output is the active screen perturbation image. The pyramid architecture of the UNet enables multi-scale analysis of the input screen-pixel model's characteristics. In the meantime, its encoding-decoding structure, enhanced by skip connections, effectively integrates features across different scales. This design successfully simulates the color response of the USC to various colored screen pixels and the image degradation caused by the limited aperture, such as blurring. For data collection, we set the smartphone's screen to sequentially display seven different colors: red, green, blue, cyan, magenta, yellow, and white. This procedure sequentially activates the respective subpixel sets, with the primary colors (red, green, blue) lighting up the subpixel set individually, the secondary colors (cyan, magenta, yellow) illuminating combinations of two subpixel sets, and white engaging all subpixel sets simultaneously. This approach reflects the screen model's operational dynamics. For each color, the USC captured 300 sequential frames. This process allows us to mitigate the effects of USC noise through averaging. By selecting these distinct hues, we approximate the screen's continuous spectral output using the screen model's discrete spectrum. This enables accurate mapping of the screen model's output onto the USC's response spectrum, establishing a comprehensive model for the USC.

B. PSF BOOSTER

The active screen perturbation generator, relying solely on the tone mapper, falls short in robustly mapping the screen model to active screen perturbations. As Figure

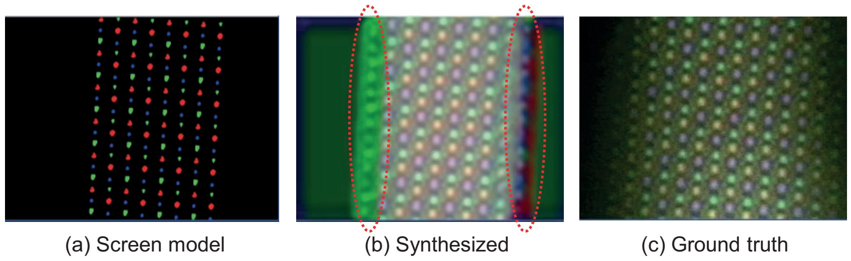


Figure 5.8: Only a subset of the screen-pixels in the screen model is lit in (a). The active screen perturbation synthesized by the tone mapper in (b) exhibits color shifts and artifacts, highlighted by the red dotted oval. The USC captured ground truth in (c) naturally diffuses.

5.8 shows, when only a subset of screen-pixels on the screen model is illuminated and processed through the tone mapper, it leads to significant color aberration and artifact in captured active perturbations. This limitation highlights the insufficiency of the tone mapper in faithfully replicating the diffusion behavior of screen-pixel illumination.

Inspired by the Point Spread Function (PSF) concept [14], [180], we introduce a PSF booster. The PSF characterizes an optical system's response to a point light source, detailing how the system's aperture affects optical signal diffraction. Ideally, the PSF is infinitesimally small for maximum resolution at the point light source. However, even in an ideal aberration-free system, aperture diffraction in USCs prevents the convergence of point light sources into a single point. Instead, it results in a PSF "diffusion" pattern. The property of the PSF effectively simulates how light diffuses from a specific targeted area of screen pixels to create the active screen perturbations captured in the image. Specifically, we incorporated two additional modules: a random mask generator and a PSF generation network, augmenting the original UNet-structured tone mapper. The random mask generator module creates a variable screen-pixel activation mask, determining which screen pixels to illuminate on the screen model. This mask then guides the PSF generation network module in producing a screen perturbation mask that depicts the diffusion of screen pixels as captured by the USC. The PSF generation network adopts the PSF model of the USC proposed in Chapter 4, which treats the screen as part of the USC aperture. The actual active screen perturbation image is constructed by superimposing this screen perturbation mask onto the ground truth image. The original lit screen-pixel image is processed through the tone mapper to generate a synthesized active screen perturbation image. By minimizing the L_2 loss between the actual and synthesized active screen perturbations, we effectively train the entire active screen perturbation generator $\mathscr{P}_a$. Details of $\mathscr{P}_a$ are in Figure 5.9.

We applied our method to two full-screen smartphones, the ZTE AXON30 and the Xiaomi MIX4, which have distinct active screen perturbations. To validate the similarity between synthetic and ground truth images, we utilized two image metrics: the Structural Similarity Index (SSIM) [154] and the Peak Signal-to-Noise Ratio (PSNR) [96]. The SSIM and PSNR values for synthesized images from the AXON30 are measured at 0.98 and 32.44 dB, respectively. For the MIX4, these metrics are 0.95 and 31.89 dB, respectively. Note that SSIM>0.9 and PSNR>30 dB indicate a high degree of similarity between

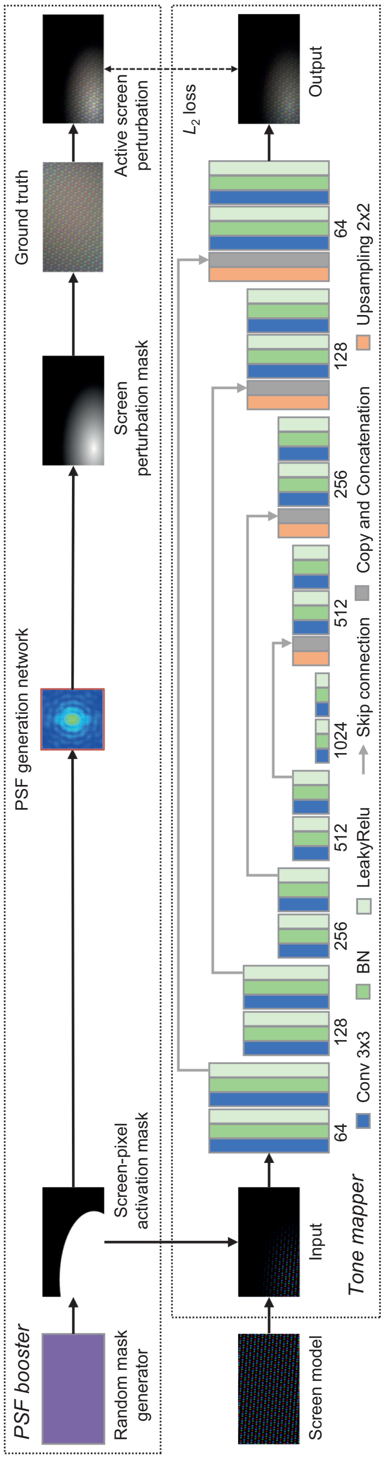


Figure 5.9: The detailed architecture of the active screen perturbation generator, consisting of *Tone mapper* and *PSF booster*.

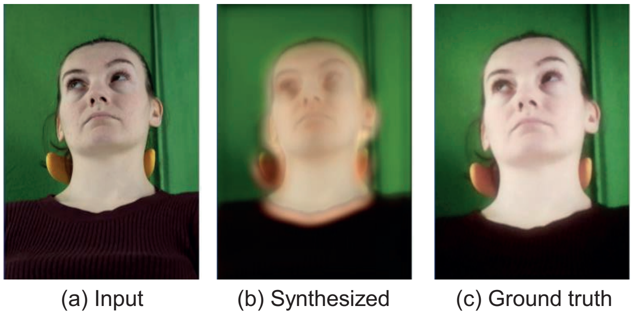


Figure 5.10: Abnormal blurring and artifact in the (b) synthesized image arise from training the blur module with an unaligned dataset lacking strict spatial constraints between (a) the input pristine image and (c) the USC captured ground truth.

two images. At this level, differences become imperceptible to the naked eye [156].

5.4.3. PASSIVE SCREEN PERTURBATION GENERATOR

The USC also introduces passive screen perturbation on the formed image even when the screen is inactive, and makes the captured image blurry. We define the pristine image as $\hat{I}$ and the actual captured blurry image as I, the network (blur module) simulating this passive screen perturbation as $\mathscr{P}_b$, and the network (deblur module) that can remove this passive screen perturbation as $\mathscr{P}_d$. We have:

$$I = \mathscr{P}_b(\hat{I}), \;\; \tilde{I} = \mathscr{P}_d(I), \tag{5.15}$$

where $\tilde{I}$ is the deblurred image. Below, we introduce how to obtain $\mathscr{P}_b$ and $\mathscr{P}_d$ in detail.

A. ALIGNED DATASET GENERATION

Effective modeling of passive perturbation requires training with pixel-wise aligned pairs of pristine (perturbation-free) and captured images. However, it is very difficult to collect large-scale and perfectly aligned paired training data. Directly training with misaligned pairs leads to undesirable outcomes. Our preliminary results indicate that using misaligned image pairs for pixel-wise supervision when training $\mathscr{P}b$ and $\mathscr{P}d$ tends to produce images with blur and notable artifacts, as shown in Figure 5.10, which is consistent with existing work [198]. This issue arises due to the lack of strict spatial constraints on regions prone to blur, with the majority of operations conducted in the feature space. Consequently, the synthesized images suffer from significant abnormal blurring and spatial displacement, as the network prioritizes image scaling and gap-filling to simulate alignment with the captured images, rather than accurately learning the characteristics of passive screen perturbation given by the screen. This highlights the challenge of obtaining perfectly aligned paired data for passive screen perturbation generation. To address this issue, we employ the PSNR > 20 dB metric as an alignment index and leverage transformer-based frameworks [199] to "clone" pristine images onto USC images, aiding in the collection of aligned datasets for USCs. This enables us to select a well-aligned subset for our training dataset.

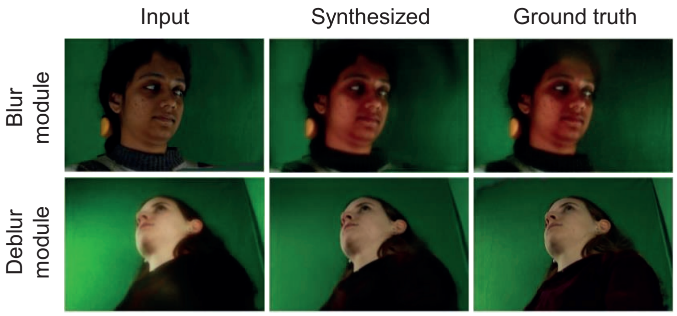


Figure 5.11: Results of the blur & deblur modules: the synthesized images are quite close to the ground truth.

B. BLUR/DEBLUR MODULE

We use the same UNet network as tone mapper but with L_1 loss to train both $\mathscr{P}_b$ and $\mathscr{P}_d$ using the collected aligned dataset, as L_1 loss is less sensitive to outliers and spatial displacement compared to L_2 loss. For the blur module, the input is a pristine image with the corresponding blurred image, captured by the USC, serving as the ground truth for loss calculation. Conversely, for the deblur module, the input is the blurred USC-captured image, and the loss is calculated against the pristine image.

As shown in Figure 5.11, the blur module accurately simulates images actually captured by the USC. The deblur module can potentially be used by hackers to deblur all images captured by USCs, thereby improving classifier recognition accuracy. To counteract such vulnerabilities, we integrate the deblur module within Equation (5.7), boosting our universal screen perturbation's resistance to hacker-employed deblurring techniques. The PSNR and SSIM calculated between the blurred image synthesized by the blur module and the ground truth image are 33.69 dB and 0.96, respectively. For the deblurred image synthesized by the deblur module and the ground truth image, these metrics are 34.59 dB and 0.95, respectively. These results show that both the blur and deblur modules perform well in simulating and removing passive screen perturbation.

5.4.4. EXPOSURE MODULE

The image formation process ($\oplus$) of the USC is determined by USC settings, such as exposure time, which decides the duration that the USC's shutter remains open. The exposure time is proportional to the number of photons that hit the image sensor. Similarly, ISO settings influence the image sensor's sensitivity to light. A higher sensitivity results in an increase in the recorded light intensity. These USC settings proportionally impact captured image luminance, affecting both blurry image luminance and active screen perturbation strength. Excessively bright screens can induce over-saturation when coupled with high-gain settings. To manage this, we adopt the gain and saturation framework as outlined in [200], [201] and use β and $clip$ to denote USC gain and saturation constraints, respectively. On the other hand, due to the transparent nature of the screen, the superposition of the captured blurry image I and the optical perturbation generated by the screen can be regarded as an alpha-blending process [202], [203].

Eventually, our imaging model is:

$$I \oplus U(\theta_{\text{fixed}}, \theta_{\text{free}}) = clip(\beta * (\alpha I + (1-\alpha)U(\theta_{\text{fixed}}, \theta_{\text{free}}))), \tag{5.16}$$

where $clip(x) = \min(\max(x,0),1)$, and α denotes the screen perturbation's opacity.

5.4.5. USC NOISE GENERATOR

We model the shot and read-out noise from the USC as Poissonian (σ_p) and Gaussian noise (σ_g) [204]. These noise patterns are superimposed onto images including both active and passive screen perturbations. The specific parameters for σ_p and σ_g are calibrated based on the noise characteristics of USCs from different smartphone models. To measure the noise statistics σ_p and σ_g, we collect 100 consecutive frames in a dark environment where there is no ambient light and the smartphone screen is turned off. We then compute their differences to estimate the noise distribution.

5.4.6. TRAINING DATA AUGMENTATION

Making universal screen perturbations that remain effective in the physical world requires consideration of varying environmental factors. Previous work has demonstrated that digital perturbations derived using direct printing methods, as discussed in [159], often fail under diverse viewing angles and distances [181]. To improve the physical robustness of universal screen perturbation, it is essential to integrate a variety of input transformations within the optimization process. We employ the expectation over transformation methodology [173], which enhances robustness by averaging the optimization loss across a set of synthesized training images. These synthesized images are manipulated through linear transformations to mimic differing environmental conditions, such as variations in lighting or perspective, thereby enhancing the generalizability of the universal screen perturbation. Thus, we have:

$$\begin{aligned}\min_{\theta_{\text{free}}} \mathbb{E}_{\hat{I}\sim\mathscr{I}_v} & \lambda_{\text{bl}} J(f(\mathscr{P}_b(\hat{I}) \oplus \mathscr{P}_a(S(\theta_{\text{fixed}}, \theta_{\text{free}}))), l^*) \\ & + \lambda_{\text{debl}} J(f(\mathscr{P}_d(\mathscr{P}_b(\hat{I}) \oplus \mathscr{P}_a(S(\theta_{\text{fixed}}, \theta_{\text{free}})))), l^*) \\ & + \lambda_{\text{vis}} \mathscr{L}_{\text{vis}} + \lambda_{\text{en}} \mathscr{L}_{\text{en}},\end{aligned} \tag{5.17}$$

where $\mathscr{I}_v$ represents a distribution over specific sensitive objects, such as face, password note, etc. Given images taken in the physical world, it is essential to ensure that the universal screen perturbation U, when added to captured image $\mathscr{P}_b(\hat{I})$, can fool the classifier under different physical conditions. By sampling instance $\hat{I}$ from $\mathscr{I}_v$, which includes pristine images of the sensitive object under variable distances, angles, environments, and lighting settings, we aim to more accurately reflect real-world dynamics. In this chapter, we have expanded the set of transformations to include additional environmental conditions. Figure 5.12 shows a complete optimization process.

USC Settings Adjustment. A hacker could control USC parameters like exposure time and ISO, either increasing USC gain (raising exposure time and ISO) to potentially enhance image details or decreasing it to reduce screen perturbations' effectiveness, aiding image recognition. To counteract such manipulations and enhance the resilience of universal

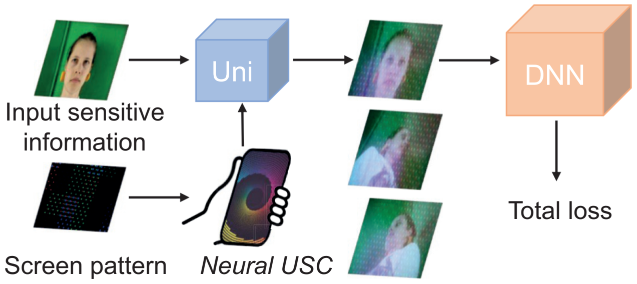


Figure 5.12: Overview of the universal screen perturbation generation pipeline. We optimize the screen pattern which passes through the neural USC architecture in order to minimize the total loss on a given surrogate model for a set of randomly generated permutations of the input. (Unicorn: Uni)

screen perturbations against varying USC gains, we randomly change the gain parameter, α from $[0.3, 0.7]$ and β from $[0.1, 10]$ throughout the gradient descent iterations. The efficacy of this approach in maintaining the robustness against USC setting manipulations by hackers will be demonstrated in Section 5.5.3.

5.5. Performance Evaluation

In this section, we first describe the models, datasets, and experimental configurations used in our evaluation. We then extensively test the Unicorn against image classification models in physical environments and real-world settings. We also explore factors affecting defense efficiency.

5.5.1. Experimental Setup

Smartphones. We employ two Commercial Off-The-Shelf (COTS) full-screen smartphones in the experiments, i.e., ZTE AXON30 and Xiaomi MIX4. Both devices feature a screen-pixel density of 400 Pixels Per Inch, with screen sizes of 2460×1080 (AXON30) and 2400×1080 (MIX4) pixels. We develop an Android application to turn on/off the screen pixels for generating the screen perturbations. Both USCs output images with a resolution of 1920×1080 pixels. The USC settings that hackers could potentially adjust, such as exposure time and ISO, are within the ranges of [1/5000 s, 10 s] and [100, 1600], respectively.

Tasks and target models. Our analysis focuses on two image classification applications used by hackers:

- *Identity recognition*. We use two representative backbone models: IncepResNet V1 [146] and MobileNet V2 [147], both pre-trained on the WebFace dataset [149].
- *Sensitive information mining*. We consider ResNet [142], which includes ResNet-18 and ResNet-50 with different network depths and parameters. All models are pre-trained on the ImageNet dataset [145].

Datasets. The identity recognition model is trained using 12,720 high-resolution facial images from the XGaze dataset [152]. These images are selected based on varied head poses and ambient lighting conditions. For the sensitive information mining task, we

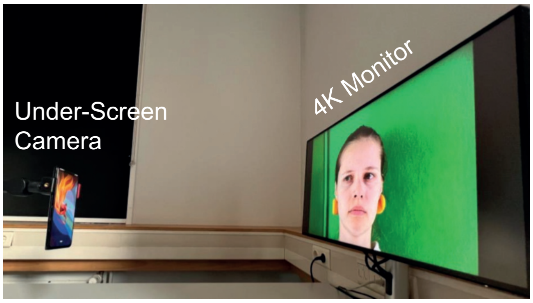


Figure 5.13: The testbed setup: we leverage a 4K monitor to display the original image and use COTS full-screen smartphones to capture images containing screen perturbations.

build our dataset using web crawlers. This dataset includes four classes of sensitive information (password notes, credit cards, files, and combination locks) and one insensitive class (vases, trash cans, etc.). Each class contains 1,000 images. The images containing sensitive information are all gathered through web crawlers. Conversely, the insensitive class comprises randomly chosen insensitive objects, such as vases and trash cans. This class's images are selected from miniImageNet dataset [150], which contains 60,000 images across 100 different classes.

Testbed. We build a monitor-USC imaging testbed to collect images captured by USCs. This monitor-based method, offering a controlled, efficient, and automated way to capture diverse scene contents, is also commonly employed in various USC image quality restoration tasks [10]. Images from our chosen dataset are displayed in full-screen mode on a 4K monitor. We adjust the aspect ratio as needed, either through rotation or resizing. To capture these images with USCs, we mount the smartphones on a stand, guaranteeing stable capturing of the images. For more setup details, please refer to Figure 5.13.

Metrics. For the identity recognition task, the goal is to prevent the hacker's target model from accurately identifying individuals. Thus, we use the ***non-targeted*** *misclassification rate of target models on USC-captured images* as the metric to assess the protection success rate. For the sensitive information mining task, the aim is to mislead the target model to classify images containing sensitive information as a unique insensitive class. Thus, we employ the ***targeted*** *insensitive classification rate of target models on USC-captured images* as the metric to represent the protection success rate.

5.5.2. OVERALL PERFORMANCE

Identity recognition task. We first evaluate Unicorn's performance in protecting identity using the XGaze dataset. We randomly select a user identity ("Subject-0119") from the XGaze dataset as the target identity and aim to prevent its identification from being recognized by applying our universal screen perturbation. We evaluate the performance of Unicorn under two scenarios: screen perturbation synthesized using the neural USC architecture, and perturbation captured by practical USC. We compare the performance with three benchmark cases: (i) *a pristine image without perturbation*, (ii) *passive perturbation with the screen OFF*, and (iii) *perturbation with randomly lit screen pixels*. Ta-

Table 5.1: Performance of Unicorn on identity recognition task (on subject-0119 identity).

Target model	Pristine image	Black screen Blurred/Deblurred	Random lit pixels Blurred/Deblurred	Screen perturbation (Synthesized) Blurred/Deblurred	Screen perturbation (USC) Blurred/Deblurred
IncepResNet	0%	12% / 2%	37% / 28%	**100% / 100%**	**100% / 100%**
MobileNet	0%	17% / 3%	44% / 39%	**100% / 100%**	**100% / 100%**

Table 5.2: Performance of Unicorn on sensitive information mining task (on password note class).

Target model	Pristine image	Black screen Blurred/Deblurred	Random lit pixels Blurred/Deblurred	Screen perturbation (Synthesized) Blurred/Deblurred	Screen perturbation (USC) Blurred/Deblurred
ResNet-50	1%	6% / 4%	8% / 4%	**96% / 92%**	**91% / 88%**
ResNet-18	3%	7% / 4%	8% / 4%	**99% / 95%**	**93% / 90%**

Table 5.3: The real-world protection success rate of Unicorn.

User 1				User 2			User 3			User 4			User 5		
AXON30 (default)	MIX4 (default)	AXON30	MIX4	Mi11 Pro	AXON30	MIX4	iPhone12	AXON30	MIX4	iPhone13 Pro	AXON30	MIX4	iPhone14 Pro	AXON30	MIX4
10%	10%	90%	95%	15%	95%	95%	5%	80%	85%	15%	80%	95%	0%	90%	100%

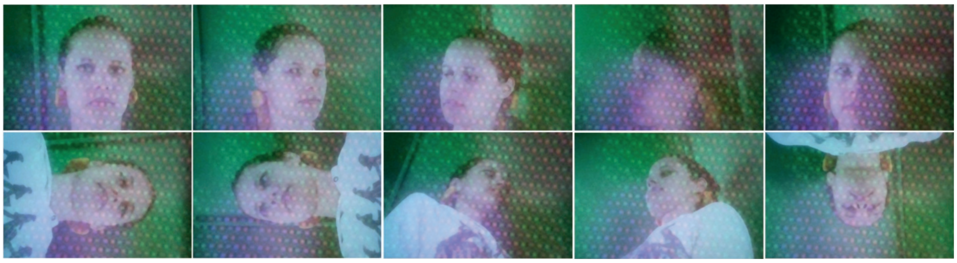


Figure 5.14: Protect on "Subject-0119" with different head pose and ambient light conditions.

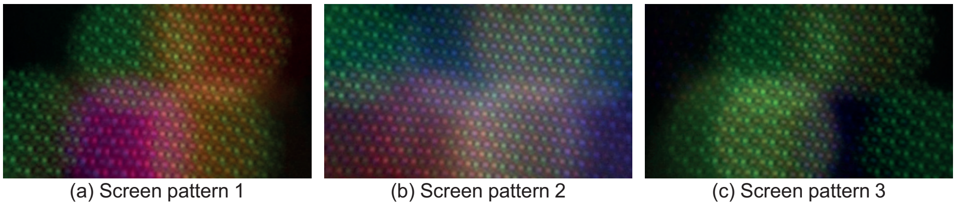


Figure 5.15: Universal screen perturbations generated by different optimized screen patterns obtained by changing the optimization starting point. They can all play a very good role in protecting user identity.

ble 5.1 shows the results. We can observe that even with only a passive perturbation like a black status bar, the protection success rate across all models increases by 15%. This rate further increases to 40% with random screen-pixel illumination due to the active yet unoptimized perturbation. In comparison, when applying both the synthesized and captured universal screen perturbations, the protection success rate on all target models reaches 100%. These results demonstrate the effectiveness of our Unicorn in disrupting identity recognition. An illustrating example is further shown in Figure 5.14, where the user's identity is protected under different head postures and ambient light conditions. We also investigate the performance of other optimized solutions in our screen perturbations. By varying the initial activation screen-pixel values and employing the gradient descent method, we generate diverse screen patterns. Figure 5.15 displays different universal screen perturbations generated by three screen patterns, each achieving a 100% protection success rate (cf. Table 5.4). Section 5.5.8 will discuss their resilience against potential adversarial training techniques employed by hackers.

Sensitive information mining task. Next, we assess the performance of Unicorn in protecting sensitive information, particularly focusing on password notes. The results are shown in Table 5.2. We can observe that with only passive perturbation, the protection success rate is only 5%. This rate further increases to approximately 7% with added random active screen perturbations. In comparison, when applying the synthesized universal screen perturbation on images in digital space, the protection success rate is boosted to higher than 95%. When employing universal screen perturbation on practical captured images, the protection success rate slightly drops to around 90%. This suggests that our established neural USC architecture effectively simulates the actual image formation process of the USC. Figure 5.16(a) illustrates the protection success rates across

Table 5.4: Protection success rate of universal screen perturbations generated by different screen patterns.

Model	Screen pattern 1	Screen pattern 2	Screen pattern 3
IncepResNet	100%	100%	100%
MobileNet	100%	100%	100%

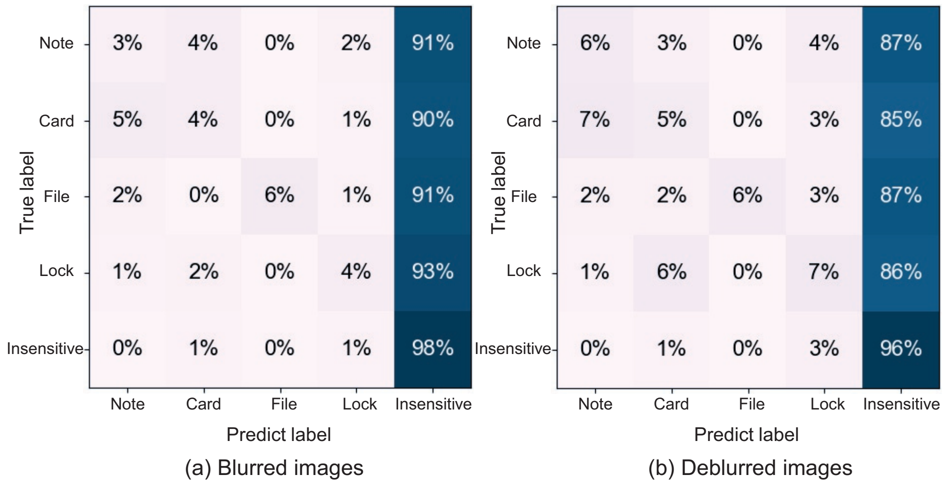


Figure 5.16: Protection performance on sensitive information mining task. All sensitive information classes are targeted classified as insensitive information class.

different classes of sensitive information, with Figure 5.17 providing corresponding examples. Our results demonstrate that images within sensitive classes are predominantly misclassified as insensitive class, achieving a targeted misclassification accuracy of over 90%, validating the effectiveness of our Unicorn in protecting sensitive information.

5.5.3. Robustness Results

Next, we evaluate the robustness of Unicorn. This assessment is particularly important given that hackers might use various technical methods and adjust physical device parameters to bypass the added universal screen perturbation.

Performance against image post-processing. We evaluate the performance of ScreenShield against state-of-the-art deblurring methods [10], [15], which hackers might use to reduce blur from passive screen perturbations and improve classifier accuracy. The results are presented in Tables 5.1 and 5.2. The effectiveness of the deblurring network is shown by the improved accuracy of all target models after deblurring, especially when the image contains only passive perturbations. However, applying universal screen perturbations results in only minor drops in protection success rates. This result is further illustrated in Figure 5.16(b), which shows that the deblurring network only marginally reduces the protection success rate of various classes by approximately 5%. This modest impact results from integrating our developed deblur module (cf. Section 5.4.3) into our loss function, strengthening our universal screen perturbation against hacker's exploiting of deblurring methods. This strategy ensures robustness against such advanced

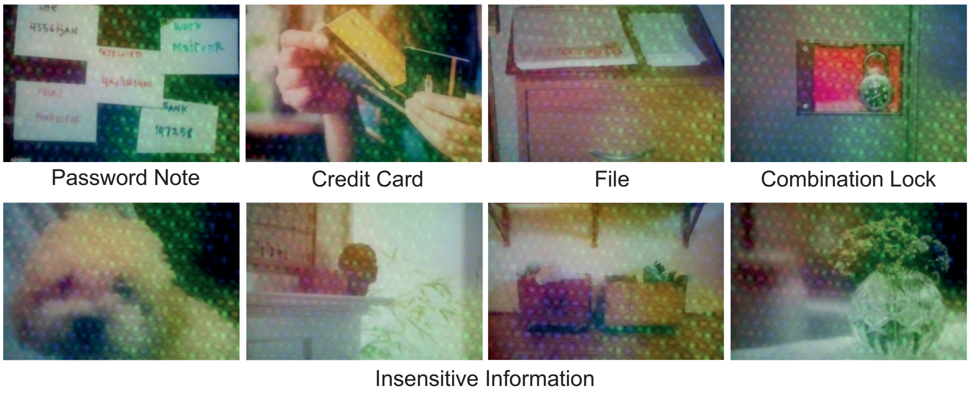


Figure 5.17: Examples of images contain sensitive/insensitive information with universal screen perturbation.

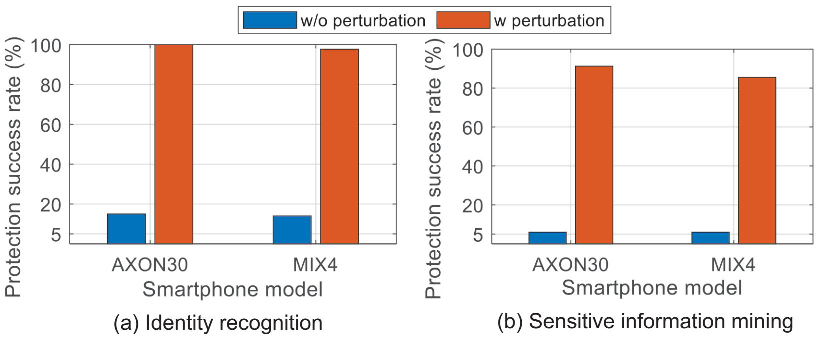


Figure 5.18: Protection performance vs. screen differences.

image restoration methods.

Screen diversity. We also evaluate Unicorn across different full-screen smartphones. The results are shown in Figure 5.18. Initially, when relying solely on passive screen perturbation, we note consistent performance across USCs from different manufacturers, below 20% in identity recognition task and 5% in sensitive information mining task. However, once implementing universal screen perturbation, there is a significant increase in the protection success rate, approximately reaching 100% in identity recognition and about 80% in sensitive information mining tasks. Specifically, ZTE AXON30 exhibits a higher protection success rate compared to that of Xiaomi MIX 4. This difference can be attributed to the distinct pixel layouts on the screens of these smartphones, which affects the effectiveness of the generated universal screen perturbations.

Different color in status bar. We assess the effectiveness of universal screen perturbation across four different status bar colors: *red, green, blue,* and *white* (see Figure 5.19 for the detailed setup). It is important to note that when the status bar is displayed in red, green, or blue, only the respective primary color subpixel sets are activated. Conversely, displaying a white status bar results in the illumination of all three primary color subpixel

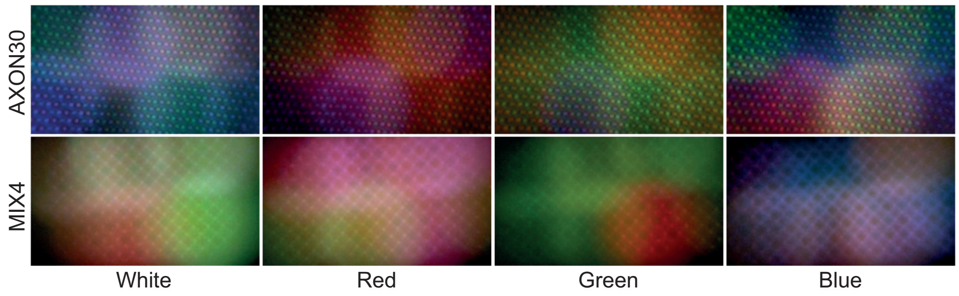


Figure 5.19: Universal screen perturbation with different color status bars at two commercial smartphones.

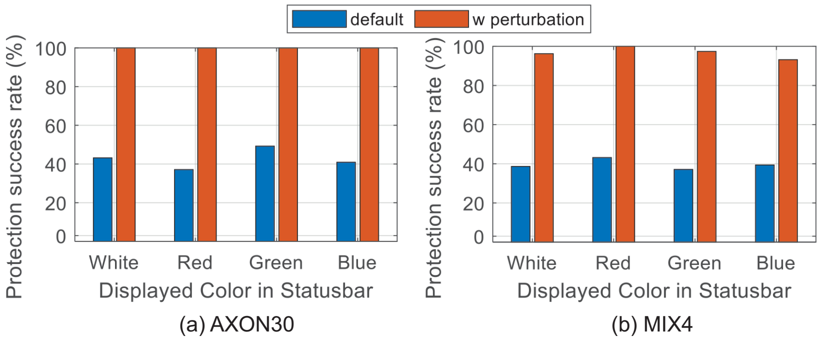


Figure 5.20: Protection performance vs. status bar color.

sets. The results are presented in Figure 5.20. Without universal screen perturbation, the protection success rate for each of the four original background colors is approximately 40%. However, once implementing the universal screen perturbation, there is a marked improvement in protection success rate. Specifically, the protection success rate reaches 100% on AXON30 and exceeds 90% on MIX4 for all four colors.

Under-screen camera settings. Considering that hackers might have full access to the USC, a potential physical countermeasure against the Unicorn involves adjusting the USC settings. These adjustments could either mitigate the impact of the universal screen perturbation or capture more detailed images. Actual shooting examples demonstrating the change in shutter time and ISO are reported in Figures 5.21 and 5.22, respectively. Figure 5.23 illustrates the impact of exposure time on images captured by the USC. Under automatic exposure (100 ms), images captured without the universal screen perturbation exhibit the highest recognition accuracy. However, further manual adjustments in exposure time, either increasing or decreasing, lead to decreased recognition accuracy. Reduced exposure times result in lower image quality, while increased exposure times cause overexposure. Both scenarios hinder the normal recognition capabilities of the target model. Similarly, Figure 5.24 examines the effects of varying ISO settings. The optimal automatic exposure setting is observed at ISO 800. Manually adjusting the ISO to either 400 or 1600 results in only a slight decrease in recognition accuracy for images captured without universal screen perturbation. Conversely, lowering the ISO further to 100

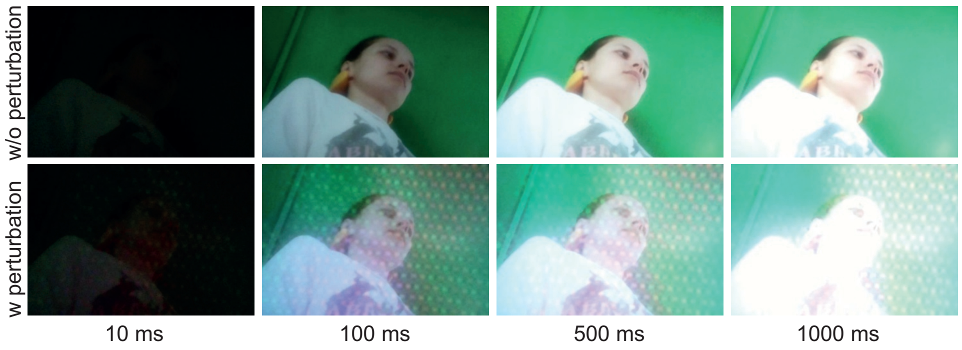


Figure 5.21: Example of images captured by USC with different exposure times. The first row shows images *without universal screen perturbation*, while the second row shows images *with universal screen perturbation*.

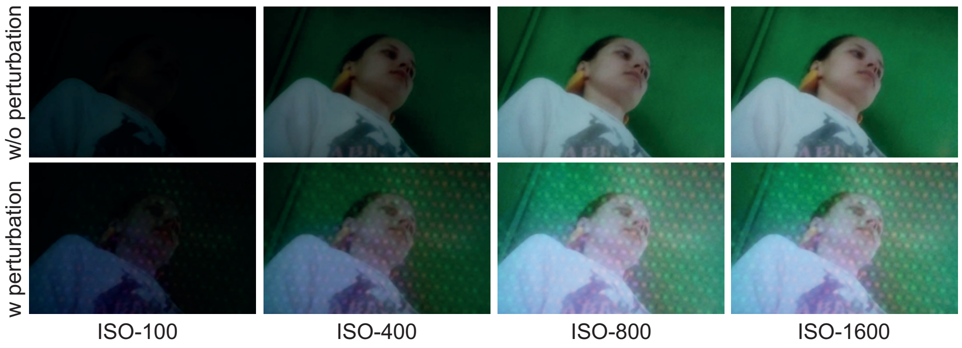


Figure 5.22: Example of images captured by USC with different ISO settings. The first row shows images *without universal screen perturbation*, while the second row shows images *with universal screen perturbation*.

or 200 leads to a rapid decline in accuracy, approaching zero. Notably, the recognition accuracy of images captured with universal screen perturbation remains consistently at 0% across all exposure time and ISO settings. These findings highlight the efficacy of the universal screen perturbation in maintaining its protective capabilities, regardless of manual exposure adjustments, thereby demonstrating its robustness against such manual exposure attacks.

DSSIM budget. We further evaluate the impact of the DSSIM budget on universal screen perturbation. The findings of this evaluation are presented in Figure 5.25. As the DSSIM perturbation budget (ρ) increases, there is a corresponding rise in the protection success rate. Specifically, at a ρ value of 0.005, the protection success rate decreases to a level comparable to that achieved with passive perturbation alone, around 20%. However, when ρ exceeds the threshold of 0.04, the protection success rate for universal screen perturbation increases to 100%. Consistent with prior research [153], this suggests that even higher DSSIM values (up to 0.2) are generally imperceptible to human eyes.

Impact of protected label density. The effectiveness of our universal screen perturbation is influenced by the number of labels designated for protection. Specifically, when the goal is to shield a single target label, the perturbation can focus on the distinct fea-

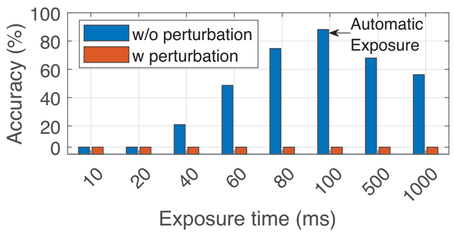


Figure 5.23: Identity recognition accuracy vs. exposure time.

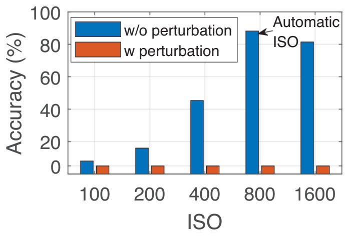


Figure 5.24: Identity recognition accuracy vs. ISO.

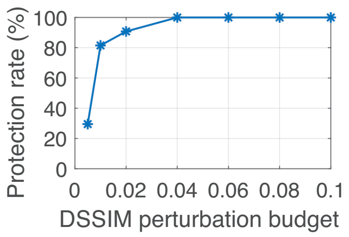


Figure 5.25: Protection rate vs. DSSIM budget.

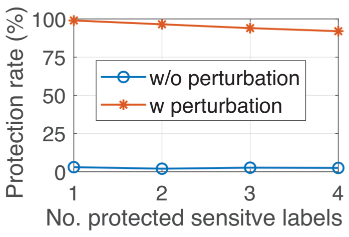


Figure 5.26: Protection rate vs. No. protected labels.

tures of that particular class within the feature space. However, as the number of protected labels increases, the perturbation strategy must adapt to find a common direction that diverges from the combined feature spaces of these multiple classes, as explained in [205]. The challenge grows as the number of classes to be protected increases, making it more difficult to identify such a common direction in the feature space, which leads to a decrease in the performance of universal screen perturbation. Figure 5.26 shows that the protection success rate drops from 100% when protecting 1 label to over 90% when 4 labels need to be protected.

5.5.4. REAL WORLD PROTECTION PERFORMANCE

To evaluate the practical applicability of Unicorn, we conduct a field test with real-world settings, which is a more challenging scenario.

Setup. Five participants (2 females and 3 males, aged between 20 and 40) were recruited. The participants were provided with stickers of different colors, shapes, and sizes (Figure 5.27(a)). Participants were asked to place these stickers in their living spaces with randomly generated password combinations, ensuring these combinations had no correlation with any passwords they currently use. Subsequently, participants were asked to take images containing these password notes. These images were captured using the front-facing cameras of their own smartphones, as well as using the USCs of full-screen smartphones provided by us. Specifically, user 1 was allowed to use AXON30 and MIX4, both with and without universal screen perturbation, to capture images as a reference group. All information collection and experimental protocols have been reviewed and approved by our Institutional Review Board (IRB). All taken images were checked by participants to ensure that no other identity-revealing or private objects were included.

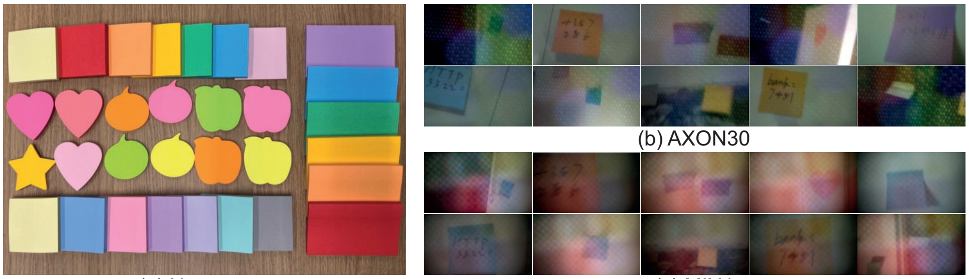

(a) Note setup (b) AXON30 (c) MIX4

Figure 5.27: (a) Password note setup. (b)-(c) Examples of collected password note images in real-world settings at different angles, distances, environments, and ambient light conditions.

Table 5.5: Universal screen perturbation is effective against large model-based vision APIs (Google and ChatGPT).

Target Model	w/o perturbation	w perturbation
Google Vision	0%	100%
ChatGPT Vision	0%	100%

These images are used to form our dataset, with explicit consent obtained from their respective participants. All participants were compensated for their time.

Results. The collected images showcase password notes captured under various conditions, including different angles, distances, environments, and lighting settings, as shown in Figures 5.27(b) and (c). The results from this field test, summarized in Table 5.3, align closely with our testbed findings. Universal screen perturbation typically achieves a protection success rate of about 90%. In comparison, all smartphones without universal screen perturbation exhibit protection success rates below 15% across different environments. These findings confirm the effectiveness and generalizability of the optimized universal screen perturbations: Images captured in the diverse living spaces of participants with universal screen perturbation lead to similar protection success rates.

5.5.5. PERFORMANCE AGAINST LARGE MODELS

Hackers may employ more powerful large-scale models to analyze captured images. In this section, we evaluate the efficacy of Unicorn against the proprietary models of Google Vision API [50] and ChatGPT Vision API [51].

Setup. The Google Vision API returns a list of labeled objects found in an image along with associated confidence scores, including the "note" label. We set the detection threshold at 0.5, meaning that a "note" is considered detected in an image if the API identifies a note object with a confidence level above 0.5. For the ChatGPT Vision API, we use the prompt *"What's in this image?"*[1] and determine whether "note" is mentioned in the response. A mention of "note" indicates successful recognition.

[1]We also tested other prompts from the official documentation [206], such as *"What's it?"* and *"Can you please describe this image?"* and found that they yielded the same results.

Results. The results of our evaluation are presented in Table 5.5. They indicate that without perturbation, both the Google Vision and ChatGPT Vision APIs successfully identify the "note" in the image. However, when universal screen perturbation is added, neither API is able to recognize the "note". Furthermore, responses from ChatGPT Vision do not include mentions of any other sensitive objects, such as "file" or "credit card". These test results demonstrate that our universal screen perturbation achieves a 100% protection rate against these two large-scale models.

5.5.6. USER STUDY

This section describes a user study conducted to assess visual impact on users of screen patterns implemented in smartphone screens. We recruited 30 participants of diverse genders (16 females and 14 males), ethnicities, nationalities, and age groups (from 18 to 45) through university mailing lists, social networks, and notice boards.

Study design. Our aim is to evaluate the perceptibility of optimized screen patterns embedded in smartphone screens and their effect on user experience during typical use. We utilize the two USC smartphones from Section 5.5.1 and let each participant use both devices. To minimize individual differences in perceiving screen perturbations, a within-subject study design [158] is employed. We consider two screen settings: (1) *Active Unicorn*, and (2) *Inactive Unicorn*. Participants are asked to use each smartphone for ten minutes, at least five minutes for each screen setting presented in random order. We also explore five status bar color states (red, green, blue, white, and black) as shown in Figure 5.28. The smartphones are randomly assigned a color for each user, ensuring each color appears at least ten times throughout the user study. This study was reviewed and approved by our IRB. All participants were compensated for their time.

Results. After interacting with the smartphones, participants completed a brief questionnaire regarding the noticeability of added screen patterns during smartphone usage. The findings reveal that none of the participants detected the presence of an optimized screen pattern on either full-screen smartphone. This result suggests that the implemented Unicorn does not noticeably impede the typical user experience, thereby confirming its inconspicuousness and user-friendliness.

5.5.7. ENERGY CONSUMPTION ANALYSIS

Power consumption model. The power modeling (in Equation (5.6)) of OLED screens at the pixel level for smartphones was proposed by [191]. This model has been validated for accuracy in practical applications by subsequent studies [207], [208]. According to [191], the blue subpixel exhibits the least energy efficiency. At full brightness, the power consumption of a blue subpixel is approximately 25 μW, compared to about 10 μW for both red and green subpixels. Although each color subpixel's power consumption is linearly proportional to its RGB value, the use of a standard gamma correction of 2.2 for signal input, which enhances display quality, creates a nonlinear relationship between OLED screen power consumption and gray levels, as depicted in Figure 5.29(a).

Results. To assess the energy efficiency of our universal screen perturbation, we use AXON30 as an example. We measure the device's power consumption in two states: *Screen-OFF state*, where the screen displays complete blackness with only the screen-

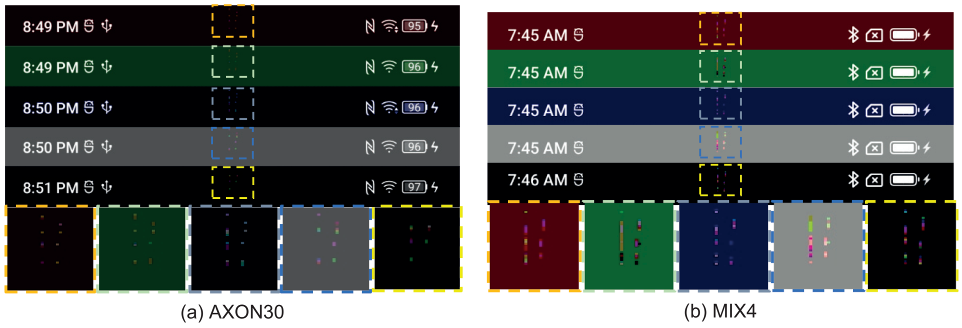


Figure 5.28: Screenshots of the full-screen smartphone status bar when running the Unicorn on (a) ZTE AXON30, (b) Xiaomi MIX4. The screen pattern is highlighted by color-dotted rectangles and magnified below. These screen-pixel changes are *imperceptible* for users.

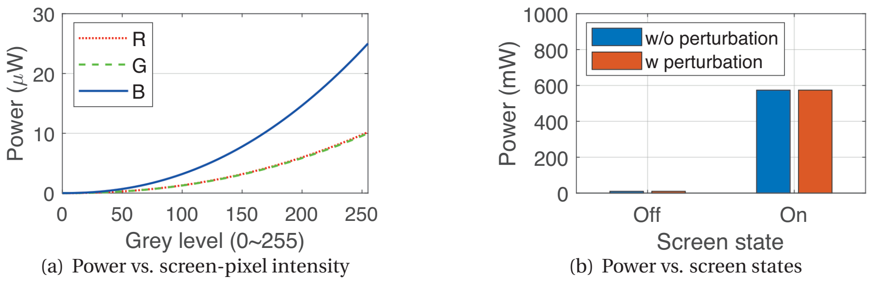


Figure 5.29: Power consumption analysis: (a) power consumption model on smartphones; (b) comparison of power consumption with and without universal screen perturbation.

pixels responsible for generating the screen perturbation activated,[2] and *Screen-ON state*, where the screen displays content along with a white status bar and additional screen-pixels activated for screen perturbation. In these scenarios, the numbers of screen-pixels modified in the Screen-OFF and Screen-ON states are 40 and 52, respectively. Given AXON30's screen resolution of 2460×1080, these modifications amount to merely about 0.0016% and 0.0020% of the total screen-pixels, respectively. A comparative analysis of the specific energy consumption in these scenarios is presented in Figure 5.29(b). The additional power consumption in the Screen-OFF and Screen-ON states is 0.086 mW and 0.138 mW, respectively. An additional power consumption of only 0.0240% is observed, which is negligible compared to the screen's unmodified state. These findings indicate only a marginal difference in energy usage between scenarios with and without screen perturbation. This observation highlights the energy efficiency of Unicorn, demonstrating that its deployment requires minimal additional energy costs.

[2]The Unicorn in the OFF state of the screen is similar to the always-on-display function commonly seen in smartphones [209].

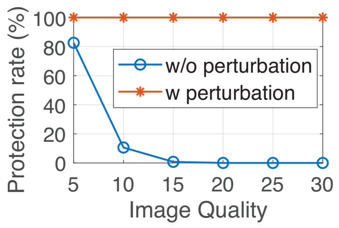


Figure 5.30: Normal classification accuracy increases as image quality increases using JPEG compression but protection success rate remains high.

Table 5.6: Protection success rate of universal screen perturbation with adversarial training on XGaze.

Model	Pristine image	Screen pattern 1	Screen pattern 2	Screen pattern 3
IncepResNet	16%	70%	**96%**	**94%**
MobileNet	20%	80%	**99%**	**97%**

5.5.8. Countermeasures

Below, we investigate possible countermeasures against Unicorn. Many defenses have been proposed to combat adversarial perturbations. These defenses can be categorized into *empirical defenses* [159], [161], [162] that lack formal robustness guarantees, and *certified defenses* [163], [164] that offer such assurances. We consider state-of-the-art defenses from both categories. For empirical defenses, we examine the classic image transformation defense method [210] and adversarial training [161], known as one of the most effective methods in this category. For certified defenses, we consider randomized smoothing [164], which is applicable to any classifier and scalable to large DNNs.

Image transformation. A simple technique to mitigate the impact of embedded perturbations is to transform captured images before inputting them into DNNs. One successful transformation-based defense method is JPEG compression [210]. The image quality ranges from 5 to 95 (lower value = higher compression). As shown in Figure 5.30, the protection success rate against screen perturbations remains at 100%, while the image compression more significantly degrades normal classification accuracy.

Adversarial training in [159], [161] uses adversarially perturbed training examples to improve a model's robustness. In our study, we conduct adversarial training with a dataset containing both pristine images and their corresponding perturbed images with screen perturbations. The training results are summarized in Table 5.6. The protection success rate increases for pristine images when perturbed images are included in the training dataset (cf. Table 5.1). However, the protection success rate for images with the same screen perturbations drops from 100% to 70%. It is important to note that this adversarial training only used perturbed images generated by screen pattern 1. Incorporating other screen perturbations (cf. Figure 5.15), derived from screen patterns 2 and 3, into the captured images maintains a protection success rate exceeding 90%. This indicates that users can effectively counter potential hacker-employed adversarial training by pre-computing and dynamically alternating among different optimized screen patterns. This strategy allows users to vary the screen perturbation applied, thereby

Table 5.7: Protection success rate of universal screen perturbation under randomized smoothing on XGaze.

Model	$\sigma = 0.5$		$\sigma = 1$	
	Pristine images	Perturbed images	Pristine images	Perturbed images
IncepResNet	0%	100%	0%	100%
MobileNet	0%	100%	8%	100%

strengthening the resilience of their devices against such adversarial training methods.

Randomized smoothing [163], [164] employs a technique where a base classifier H is used alongside a test input $\mathbf{x}$ to develop a robust smoothed classifier G. This process involves adding zero-mean isotropic Gaussian noise $\mathcal{N}(0, \sigma^2 \mathbf{I})$ to $\mathbf{x}$, with σ representing the standard deviation and $\mathbf{I}$ the identity matrix. Prior research [164] has indicated that the predicted label by classifier G remains stable even when the adversarial perturbation to $\mathbf{x}$ is constrained within certain limits. In practice, randomized smoothing generates M noisy variants of the test input $\mathbf{x}$ by adding random Gaussian noise, utilizes the base classifier H to predict labels for these noisy inputs, and then employs a majority vote from these M predictions as the final output. Moreover, we augment the training inputs with Gaussian noise to enhance the robustness of the smoothed classifier. Following [164], we set $M = 10^5$ and test σ at both 0.5 and 1. Our experimental results, presented in Table 5.7, reveal that the protection success rate of the smoothed classifier remains high under both $\sigma = 0.5$ and $\sigma = 1$ settings. This observation suggests that randomized smoothing primarily certifies smaller perturbations and therefore is not effective enough to mitigate the proposed screen perturbations.

5

5.6. RELATED WORK

Digital perturbations change pixel values in the digital domain, using algorithms like PGD [161] and C&W [172]. These perturbations, crafted by exploiting the gradient information of DNNs, are added onto digital images to deceive DNNs. Subsequent studies explored their transferability, examining if attacks on one model can affect another [127], [129]–[131]. These findings imply that perturbations have model-agnostic properties, enabling cross-model vulnerabilities. However, their real-world effectiveness is limited since digital perturbations are often too subtle to be detected by cameras due to output resolutions. Our work, in contrast, focuses on physical screen perturbations and their real-world effectiveness under various environmental conditions.

Physical perturbations. Not all images with digital perturbations, once printed and observed through a camera, maintain their deceptive characteristics [120], [211]. Thus, the physical perturbations that can be realized in the real world have gained significant interest [212]. Athalye et al. [213] created images with perturbations that resist various two-dimensional transformations, although these did not alter physical objects but only printed perturbed images. Such studies are crucial in understanding the performance of physical perturbations under diverse environmental conditions. The adversarial patch [214] is a notion that manifests as a physical sticker designed to disrupt autonomous driving [215] or facial recognition systems. Further advancements by Eykholt et al. [173] involved developing graffiti-like perturbations that persist under real-world conditions.

Subsequent innovations include embedding adversarial attacks in eyeglass frames to trick facial recognition systems [119], and using LED lamps to create subtle perturbations [176], [185], [186]. The creation of accessories such as attaching stickers to hats [174], and applying conspicuous patches to clothes [183] or wearable items [216] have presented alternative strategies. The SLAP [175], which uses a projector to generate controllable physical perturbations, is limited by the flatness and material of the projection surface. However, these patch-based perturbations often face real-world constraints as they require direct modification of each target object for effective deception.

Other methods, like attaching a transparent patch on a camera lens [202], [203], are less invasive but have drawbacks. These patches cause misclassification of all objects, which is essentially equivalent to removing the camera from the device or covering it, and also hinders the camera's normal visual function. However, with full-screen devices' emergence, their bezel-less, transparent screens offer a special advantage to their under-screen cameras. The screens act as dynamic mediums, adjusting color and brightness for screen-pixel-level attacks on the under-screen cameras. This method exhibits flexibility, as it allows users to temporarily switch the potentially short-lived screen perturbation on and off as they please, ensuring the usability of benign services of the under-screen camera. Also, it is robust since it does not require any hardware modifications.

5.7. CONCLUSION

Protecting sensitive information in passive indoor environments remains a critical yet unresolved issue. In this chapter, we propose Unicorn, a novel defense mechanism that creates physical perturbations using the transparent screen. Given the physical constraints of displaying specific patterns on various device screens under different conditions, we formulated an optimization problem. This problem was addressed by fitting a neural USC architecture and enhancing its robustness through a data augmentation pipeline. We evaluated Unicorn on two commercial full-screen smartphones across diverse environments and USC settings, and against state-of-the-art image classifiers such as IncepResNet, MobileNet, ResNet, as well as other large models deployed by Google and OpenAI. The results show that the generated universal screen perturbations are robust in real-world scenarios. Unicorn offers a flexible approach to image source protection, emerging as a key defense vector and prompting further research in USC security.

Future work. While the added pixel perturbation on captured images is indeed visually apparent, it is crucial that these changes do not disrupt the user's normal use of the smartphone, especially in scenarios where users utilize the proposed universal screen perturbations as a defense against unauthorized face recognition systems. This is particularly relevant in applications involving face verification, where users generally cannot access the photos taken during the verification process and thus do not care about perceptible perturbations on images. Therefore, our results demonstrate that the modifications made to the screen pixels are almost imperceptible. However, it is important to note that if the perturbation added by Unicorn to the image is imperceptible, it may also prevent hackers from realizing that the image has been protected. Future work for Unicorn could involve preventing the misuse of technologies such as Deepfake. For example, the imperceptible screen perturbation on the image could be used as a potential watermark technology for the user's device, helping third parties verify whether a photo was taken by an actual physical device or generated by generative AI models.

6 CONCLUSION

6.1. LOOKING BACK

The advancement in transparent screen technology has led to the development of full-screen devices, enhancing the screen-to-body ratio and user experience by integrating front-facing optical sensors under the transparent screens. This dissertation is the first academic attempt to apply transparent screen features to expand the connectivity and user privacy & security of full-screen devices in the context of through-screen computing. We have explored the primary challenges related to Visible Light Communication (VLC) by recovering attenuated modulated light information from LED luminaires with strong screen light interference, and realizing visual privacy protection by generating perturbation from transparent screens. To this end, we consider two subsystems of through-screen computing: *Through-Screen VLC* and *Screen Perturbation for Visual Privacy Protection*. We provide innovative solutions to overcome or utilize the screen's light interference and perturbation. In this chapter, we briefly discuss the accomplishments and contributions of this dissertation in Section 6.2, and discuss the advantages, applicability, broader implications, and future research directions in Section 6.3.

6.2. ACCOMPLISHMENTS AND CONTRIBUTIONS

In this section, we recapitulate the major contributions of this dissertation.

A. Redesigning Color Shift Keying for Through-Screen VLC

The bezel/narrow bezel on today's devices, which hosts various line-of-sight optical sensors, is disappearing. This trend not only forces optical sensors like ambient light sensors to be placed under the screen but also challenges the deployment of the emerging VLC technology, a paradigm for next-generation wireless communication. In Chapter 2, we proposed the concept of through-screen VLC with under-screen sensors, such as ambient light sensors placed under the Organic Light-Emitting Diode (OLED) screen. A transparent OLED screen greatly attenuates the intensity of passing-through light, degrading the efficiency of intensity-based VLC systems. To address this shortcoming, we exploited the color domain to build SpiderWeb, a through-screen VLC system. For the first time, we observed that an OLED screen introduces a color-pulling effect at under-screen sensors, which pulls the original modulated colors toward the screen's color. This severely affects the decoding of color-based VLC signals. Motivated by this observation and the structure of spider's web, we designed a new SpiderWeb Color-Shift Keying (SWebCSK) modulation scheme and a novel slope-based demodulation method, which can eliminate the color-pulling effect. We prototyped SpiderWeb with off-the-shelf hardware and evaluated its performance thoroughly under various scenarios. The results showed that compared to existing solutions, our approach can reduce the bit error rate by two orders of magnitude and achieve a 3.4× data rate.

B. Through-Screen VLC with Under-Screen Camera

In addition to under-screen color sensors with single-pixel output, recent studies have shown that the pervasive front camera of mobile devices is an ideal candidate to serve as the VLC receiver, which has higher signal sampling speed and native spatiotemporal characteristics. While promising, the full-screen trend of mobile devices also forces front cameras to be placed under the devices' screen – leading to the so-called Under-Screen Camera (USC). We observed severe performance degradation in VLC with USC: the transmission range is reduced from a few meters to merely 0.04 m, and the throughput is decreased by more than 90%. To address this issue, we leveraged the unique spatiotemporal characteristics of the rolling shutter effect on USC to design a pixel-sweeping algorithm to identify the sampling points with minimal interference from the transparent screen. We further proposed a novel slope-boosting demodulation method to deal with color shifts brought by leakage interference. We built a proof-of-concept prototype using two commercial smartphones. Experiment results showed that our proposed design reduces the BER by two orders of magnitude on average and improves the data rate by 59×: from 914 b/s to 54.43 kb/s. The transmission range was also extended by roughly 100×: from 0.04 m to 4.2 m.

C. Screen Perturbation for New Security Vector

The widespread use of mobile device cameras, especially in sensitive environments like home and office, has raised significant privacy concerns due to cameras' visual sensing capabilities. The advent of full-screen devices integrating USC, brings a new security vector. Although screens pose great challenges to under-screen sensors in receiving light signals, the USC can detect different screen perturbations produced by transparent screens when they are activated and deactivated in the captured photos. In Chapter 4, we first utilized the transparent screen's features to inconspicuously modify its displayed screen-pixels, imperceptible to human eyes but inducing perturbations on USC images. These screen perturbations affect deep learning models in image classification and face recognition. This proposed new type of screen perturbation is a double-edged sword for users. On one hand, malicious attackers can exploit it to embed adversarial perturbations into captured images, launching adversarial attacks aimed at disrupting the performance of legitimate face recognition systems or fooling image classification models. On the other hand, benign defenders, e.g., users, can leverage screen perturbation to protect themselves from unauthorized deep learning models. For instance, users can proactively activate specific screen pixels in use. Thus, any images covertly captured by the under-screen camera would not be correctly identified by unauthorized facial recognition models. We designed two methods, one-pixel and multiple-pixel perturbations, that can add screen perturbations to images captured by USC and successfully fool various deep learning models. Our evaluations on three commercial full-screen smartphones using testbed and synthesized datasets showed that screen perturbations significantly decrease the average image classification accuracy, dropping from 85% to 14% for one-pixel perturbation and 5.5% for multiple-pixel perturbation. For face recognition, the average accuracy drops from 91% to merely 1.8% and 0.25%, respectively.

D. An Universal Screen Perturbation to Secure USC

From the perspective of privacy protection, the image-specific perturbation method is not sufficient for real-world scenarios, as it requires a pristine image to generate image perturbations for each scene and object, making it ineffective for dynamic and complex real-world situations. Instead, we need to develop a universal, scene-independent active perturbation without the need for a pristine image. Furthermore, the developed method should be optimized for robustness and effectiveness in various conditions, including different shooting distances, angles, and camera settings. Universal screen perturbations offer greater scalability and generalizability compared to image-specific screen perturbations, making them more suitable for real-world physical perturbations and deceiving classifiers in diverse contexts. In Chapter 5, we design Unicorn, a novel system that subtly alters the pixels on the transparent screen to create perturbation in images captured by USC. These modifications are imperceptible to human eyes but effectively induce universal screen perturbations on USC images without any hardware modification. The inclusion of perturbations renders images with sensitive content, such as personal identity or password information, unrecognizable by unauthorized image classification models. We formulated Unicorn as an optimization problem and developed a differentiable neural USC architecture to solve it. Extensive real-world experiments with two commercial smartphones across various settings confirmed Unicorn's robustness, offering over 90% success rates in protecting against state-of-the-art neural networks, including advanced image classification services from Google and ChatGPT Vision APIs.

In this dissertation, we have introduced transparent screens and full-screen devices as a new field of study. Through-screen computing is an emerging area that, while building on decades-old mobile device technology, offers new dimensions and possibilities for innovation. This interdisciplinary technology integrates material, structural, mechanical, electrical, communications, and importantly, embedded systems and software engineering. There are numerous opportunities and challenges to address in the development of through-screen computing. This dissertation represents the first step toward realizing this vision and exploring the potential of through-screen computing. We have demonstrated how to progress towards making this vision a reality. While the field is vast, this dissertation has only focused on the aspects of embedded and networked systems. Next, we discuss other opportunities and potentials of through-screen computing.

6.3. NEXT HORIZON

The unique features of transparent screens, along with their interdisciplinary nature, offer fertile ground for the ubiquitous computing community to conceive new problems and solutions or revisit existing ones in the new context of through-screen computing.

While not all the potential research directions can be presented here, we discuss some examples that would achieve breakthroughs and become integral to future daily life.

6.3.1. UNDER-SCREEN IMAGING

Maintaining the full functionality of a camera after placing it under a transparent screen is challenging. The imaging quality of a camera will be severely degraded due to the low light transmittance and diffraction effects. As a result, the captured images are noisy and blurry. While improving the user experience by providing a full screen, USC sacrifices the quality of photography, face recognition, and other vision tasks. Restoring and enhancing the images captured by USC are essential. Methods to mitigate these effects include techniques to restore diminished spatial frequencies in the captured images [99] and conventional deep neural networks for correcting the significantly blurred images and enhancing the SNR in the images [11]–[13], [217]. This is an important research direction of through-screen computing. Further work could continue to restore the quality of images captured by USC through advanced *under-screen computational imaging* system design driven by large foundation models.

6.3.2. TRANSPARENT 'SCREEN' FOR RF COMMUNICATIONS

With the advent of self-driving vehicles, there is a growing need for more reliable wireless connectivity. Besides enabling through-screen VLC, transparent screens can be designed as Radio-Frequency (RF) antennas to receive modulated RF signals. Transparent screen antennas offer excellent signal reception capability. Due to their transparency and ultra-thin nature, these antennas provide placement options that traditional antennas cannot match. They can be integrated into vehicle windows, sunroofs, or windshields to deliver connectivity in areas where it is most needed, which enables unparalleled high conductivity while retaining exceptional transparency. For example, the world's first transparent screen antenna for extended reality applications was presented at AWE USA 2022, as shown in Figure 6.1. Moreover, Reconfigurable Intelligent Surface (RIS) has been proposed to intelligently control wireless channels by reflecting or refracting RF signals. Transparent screens can be designed for RIS and seamlessly integrated into windows without affecting their transparency [218]. For instance, Figure 6.2 shows how transparent screen windows help signals reflect at design-specific angles to eliminate blind spots in communications of street blocks and help RF signals penetrate throughout buildings.

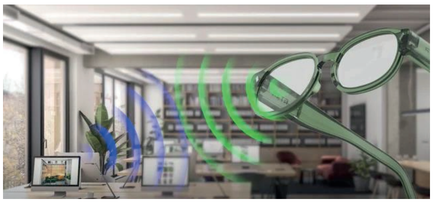


Figure 6.1: Transparent 5G antenna in ARfusion smart lens [219].

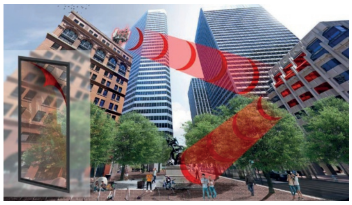


Figure 6.2: Transparent screen antenna reflects signal to cover dead zones [220].

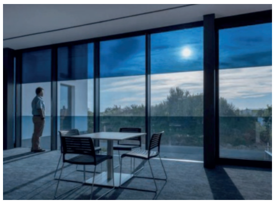


Figure 6.3: Sage smart window with transparent screen [226].

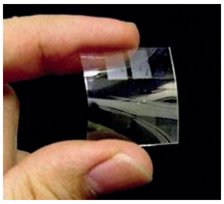


Figure 6.4: Flexible, see-through transparent battery [227].

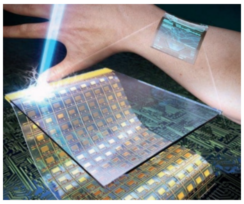


Figure 6.5: Transparent flexible screens for skin-like wearables [228].

6.3.3. Charging Behind/On Transparent Screen

Energy harvesting could replace traditional batteries in low-end mobile devices [221], [222]. The energy can be harvested from various sources, including light, radio waves, vibrations, temperature differentials, wind, etc. Transparent screen can enable energy harvesting behind/on the screen, promoting battery-free through-screen computing. Specifically, the transparent screen can absorb and harvest energy from infrared and ultraviolet light, as well as portions of visible light, while allowing the remaining light to pass through [223], [224]. Therefore, transparent screens can be used to design smart windows, as shown in Figure 6.3. Meanwhile, transparent screen can maintain their electrochemical properties under wearable conditions, and thus have great potential in acting as batteries [225], as shown in Figure 6.4. With these features, a transparent screen can be fitted to the entire mobile device body, including on top of the device, to simultaneously display information, harvest energy, and store energy to enable battery-free through-screen computing. Besides, transparent screen can be seamlessly integrated into various parts of the human body or accessories like arms and necks without affecting the wearer's vision (See Figure 6.5). This through-screen computing on the human body can empower applications such as motion detection and health monitoring.

6.3.4. More Natural Interaction When Facing the Screen

Positioning the camera behind the screen eliminates the constraints imposed by the device's bezel, notch, or hole, thereby enhancing gaze awareness during video calls and conferencing–key functions of a front-facing camera. In conventional setups, cameras are often positioned at the top or bottom of the screen, causing an offset from the natural interaction where the user's eyes focus on the screen's center. These offset camera positions disrupt natural eye contact, as participants must choose between looking directly at the camera or at the screen, resulting in a loss of subtle non-verbal cues [229]. This misalignment between the camera's perspective and the on-screen image of the remote participant persists in modern video conferencing systems. We could optimize camera placement behind the screen based on the UI design of traditional video apps. For example, by positioning both the camera and the speaker's face at the center of the screen, the offset between the user's viewpoint and the displayed face of the speaker can be minimized, enabling a more natural eye contact experience. We envision that this new through-screen computing paradigm could refine the user's on-screen appearance and perspective by optimizing the positions of under-screen sensors and applying gaze correction technology, creating a more natural and immersive interaction experience.

6.3.5. THROUGH-SCREEN SENSING WITH REFLECTED SCREEN LIGHT

Since the light spectrum is license-free and fine-grained optical sensors such as cameras are widely deployed on mobile devices, pervasive sensing with light has received significant interest. Existing sensing systems with light rely on identifying the shadow caused by a moving object blocking the light source [230], [231] or the reflected light [232], [233]. Future work could leverage the screen as the light source and the under-screen sensors as the receiver to enable *through-screen sensing with reflected screen light*: under-screen sensors sense various information from the light emitted by the screen and reflected by objects. The advantage is that an additional active light source is not required. Below we discuss some directions on this topic.

Gesture recognition. One direction is to sense people's movements (motions, postures, positions, etc.) in a two-dimensional or three-dimensional (3D) plane, utilizing the light emitted by smartphone screens or TVs. The users can perform the gestures in front of a device and the gestures can be perceived and recognized by under-screen sensors, enabling convenient and non-invasive control of future mobile devices. This can also be combined with popular motion-sensing games to achieve higher recognition accuracy and finer resolution in 3D human body modeling.

Novel authentication scheme. One existing novel under-screen authentication scheme is under-screen fingerprint sensors. We can design similar authentication schemes, for instance, facial liveness detection using an under-screen camera. The screen can emit light with carefully designed brightness or color; our faces with unique shapes and flatness can reflect screen light differently to the under-screen sensors, enabling novel face authentication. Another important biometric measure is cardiac patterns, uniquely defined by the heart, lung, and vein structures of individuals. These cardiac patterns can be obtained with a photoplethysmogram (PPG), which measures changes in blood volume via light absorption. Traditionally, PPG is obtained using a pulse oximeter on a finger, consisting of a small LED that emits light and an optical sensor that captures how much light is absorbed through the body. With through-screen sensing with reflected light, the screen could utilize its RGB pixels to emit the light, without using an additional LED.

Integrated sensing and communication behind screen. In future through-screen communication, there will still be a need for sensing. It is essential to design new protocols in through-screen computing to integrate communication and sensing. At the communication level, overcoming the dynamic interference of the screen and designing a scheme to address signal interruptions caused by real-world actions that need to be sensed is necessary. At the sensing level, overcoming the influence of dynamic modulated signals and screen interference to complete sensing is essential. Hence, new communication signal processing and intelligent sensing recognition schemes must be designed. We could develop a hierarchical system to accomplish various sensing tasks alongside communication functions.

6.3.6. RISK AND PROTECTION

Visual sensors are not only pervasive in the cameras of mobile devices but also ubiquitous in Internet of Things (IoT) devices, including security cameras, doorbell cameras, smart TVs [234], refrigerators [235], pet monitors [236], and more. However, visual sensors are bringing privacy issues in the advanced artificial intelligence era. In today's digital world, the visual sensors on mobile devices have become our eyes to connect, work, and share moments. However, this incredible convenience has pitfalls. Privacy concerns have appeared as hackers could turn any visual sensors into instruments of intrusion. Networked mobile devices are often neither transparent nor secure, with recorded videos being subject to various vulnerabilities [237] and unauthorized leakages. Nest security cameras might leak videos about the former user [238], and when Xiaomi IoT cameras are connected to the Google Hub, the monitoring images might show scenes of other users [239]. Cybercriminals, equipped with sophisticated artificial intelligence like deep neural networks, can now mine images for valuable information. This highlights the pressing need for innovative privacy solutions that not only protect us but also maintain the aesthetic integrity of our devices.

On the other hand, the idea of programmable apertures involves using an amplitude/phase mask to code the aperture of the camera lens [240], [241]. They are usually used to construct a new type of programmable-aperture-based camera that encodes the measurements of the scene and requires carefully designed computational algorithms to decode the target scene. Programmable apertures have potentially wide-ranging privacy protection uses, but they are typically only available in custom cameras. Transparent screens can offer an alternative flexible protection solution. Transparent screens can also be considered a special type of programmable aperture due to their self-luminous nature, allowing them to dynamically display screen pixels to change the scenes captured on cameras. The proposed Screen Perturbations in this dissertation can be extended beyond under-screen cameras, with potential applications involving transparent screen cover for any visual device to create a visual shield. For example, users could attach a transparent screen over any visual sensors to obscure sensitive information while preserving core visual sensing functions, such as gesture or motion recognition. Overall, the transparent screen cover has the potential to unlock a range of new applications and possibilities, building trustworthy mobile devices with any visual sensors. The list below provides a glimpse into the near future:

Side information leakage. Side-channel attacks exploit unintended leakage information from mobile devices to launch unexpected attacks and violate user privacy. In through-screen computing, the advent of transparent screens not only provides opportunities to implement new side-channel attacks but also brings new possibilities for preventing potential side-channel attacks and protecting user privacy. Below, we outline potential side information leakage in through-screen computing:

- *Side App activity leakage.* Wireless data transmission patterns could reveal the activities of common mobile apps (e.g., YouTube, Facebook, and WhatsApp) across different categories: video, music, social media, communication, and gaming [242], [243]. The screen, as a crucial interface, can also be used to reveal the activity of mobile devices. UI design is prioritized for each app upon launch, and under-screen sensors can track screen light emissions during app usage to infer user activities on mobile devices, thereby compromising user privacy. The primary reasons for this include: 1) Each app has a distinct design style and color scheme. For example, YouTube predominantly uses red, Facebook uses blue, and WhatsApp uses green. By detecting the main color of the screen display, under-screen sensors can infer the app in use. 2) Apps exhibit different interactive behaviors. Social media apps, for instance, do not update displays as frequently as gaming or video apps and often display static content. The content presentation and user interaction logic of these apps can further expose app activity on the screen. Additionally, distinctive startup animations of different apps can help hackers infer app activity history solely by analyzing the reflected screen light.

- *Side visual leakage.* Side visual leakage refers to sensors whose main function is not visual imaging, such as ambient light sensors and Time-of-Flight (ToF) sensors, which can recover visual information from their readings after applying signal processing algorithms. Recent studies [244] show that ambient light sensors could present imaging privacy threats. Also, ToF measurements not only can display depth information, such as the three-dimensional structure of a face used in Face-ID, but also can be manipulated to expose visual information, including key facial details of users [245], [246]. To address these leakage concerns, the screen can slightly and infrequently change its brightness, color, and displayed contents, modifying the under-screen sensor's single-pixel readings in through-screen computing. This slight modification will not trigger the entire screen brightness or alter the measured Face-ID but will prevent the optical sensor's low-resolution and even single-pixel readings from being restored to a high-resolution visual image through generative AI models.

Under-screen hidden spy camera detection. Hidden spy cameras monitoring people in private spaces have increasingly become a worldwide problem. For example, in South Korea alone, a total of 5541 spycam-related crimes were reported in 2021 [247]. Locations such as Airbnb accommodations are particularly attractive for attackers to install these hidden cameras. Recent surveys show that 11% rental accommodation visitors had found a hidden camera in one of their rental visits [248]. Further, hackers can customize devices to hide spy cameras behind transparent screens, such as large TVs and monitors. The transparent screens allow light to reach the under-screen spy camera while displaying content, making the camera unnoticeable to the human eye, especially when content like videos is being displayed. This makes the under-screen camera a new attack vector in the field of spy camera crimes. Detecting hidden under-screen spy cameras is a challenging task and needs to be addressed. There are research opportunities to detect under-screen spy cameras using network traffic [249], [250], thermal emissions [251], or interesting optical reflections [252] due to the limited field of view for combined screen and under-screen spy cameras.

6.3.7. SPATIAL INTELLIGENCE OVER SCREEN

Spatial intelligence represents advanced AI capabilities, enabling large models to perform complex visual reasoning and action planning, akin to human abilities. High-resolution transparent screens can create realistic AR objects and scenes [253], seamlessly integrating with the real world to facilitate spatial intelligence. Depth information is another critical factor for large models to understand the 3D world, significantly influencing their interaction with objects in space. Transparent screens can aid under-screen sensors in obtaining depth information, enhancing spatial intelligence. Innovative 3D holographic AR glasses [254], using transparent screens and under-screen sensors, can present vivid full-color 3D objects, offering immersive experiences without bulky devices like Apple's Vision Pro or Meta's Quest headset. By using the feedback from the under-screen sensor and the AI algorithm to calibrate the depth information, we can better present the future of spatial intelligence. This novel screen application promises substantial impacts across gaming, social media, training, and education.

Epilogue

Transparent screen, like a magical “window”, has quietly broadened our horizons, transforming the way we interact with the world. This journey of development has been marked by relentless technological advancement and the ingenious integration of design aesthetics. Each leap in pixel design and each material innovation represents the splendid bloom of technological power in the microscopic world. It has not only revolutionized our visual experience but also fundamentally reshaped human interactions. Let us follow the light through the transparent screen into the vibrant and colorful underscreen world, exploring the infinite possibilities behind the “window” that lies ahead.

ACKNOWLEDGEMENTS

This thesis would not have been possible without the care and support of many people. Your guidance and encouragement have been like a light passing through the darkness, illuminating the path ahead. I would like to take this opportunity to express my heartfelt gratitude to everyone who has shared this journey with me.

First and foremost, I am truly grateful for the supervision and guidance of my daily supervisor and promoter, Dr. Qing Wang. You taught me to dream big and to strive step by step to pursue those dreams. I was always the anxious and unconfident one—spending two months just identifying my future research direction and fearing that I might disappoint you by messing things up. Despite these typical early-stage PhD anxieties, you never criticized me for my rash actions—such as sharing a promising research idea at 4 AM—or for my stagnant progress, like spending several months just to get sensor readings. Instead, you patiently analyzed the feasibility of my proposed research directions, thoroughly cared for my well-being, and continuously encouraged me to move forward. Without your encouragement, I might have fallen by the wayside early on. Your unwavering belief in me taught me resilience in facing frustrating research setbacks and repeated rejections from top-tier conferences. Looking back over the past four years, I am deeply thankful that you led me into the field of mobile systems, allowing me to freely enjoy the research process of chasing my dreams. You have demonstrated the profound influence supervisors can have on their PhD students, setting a great all-around example—not only in academics but in all aspects of life.

I also extend my heartfelt thanks to my promoter, Dr. Marco Antonio Zúñiga Zamalloa. I am profoundly grateful for your expertise and unwavering support in shaping this thesis. Over the past four years, you have consistently listened patiently during our meetings and provided invaluable feedback on writing, communication, presentation, and networking—key aspects of my personal development. Through your guidance, I have learned how to grow in all aspects. Your positive feedback and sincere advice for my future career development have played a crucial role in my professional journey. Thank you, Marco, for your words at EWSN 2024—I will always keep them in mind!

I also wish to extend my gratitude to Prof. dr. Koen Langendoen for your understanding and wisdom. As an office neighbor (perhaps an annoying one, I know—apologies for the occasional coughing sounds or forgotten open windows), I greatly appreciate your patience. I admire your exceptional critical thinking and problem-solving skills, which you demonstrate during seminars on a wide range of topics. I appreciate that your door is always open to me, giving me time and guidance to find solutions when I encounter problems or dilemmas. During my first group meeting, when I naively expressed my goal to work hard and achieve high "grades" as a PhD student, you kindly clarified that the true measure of success lies in academic contributions and impact. Thank you, Koen, for sharing your wisdom and personal experiences during my most vulnerable moments, providing comfort and insight when I needed it most.

I am fortunate to have guidance from another supervisor, Dr. Guohao Lan. You taught me to seize every opportunity to achieve excellence, instead of being satisfied with meeting basic goals. I once complained to you that user studies always involved so many tedious steps. You shared your past experience with me and helped me understand the pursuit of rigor and excellence in the research process. Eventually, I grew to like this unique process of conducting human-centered experiments and creating a positive impact on users and society. I also sincerely appreciate your invaluable suggestions for my career development and your encouragement to embrace opportunities bravely, reminding me that if I don't try, I'll never know the result.

I am also very grateful to my collaborator, Dr. Jie Xiong, who is intelligent and humble. You taught me to think deeply about the essence of things and how to tell a story that everyone appreciates by grasping that essence. It was your guidance that made me reflect on whether I had truly thought through the fundamentals of my research and what principles I relied on to solve tricky and significant problems. I also want to thank another collaborator, Dr. Jinyuan Jia. Your insights into the nuances of the security community provided me with new perspectives that shaped my approach to research.

Besides my supervisors and collaborators, I am grateful to the chairperson and my doctoral committee members: Prof. dr. Eberhard Karl Albert Gill, Prof. dr. George Smaragdakis, Prof. dr. Danny Hughes, Prof. dr. Nirvana Meratnia, Prof. dr. Haitham Al Hassanieh, and Prof. dr. Mathijs Michiel de Weerdt. It is a true honor to have esteemed researchers like you as part of my PhD committee. Thank you for taking your precious time to review my dissertation, and offering invaluable evaluations and feedback.

I also want to express my thanks to the support I received from the Marie-Curie project, ENLIGHT'EM, and extend my thanks to Dr. Domenico Giustiniano, Prof. dr. Harald Haas, and Dr. Rui Bian for your academic insights and thoughtful advice on my research and career development. Thanks to the other early-stage researchers in the project: Jagdeep, Behnaz, Ömer, Kien, Gianluca, Janis, Dayrene, Burak, Tilahun, Khadijeh, Daniel, Nasir... whose camaraderie made this journey unforgettable. I will fondly remember every beer shared and the bond we built during joyful conversations.

I wish to thank other professors from the Embedded and Networked Systems group: Dr. Przemysław (Przemek) Lesiczka, Dr. Arish Asadi, Dr. Qun Song, Prof. dr. Fernando Kuipers, Dr. R. Venkatesha Prasad, and Dr. George Iosifidis, who have all enriched my research journey with your insights. Przemek, thank you for nice photos and posts you took for me at conferences. You inspire me to be more confident and showcase our achievements. Thanks to the Mahjong group: Ran, Mingkun, and Mengyuan. Our shared interest in Mahjong brought moments of happiness—there's truly no problem that a "Gang Shang Hua" can't solve. Mingkun also served as my beginner fitness coach; without your initial careful guidance, I couldn't have successfully lost 25 kg in six months. To my foodie group—Lingyu, Eric, Hao, Shenxiu, Yukuang, Shun, and Bo—thank you for the countless dining adventures. Lingyu, my best foodie partner, who would have thought we'd catch the earliest NS train to Düsseldorf for BBQ and return to Delft at midnight? Our countless trips to Amsterdam and Rotterdam for Yuan's Hot Pot will always remain cherished memories. Thanks to Gabe for organizing wonderful parties that delighted everyone, and to visiting PhD students—Longyu, Xiujuan, and Bingyang, whose participation brought more joy to our gatherings. Our thrilling party game "Werewolf" will always be a cherished memory. Thanks to Keyarash for the times we did TA work together and shared meaningful conversations that happened in the lab. Thanks to Miguel, a genius in learning languages—gracias for sharing different cultures with us. Thanks to Talia for taking care of me during my first year of studying abroad and helping me adapt to this new environment. Thanks to Vivian, Andrian, and Anup for the valuable conversations during coffee breaks and lunches. To my former roommates, Vineet and Girish, thank you for sharing your academic experiences and offering travel tips. Thanks to my supervised master students Niels and Tolga for every meaningful and interesting brainstorming session. And to so many other colleagues—your friendship, the countless coffee breaks, parties, and the wonderful moments we shared have added the most beautiful and precious memories to my time at TU Delft. Beyond these joyful times, I am also deeply grateful to many of you for offering comfort and encouragement during moments when I received frustrating and discouraging news. Together with the happy memories we created, your support helped me pick myself up and keep moving forward.

Thanks to our wonderful secretaries, Pam, Kim, and Minaksie. Your quick responses and consistent support for all inquiries have been invaluable. Thanks also for organizing engaging activities that brought joy to our group. I am also grateful to Mrs. Hongmei Song (Qing's wife) for your warm help in driving my experimental equipment from home to the labs, and the delicious food (especially my favorite shrimp) you prepared.

I am grateful for the precious friends I met after coming to Delft: Minfei Liang, Yubao Zhou, Xinling Yue, Zichao Li, Shengnan Zhang, Shuaiqi Yuan, Tianlong Jia, Yidong Gan, Ze Chang, Shilun Zhang, and Siyuan Wang. As someone who is clumsy and timid, I feel fortunate to have formed such meaningful friendships. Your incredible cooking, late-night League of Legends gaming sessions, and lively card games at parties turned ordinary days into extraordinary memories. I hope all of you achieve your dreams.

I express my deepest gratitude to my family. Thank you to my father for your unconditional financial support and to my mother for sending me chili powder over the past four years, which became a bridge that connected me with my friends during homesick times. Your sacrifices have been instrumental in shaping this journey. Thanks to my grandparents, who raised me in a 20 m^2 rented room and taught me kindness and integrity. Although I don't often contact you, I always keep in mind that you have given me your best love and belief that I am the best, giving me the freedom to grow. Even though I clearly know that I am not, I still hope to make you proud in all my endeavors.

Thank you to my girlfriend, Pei (Gupplia/Guppy/Peggy) Tian. Meeting you at EWSN and SenSys 2021 felt like our destiny. Over these three years, we have grown through arguments, reconciliations, and shared dreams. Your companionship has been my compass, guiding me through this journey and creating cherished memories across Europe. I believe you will continue to lead our journey as we explore the world.

Finally, I thank myself for persevering through the ups and downs of the journey to reach this PhD milestone. This is not the endpoint but rather a new beginning. I hope to continue facing the unknown fearlessly and remaining curious in my explorations.

Hanting (Damon) Ye
Delft, December 2024

BIBLIOGRAPHY

[1] Statista. "Charted: There are more mobile phones than people in the world." (2023), [Online]. Available: https://www.weforum.org/agenda/2023/04/charted-there-are-more-phones-than-people-in-the-world/.

[2] P.Gagnon., "Presentation at OLED World Summit 2019," *AMOLED Market & Technology Trend*, 2019.

[3] V. D. J. Evans, X. Jiang, A. E. Rubin, M. Hershenson, and X. Miao, *Optical sensors disposed beneath the display of an electronic device*, US Patent App. 15/336,620, 2017.

[4] LG. "Transparent OLED." (2023), [Online]. Available: https://www.oledspace.com/en/oled-inside/experience/a-new-world-beyond-the-display-transparent-oled/.

[5] LEDinside. "Osram Joins Micro LED Project to Develop Transparent Automotive Display Applications." (2019), [Online]. Available: https://www.ledinside.com/news/2019/4/osram_joins_microled_project_develop_transparent_automotive_display_applications.

[6] Z. Wang, Y. Chang, Q. Wang, Y. Zhang, J. Qiu, and M. Helander, "55-1: Invited paper: Self-assembled cathode patterning in AMOLED for under-display camera," in *SID Symposium Digest of Technical Papers*, 2020.

[7] Samsung. "What is the Under Display Camera (UDC) on the Galaxy Z Fold 4?" (2022), [Online]. Available: https://www.samsung.com/uk/support/mobile-devices/what-is-the-udc-under-display-camera-of-the-galaxy-z-fold4.

[8] ams. "Ams launches optical sensor which measures ambient light from behind a smartphone's OLED screen." (2019), [Online]. Available: https://ams.com/-/ams-launches-optical-sensor-which-measures-ambient-light-from-behind-a-smartphone-s-oled-scre-1.

[9] K. Kwon, E. Kang, S. Lee, *et al.*, "Controllable image restoration for under-display camera in smartphones," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2021.

[10] Y. Zhou, D. Ren, N. Emerton, S. Lim, and T. Large, "Image restoration for under-display camera," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2021.

[11] H. Panikkasseril Sethumadhavan, D. Puthussery, M. Kuriakose, and J. Charangatt Victor, "Transform domain pyramidal dilated convolution networks for restoration of under display camera images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[12] V. Sundar, S. Hegde, D. Kothandaraman, and K. Mitra, “Deep atrous guided filter for image restoration in under display cameras,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[13] S. Lim, Y. Zhou, N. Emerton, T. Large, and S. Bathiche, “Image restoration for display-integrated camera,” in *SID Symposium Digest of Technical Papers*, 2020.

[14] A. Yang and A. C. Sankaranarayanan, “Designing display pixel layouts for under-panel cameras,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[15] Y. Zhou, M. Kwan, K. Tolentino, *et al.*, “UDC 2020 challenge on image restoration of under-display camera: Methods and results,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[16] “IEEE Standard for Local and Metropolitan Area Networks–Part 15.7: Short-Range Optical Wireless Communications,” *IEEE Std 802.15.7-2018*, 2019.

[17] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[18] 6G Flagship. “Key Drivers and Research Challenges for 6G Ubiquitous Wireless Intelligence.” (2019), [Online]. Available: `https://www.6gflagship.com/key-drivers-and-research-challenges-for-6g-ubiquitous-wireless-intelligence/`.

[19] F. Tariq, M. R. A. Khandaker, K.-K. Wong, M. A. Imran, M. Bennis, and M. Debbah, “A speculative study on 6G,” *IEEE Wireless Communications*, 2020.

[20] pureLiFi. “World’s First LiFi Integrated Devices Shown Off by pureLiFi at MWC.” (2018), [Online]. Available: `https://beebom.com/purelifi-demonstrate-the-worlds-first-lifi-integrated-devices-at-mwc/`.

[21] pureLiFi. “pureLiFi launches Light Antenna ONE, a LiFi module ready for integration into billions of devices at MWC Barcelona.” (2023), [Online]. Available: `https://www.purelifi.com/purelifi-launches-light-antenna-one-a-lifi-module-ready-for-integration-into-billions-of-devices-at-mwc-barcelona/`.

[22] M. Raquibuzzaman, K. M. N. Akbar, and M. M. Uddin, “Simple, low cost microcontroller based data communication system using ambient light sensor for VLC,” in *2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, IEEE, 2016.

[23] J. Green, H. Pérez-Olivas, S. Martínez-Díaz, *et al.*, “VLC-beacon detection with an under-sampled ambient light sensor,” *Optics Communications*, 2017.

[24] Z. Wang, Z. Yang, Q. Huang, L. Yang, and Q. Zhang, “ALS-P: Light weight visible light positioning via ambient light sensor,” in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2019.

[25] C. Danakis, M. Afgani, G. Povey, I. Underwood, and H. Haas, “Using a CMOS camera sensor for visible light communication,” in *2012 IEEE Globecom Workshops*, 2012.

[26] H.-Y. Lee, H.-M. Lin, Y.-L. Wei, H.-I. Wu, H.-M. Tsai, and K. C.-J. Lin, “Rollinglight: Enabling line-of-sight light-to-camera communications,” in *Proceedings of the ACM Mobile Systems, Applications, and Services (MobiSys)*, 2015.

[27] J. Ferrandiz-Lahuerta, D. Camps-Mur, and J. Paradells-Aspas, “A reliable asynchronous protocol for VLC communications based on the rolling shutter effect,” in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, 2015.

[28] Y. Yang, J. Hao, and J. Luo, “Ceilingtalk: Lightweight indoor broadcast through LED-camera communication,” *IEEE Transactions on Mobile Computing*, 2017.

[29] Y. Yang, J. Nie, and J. Luo, “Reflexcode: Coding with superposed reflection light for LED-camera communication,” in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2017.

[30] P. Hu, P. H. Pathak, X. Feng, H. Fu, and P. Mohapatra, “Colorbars: Increasing data rate of led-to-camera communication using color shift keying,” in *Proceedings of the ACM Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2015.

[31] pureLiFi. “LiFi-XC.” (2018), [Online]. Available: `https://www.purelifi.com/products/lifi-xc/`.

[32] A. Senior, S. Pankanti, A. Hampapur, *et al.*, “Enabling video privacy through computer vision,” *IEEE Security & Privacy*, 2005.

[33] D. N. Serpanos and A. Papalambrou, “Security and privacy in distributed smart cameras,” *Proceedings of the IEEE*, 2008.

[34] T. Winkler and B. Rinner, “Security and privacy protection in visual sensor networks: A survey,” *ACM Computing Surveys (CSUR)*, 2014.

[35] H. Yu, J. Lim, K. Kim, and S.-B. Lee, “Pinto: Enabling video privacy for commodity iot cameras,” in *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, 2018.

[36] J. Li, Z. Li, G. Tyson, and G. Xie, “Your privilege gives your privacy away: An analysis of a home security camera service,” in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2020.

[37] T. Winkler and B. Rinner, “A systematic approach towards user-centric privacy and security for smart camera networks,” in *Proceedings of the ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC)*, 2010.

[38] T. Winkler and B. Rinner, “Trustcam: Security and privacy-protection for an embedded smart camera based on trusted computing,” in *Proceedings of the IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2010.

[39] 4Customize. “5 Ways to Cover Your Laptop Camera.” (2023), [Online]. Available: `https://www.4customize.com/5-ways-to-cover-your-laptop-camera/`.

[40] CBC News. “Tape over your laptop camera? Why it might not be as paranoid as it seems.” (2016), [Online]. Available: `https://www.cbc.ca/news/science/laptop-camera-security-tape-1.3649678`.

[41] F. Pittaluga and S. J. Koppal, “Pre-capture privacy for small vision sensors,” *IEEE Transactions on pattern analysis and machine intelligence*, 2016.

[42] J. Tan, S. S. Khan, V. Boominathan, *et al.*, “Canopic: Pre-digital privacy-enhancing encodings for computer vision,” in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, 2020.

[43] LG. “LG Transparent OLED Signage.” (2021), [Online]. Available: `https://www.lg-informationdisplay.com/oled-signage/brand`.

[44] Cecilia Duong. “Camfecting: how hackers attack by gaining access to your webcam.” (2021), [Online]. Available: `https://www.unsw.edu.au/newsroom/news/2021/10/camfecting--how-hackers-attack-by-gaining-access-to-your-webcam`.

[45] Digital Trends. “Clearview AI’s facial-recognition app is a nightmare for stalking victims.” (2020), [Online]. Available: `https://bit.ly/3SZ5FZq`.

[46] daily dot. “Hawaii agency that sent false missile alert exposed a password on Post-it note.” (2021), [Online]. Available: `https://bit.ly/47yimi6`.

[47] ARSTCHNICA. “Hacked French network exposed its own passwords during TV interview.” (2015), [Online]. Available: `https://bit.ly/46D98Qq`.

[48] Digital Security Guide. “Machine learning: How does it help IT admins (and hackers too)?” (2023), [Online]. Available: `https://bit.ly/3SQGbgo`.

[49] Medium. “Machine Learning for Cybercriminals 101.” (2018), [Online]. Available: `https://bit.ly/49sOynC`.

[50] Google. “Google Vision API.” (2023), [Online]. Available: `https://cloud.google.com/vision`.

[51] OpenAI. “ChatGPT Vision API – GPT-4 Turbo with vision.” (2023), [Online]. Available: `https://platform.openai.com/docs/guides/vision`.

[52] “Laser light pioneer now moving in on Li-Fi,” *https://www.ledsmagazine.com/specialty-ssl/article/14074310/laser-light-pioneer-now-moving-in-on-lifi*,

[53] A. Gomez, K. Shi, C. Quintana, M. Sato, G. Faulkner, and et. al., “Beyond 100-Gb/s Indoor Wide Field-of-View Optical Wireless Communications,” *IEEE Photonics Technology Letters*, 2015.

[54] OPPO Guangdong Mobile Telecommunications CO., LTD, “Electronic equipment and LiFi communication system,” 2020.

[55] pureLiFi. “First-ever large-scale deployment of LiFi.” (2021), [Online]. Available: `https://purelifi.com/first-ever-large-scale-deployment-of-lifi`.

[56] ams. “Tcs3701.” (2019), [Online]. Available: `https://ams.com/documents/20143/36005/Proximity%20Sensors_AN000607_1-00.pdf/`.

[57] MAXVAL. “Featured technologies: Under display sensors.” (2020), [Online]. Available: `https://www.maxval.com/blog/featured-technologies-under-display-sensors/`.

[58] Q. Wang, D. Giustiniano, and D. Puccinelli, “OpenVLC: Software-Defined Visible Light Embedded Networks,” in *Proceedings of the ACM MobiCom Workshop on Visible Light Communication Systems (VLCS)*, 2014.

[59] Z. Tian, K. Wright, and X. Zhou, “The darklight rises: Visible light communication in the dark,” in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2016.

[60] G. Ntogari, T. Kamalakis, J. W. Walewski, and T. Sphicopoulos, “Combining illumination dimming based on pulse-width modulation with visible-light communications based on discrete multitone,” *Journal of Optical Communications and Networking*, 2011.

[61] Y. Yang, J. Luo, C. Chen, Z. Chen, W.-D. Zhong, and L. Chen, “Pushing the data rate of practical VLC via combinatorial light emission,” *IEEE Transactions on Mobile Computing*, 2021.

[62] Y. Wu, P. Wang, K. Xu, L. Feng, and C. Xu, “Turboboosting visible light backscatter communication,” in *Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM)*, 2020.

[63] P. Wang, L. Feng, G. Chen, *et al.*, “Renovating road signs for infrastructure-to-vehicle networking: A visible light backscatter communication and networking approach,” in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2020.

[64] R. Bloom, M. Zuniga, and C. Pai, “Luxlink: Creating a wireless link from ambient light,” in *Proceedings of the ACM Embedded Networked Sensor Systems (SenSys)*, 2019.

[65] R. Bloom, M. Zuniga, Q. Wang, and D. Giustiniano, “Tweeting with sunlight: Encoding data on mobile objects,” in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2019.

[66] C. CIE, “Commission internationale de l’eclairage proceedings, 1931,” *Cambridge University, Cambridge*, 1932.

[67] ams. “Tcs34725.” (2020), [Online]. Available: `https://ams.com/documents/20143/36005/TCS3472_DS000390_3-00.pdf/`.

[68] E. Monteiro and S. Hranilovic, “Design and implementation of color-shift keying for visible light communications,” *Journal of Lightwave Technology*, 2014.

[69] ams. “As73211.” (2018), [Online]. Available: `https://ams.com/as73211`.

[70] ams. “Lux and cct calculations using ams color sensors.” (2013), [Online]. Available: `https://www.mouser.com/pdfDocs/ColorProxDetect_AN000520_1-00.pdf`.

[71] G. Sharma and R. Bala, *Digital color imaging handbook*. 2017.

[72] S. Rajagopal, R. D. Roberts, and S.-K. Lim, “Ieee 802.15.7 visible light communication: Modulation schemes and dimming support,” *IEEE Communications Magazine*, 2012.

[73] A. Galisteo, D. Juara, Q. Wang, and D. Giustiniano, "OpenVLC1.2: Achieving higher throughput in low-end visible light communication networks," in *Proceedings of the Annual Conference on Wireless On-demand Network Systems and Services (WONS)*, 2018.

[74] Kingbright. "WP154A4SUREQBFZGC." (2014), [Online]. Available: `https://www.kingbrightusa.com/images/catalog/SPEC/WP154A4SUREQBFZGC.pdf`.

[75] Sparkfun. "LCD-15079." (2019), [Online]. Available: `https://www.sparkfun.com/products/15079`.

[76] R. A. Martínez-Ciro, F. E. López-Giraldo, A. F. Betancur-Perez, and J. M. Luna-Rivera, "Design and implementation of a multi-colour visible light communication system based on a light-to-frequency receiver," *Photonics*, 2019.

[77] T. Li, C. An, X. Xiao, A. T. Campbell, and X. Zhou, "Real-time screen-camera communication behind any scene," in *Proceedings of the ACM Mobile Systems, Applications, and Services (MobiSys)*, 2015.

[78] K. Zhang, Y. Zhao, C. Wu, *et al.*, "Chromacode: A fully imperceptible screen-camera communication system," *IEEE Transactions on Mobile Computing*, 2021.

[79] X. Xu, Y. Shen, J. Yang, *et al.*, "PassiveVLC: Enabling practical visible light backscatter communication for battery-free iot applications," in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2017.

[80] S. K. Ghiasi, M. Zuniga, and K. Langendoen, "A principled design for passive light communication," in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2021.

[81] M. Rahman, K. Adedara, and A. Ashok, "Enabling multiple access in visible light communication using liquid crystal displays: A proof-of-concept study," *Electronics*, 2020.

[82] Y.-S. Kuo, P. Pannuto, K.-J. Hsiao, and P. Dutta, "Luxapose: Indoor positioning with mobile phones and visible light," in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2014.

[83] Z. Yang, Z. Wang, J. Zhang, C. Huang, and Q. Zhang, "Wearables Can Afford: Light-Weight Indoor Positioning with Visible Light," in *Proceedings of the ACM Mobile Systems, Applications, and Services (MobiSys)*, 2015.

[84] Z. Yang, W. Zeyu, J. Zhang, C. Huang, and Q. Zhang, "Polarization-based visible light positioning," *IEEE Transactions on Mobile Computing*, 2018.

[85] J. Hao, Y. Yang, and J. Luo, "Ceilingcast: Energy efficient and location-bound broadcast through LED-camera communication," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2016.

[86] T. Tsujimura, *OLED display fundamentals and applications*. John Wiley & Sons, 2017.

[87] Adrian Wong. "Xiaomi 3rd Generation Under-Display Camera Technology!" (2020), [Online]. Available: `https://www.engadget.com/xiaomis-under-display-camera-tech-is-coming-to-phones-next-year-130012911.html`.

[88] H. Ye and Q. Wang, “SpiderWeb: Enabling Through-Screen Visible Light Communication,” in *Proceedings of the ACM Embedded Networked Sensor Systems (SenSys)*, 2021.

[89] N. Otsu, “A threshold selection method from gray-level histograms,” *IEEE Transactions on Systems, Man, and Cybernetics*, 1979.

[90] P.-S. Liao, T.-S. Chen, P.-C. Chung, *et al.*, “A fast algorithm for multilevel thresholding,” *Journal of Information Science and Engineering*, 2001.

[91] E. R. Davies, *Computer vision: principles, algorithms, applications, learning*. Academic Press, 2017.

[92] K. Jack, *Video demystified: a handbook for the digital engineer*. Elsevier, 2011.

[93] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4.

[94] X. Zhang and L. Xiao, “Rainbowrow: Fast optical camera communication,” in *Proceedings of the IEEE International Conference on Network Protocols (ICNP)*, IEEE, 2020.

[95] Florian Schmitt. “Analysis: DC Dimming vs. PWM – Can you dim AMOLED displays without the flickering?” (June 2019), [Online]. Available: `https://www.notebookcheck.net/Analysis-DC-Dimming-vs-PWM-Can-you-dim-AMOLED-displays-without-the-flickering.423121.0.html`.

[96] R. Szeliski, *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[97] P. Wang, L. Feng, G. Chen, *et al.*, “Renovating road signs for infrastructure-to-vehicle networking: A visible light backscatter communication and networking approach,” in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2020.

[98] Android. “Slow rendering.” (2022), [Online]. Available: `https://developer.android.com/topic/performance/vitals/render`.

[99] N. Emerton, D. Ren, and T. Large, “28-1: Image capture through TFT arrays,” in *SID Symposium Digest of Technical Papers*, 2020.

[100] H. Du, J. Han, X. Jian, *et al.*, “Martian: Message broadcast via LED lights to heterogeneous smartphones,” *IEEE Journal on Selected Areas in Communications*, 2017.

[101] N. Rajagopal, P. Lazik, and A. Rowe, “Visual light landmarks for mobile devices,” in *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2014.

[102] R. D. Roberts, “Undersampled frequency shift ON-OFF keying (UFSOOK) for camera communications (CamCom),” in *Proceedings of the IEEE Wireless and Optical Communications Conference (WOCC)*, 2013.

[103] Y. Hokazono, A. Koizuka, G. Zhu, M. Suzuki, Y. Narusue, and H. Morikawa, “Iotorch: Reliable LED-to-camera communication against inter-frame gaps and frame drops,” *IEEE Transactions on Mobile Computing*, 2019.

[104] S. D. Perli, N. Ahmed, and D. Katabi, "Pixnet: Interference-free wireless links using lcd-camera pairs," in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2010.

[105] T. Hao, R. Zhou, and G. Xing, "Cobra: Color barcode streaming for smartphone systems," in *Proceedings of the ACM Mobile Systems, Applications, and Services (MobiSys)*, 2012.

[106] A. Wang, S. Ma, C. Hu, J. Huai, C. Peng, and G. Shen, "Enhancing reliability to boost the throughput over screen-camera links," in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2014.

[107] F. Hermans, L. McNamara, G. Sörös, C. Rohner, T. Voigt, and E. Ngai, "Focus: Robust visual codes for everyone," in *Proceedings of the ACM Mobile Systems, Applications, and Services (MobiSys)*, 2016.

[108] B. Zhang, K. Ren, G. Xing, X. Fu, and C. Wang, "SBVLC: Secure barcode-based visible light communication for smartphones," *IEEE Transactions on Mobile Computing*, 2015.

[109] A. Wang, C. Peng, O. Zhang, G. Shen, and B. Zeng, "Inframe: Multiflexing full-frame visible communication channel for humans and devices," in *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*, 2014.

[110] A. Wang, Z. Li, C. Peng, G. Shen, G. Fang, and B. Zeng, "Inframe++ achieve simultaneous screen-human viewing and hidden screen-camera communication," in *Proceedings of the ACM Mobile Systems, Applications, and Services (MobiSys)*, 2015.

[111] M. Izz, Z. Li, H. Liu, Y. Chen, and F. Li, "Uber-in-light: Unobtrusive visible light communication leveraging complementary color channel," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2016.

[112] H. Liu, B. Liu, C. Shi, and Y. Chen, "Secret key distribution leveraging color shift over visible light channel," in *Proceedings of the Conference on Communications and Network Security (CNS)*, 2017.

[113] H. Liu, H. Ye, J. Yang, and Q. Wang, "Through-screen visible light sensing empowered by embedded deep learning," in *Proceedings of the ACM SenSys Workshop on Challenges in Artificial Intelligence and Machine Learning for Internet of Things (AIChallengeIoT)*, 2021.

[114] R. Feng, C. Li, H. Chen, S. Li, J. Gu, and C. C. Loy, "Generating aligned pseudo-supervision from non-aligned data for image restoration in under-display camera," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2023.

[115] E. Wenger, S. Shan, H. Zheng, and B. Y. Zhao, "SoK: Anti-facial recognition technology," in *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2023.

[116] ZTE. "ZTE launches its new-generation under-display camera smartphone Axon 30." (2021), [Online]. Available: `http://bit.ly/3GiJMfW`.

[117] TheVerge. "Xiaomi announces Mix 4 with under-display camera." (2021), [Online]. Available: `https://bit.ly/3urrnLe`.

[118] H. Ye, J. Xiong, and Q. Wang, “When VLC meets under-screen camera,” in *Proceedings of the ACM Mobile Systems, Applications, and Services (MobiSys)*, 2023.

[119] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, “Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition,” in *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, 2016.

[120] A. Kurakin, I. J. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” in *Artificial intelligence safety and security*, 2018.

[121] S. Shan, E. Wenger, J. Zhang, H. Li, H. Zheng, and B. Y. Zhao, “Fawkes: Protecting privacy against unauthorized deep learning models,” in *Proceedings of the USENIX Security Symposium (USENIX Security)*, 2020.

[122] A. developers. “Canvas.” (2023), [Online]. Available: `https://developer.android.com/reference/android/graphics/Canvas`.

[123] A. developers. “Multi-window support.” (2023), [Online]. Available: `https://developer.android.com/guide/topics/large-screens/multi-window-support`.

[124] A. developers. “Notifications.” (2023), [Online]. Available: `https://developer.android.com/design/ui/mobile/guides/home-screen/notifications`.

[125] J. W. Goodman and P. Sutton, “Introduction to fourier optics,” *Quantum and Semiclassical Optics-Journal of the European Optical Society Part B*, 1996.

[126] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-CAM: Visual explanations from deep networks via gradient-based localization,” in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2017.

[127] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?” In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[128] N. Papernot, P. McDaniel, and I. Goodfellow, “Transferability in machine learning: From phenomena to black-box attacks using adversarial samples,” *arXiv*, 2016.

[129] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, “Practical black-box attacks against machine learning,” in *Proceedings of the ACM Asia Conference on Computer and Communications Cecurity (ASIACCS)*, 2017.

[130] O. Suciu, R. Marginean, Y. Kaya, H. Daume III, and T. Dumitras, “When does machine learning $FAIL$? generalized transferability for evasion and poisoning attacks,” in *Proceedings of the USENIX Security Symposium (USENIX Security)*, 2018.

[131] A. Demontis, M. Melis, M. Pintor, *et al.*, “Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks,” in *Proceedings of the USENIX Security Symposium (USENIX Security)*, 2019.

[132] A. S. Shamsabadi, R. Sanchez-Matilla, and A. Cavallaro, “ColorFool: Semantic adversarial colorization,” in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2020.

[133] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *IEEE European Symposium on Security and Privacy (EuroS&P)*, IEEE, 2016.

[134] A. Modas, S.-M. Moosavi-Dezfooli, and P. Frossard, "Sparsefool: A few pixels make a big difference," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2019.

[135] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: A simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2016.

[136] Z. Zhao, Z. Liu, and M. Larson, "Towards large yet imperceptible adversarial image perturbations with perceptual color distance," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2020.

[137] G. A. Koulieris, K. Akşit, M. Stengel, R. K. Mantiuk, K. Mania, and C. Richardt, "Near-eye display and tracking technologies for virtual and augmented reality," in *Computer Graphics Forum*, Wiley Online Library, 2019.

[138] X. Wei, Y. Guo, and J. Yu, "Adversarial sticker: A stealthy attack method in the physical world," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[139] B. Wang, Y. Yao, B. Viswanath, H. Zheng, and B. Y. Zhao, "With great training comes great vulnerability: Practical attacks against transfer learning," in *Proceedings of the USENIX Security Symposium (USENIX Security)*, 2018.

[140] S.-H. Chae, C.-H. Yoo, J.-Y. Sun, M.-C. Kang, and S.-J. Ko, "Subpixel rendering for the pentile display based on the human visual system," *IEEE Trans. Consum. Electron.*, 2017.

[141] K. Gu, M. Liu, G. Zhai, X. Yang, and W. Zhang, "Quality assessment considering viewing distance and image resolution," *IEEE Trans. Broadcast.*, 2015.

[142] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2016.

[143] A. Howard, M. Sandler, G. Chu, *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[144] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[145] O. Russakovsky, J. Deng, H. Su, *et al.*, "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, 2015.

[146] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," in *Proceedings of the AAAI conference on artificial intelligence*, 2017.

[147] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018.

[148] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "VGGFace2: A Dataset for Recognising Faces across Pose and Age," in *Proceedings of the IEEE Automatic Face & Gesture Recognition (FG)*, 2018.

[149] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," *arXiv*, 2014.

[150] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, *et al.*, "Matching networks for one shot learning," in *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[151] H.-W. Ng and S. Winkler, "A data-driven approach to cleaning large face datasets," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2014.

[152] X. Zhang, S. Park, T. Beeler, D. Bradley, S. Tang, and O. Hilliges, "ETH-XGaze: A large scale dataset for gaze estimation under extreme head pose and gaze variation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[153] Y. Li, X. Yang, B. Wu, and S. Lyu, "Hiding faces in plain sight: Disrupting ai face synthesis with adversarial perturbations," in *Proceedings of International Conference on Learning Representations (ICLR)*, 2019.

[154] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, 2004.

[155] X. Lv and Z. J. Wang, "Reduced-reference image quality assessment based on perceptual image hashing," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2009.

[156] S. Pudlewski and T. Melodia, "Compressive video streaming: Design and rate-energy-distortion analysis," *IEEE Trans Multimedia*, 2013.

[157] G. D. Ruxton, "The unequal variance t-test is an underused alternative to student's t-test and the mann–whitney u test," *Behavioral Ecology*, 2006.

[158] G. Charness, U. Gneezy, and M. A. Kuhn, "Experimental methods: Between-subject and within-subject design," *Journal of Economic Behavior and Organization*, 2012.

[159] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proceedings of International Conference on Learning Representations (ICLR)*, 2015.

[160] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2016.

[161] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proceedings of International Conference on Learning Representations (ICLR)*, 2018.

[162] C. Guo, M. Rana, M. Cisse, and L. Van Der Maaten, "Countering adversarial images using input transformations," in *Proceedings of International Conference on Learning Representations (ICLR)*, 2018.

[163] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana, "Certified robustness to adversarial examples with differential privacy," in *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2019.

[164] J. Cohen, E. Rosenfeld, and Z. Kolter, "Certified adversarial robustness via randomized smoothing," in *Proceedings of International Conference on Machine Learning (ICML)*, 2019.

[165] A. Raghunathan, J. Steinhardt, and P. Liang, "Certified defenses against adversarial examples," in *Proceedings of International Conference on Learning Representations (ICLR)*, 2018.

[166] T. Pang, X. Yang, Y. Dong, H. Su, and J. Zhu, "Bag of tricks for adversarial training," in *Proceedings of International Conference on Learning Representations (ICLR)*, 2021.

[167] R. Feng, C. Li, H. Chen, S. Li, C. C. Loy, and J. Gu, "Removing diffraction image artifacts in under-display camera via dynamic skip connection network," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2021.

[168] K. Gao, M. Chang, K. Jiang, *et al.*, "Image restoration for real-world under-display imaging," *Optics Express*, 2021.

[169] N. Antipa, G. Kuo, R. Heckel, *et al.*, "Diffusercam: Lensless single-exposure 3d imaging," *Optica*, 2018.

[170] M. S. Asif, A. Ayremlou, A. Sankaranarayanan, A. Veeraraghavan, and R. G. Baraniuk, "Flatcam: Thin, lensless cameras using coded aperture and computation," *IEEE Transactions on Computational Imaging*, 2016.

[171] C. Szegedy, W. Zaremba, I. Sutskever, *et al.*, "Intriguing properties of neural networks," in *Proceedings of International Conference on Learning Representations (ICLR)*, 2014.

[172] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2017.

[173] K. Eykholt, I. Evtimov, E. Fernandes, *et al.*, "Robust physical-world attacks on deep learning visual classification," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018.

[174] S. Komkov and A. Petiushko, "AdvHat: Real-world adversarial attack on arcface face ID system," in *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR)*, 2021.

[175] G. Lovisotto, H. Turner, I. Sluganovic, M. Strohmeier, and I. Martinovic, "SLAP: Improving physical adversarial examples with $Short-Lived$ adversarial perturbations," in *Proceedings of the USENIX Security Symposium (USENIX Security)*, 2021.

[176] S. Zhu, C. Zhang, and X. Zhang, "Automating visual privacy protection using a smart LED," in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2017.

[177] A. Sayles, A. Hooda, M. Gupta, R. Chatterjee, and E. Fernandes, “Invisible perturbations: Physical adversarial examples exploiting the rolling shutter effect,” in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2021.

[178] A. developers. “Update other apps’ media files.” (2023), [Online]. Available: `https://developer.android.com/training/data-storage/shared/media`.

[179] Techradar. “Stop using sticky notes to write down passwords.” (2021), [Online]. Available: `https://bit.ly/3uz4HZA`.

[180] H. Ye, G. Lan, J. Jia, and Q. Wang, “Screen perturbation: Adversarial attack and defense on under-screen camera,” in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2023.

[181] D. Song, K. Eykholt, I. Evtimov, *et al.*, “Physical adversarial examples for object detectors,” in *Proceedings of the USENIX Workshop on Offensive Technologies (WOOT)*, 2018.

[182] Q. Song, Z. Yan, and R. Tan, “Moving target defense for embedded deep visual sensing against adversarial examples,” in *Proceedings of the ACM Embedded Networked Sensor Systems (SenSys)*, 2019.

[183] Z. Wu, S.-N. Lim, L. S. Davis, and T. Goldstein, “Making an invisibility cloak: Real world adversarial attacks on object detectors,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[184] C. Ren, X. Du, Y. Xu, Q. Song, Y. Liu, and R. Tan, “Vulnerability analysis, robustness verification, and mitigation strategy for machine learning-based power system stability assessment model under adversarial examples,” *IEEE Transactions on Smart Grid*, 2021.

[185] C. Yan, Z. Xu, Z. Yin, *et al.*, “Rolling colors: Adversarial laser exploits against traffic light recognition,” in *Proceedings of the USENIX Security Symposium (USENIX Security)*, 2022.

[186] S. Köhler, G. Lovisotto, S. Birnbach, R. Baker, and I. Martinovic, “They see me rollin’: Inherent vulnerability of the rolling shutter in CMOS image sensors,” in *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, 2021.

[187] “CVE-2019-2234,” *https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-2234*, 2018.

[188] “rdar://35116272,” *https://openradar.appspot.com/35116272*, 2017.

[189] Europol. “World’s most dangerous malware EMOTET disrupted through global action.” (2022), [Online]. Available: `https://bit.ly/4bo7TIB`.

[190] Chuck Gallagher. “Your Camera Is Watching You, Are You Protected?” (2021), [Online]. Available: `https://bit.ly/3HF6XSf`.

[191] M. Dong, Y.-S. K. Choi, and L. Zhong, “Power modeling of graphical user interfaces on OLED displays,” in *Proceedings of the Annual Design Automation Conference (DAC)*, 2009.

[192] J. W. Goodman, *Introduction to Fourier optics*. 2005.

[193] C. C. Stearns and K. Kannappan, *Method for 2-D affine transformation of images*, US Patent 5,475,803, 1995.

[194] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2016.

[195] F. Dufaux, P. Le Callet, R. Mantiuk, and M. Mrak, *High dynamic range video: from acquisition, to display and applications*. Academic Press, 2016.

[196] F. Isensee, P. F. Jaeger, S. A. Kohl, J. Petersen, and K. H. Maier-Hein, "Nnu-net: A self-configuring method for deep learning-based biomedical image segmentation," *Nature methods*, 2021.

[197] T. Brooks, B. Mildenhall, T. Xue, J. Chen, D. Sharlet, and J. T. Barron, "Unprocessing images for learned raw denoising," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2019.

[198] X. Shen, F. Darmon, A. A. Efros, and M. Aubry, "Ransac-flow: Generic two-stage image alignment," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[199] R. Feng, C. Li, H. Chen, S. Li, J. Gu, and C. C. Loy, "Generating aligned pseudo-supervision from non-aligned data for image restoration in under-display camera," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2023.

[200] A. Abdelhamed, S. Lin, and M. S. Brown, "A high-quality denoising dataset for smartphone cameras," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018.

[201] H. Ouyang, Z. Shi, C. Lei, K. L. Law, and Q. Chen, "Neural camera simulators," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2021.

[202] J. Li, F. Schmidt, and Z. Kolter, "Adversarial camera stickers: A physical camera-based attack on deep learning systems," in *Proceedings of International Conference on Machine Learning (ICML)*, 2019.

[203] A. Zolfi, M. Kravchik, Y. Elovici, and A. Shabtai, "The translucent patch: A physical and universal attack on object detectors," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2021.

[204] A. Foi, M. Trimeche, V. Katkovnik, and K. Egiazarian, "Practical poissonian-gaussian noise modeling and fitting for single-image raw-data," *IEEE Transactions on Image Processing*, 2008.

[205] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, 2017.

[206] OpenAI. "GPT-4V(ision)SystemCard." (2023), [Online]. Available: `https://bit.ly/481qs2G`.

[207] R. Mittal, A. Kansal, and R. Chandra, "Empowering developers to estimate app energy consumption," in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2012.

[208] X. Chen, Y. Chen, Z. Ma, and F. C. Fernandes, “How is energy consumed in smartphone display applications?” In *Proceedings of the International Workshop on Mobile Computing Systems and Applications (HotMobile)*, 2013.

[209] TechSpot. “Tested: The Galaxy S7’s always-on display consumes very little battery.” (2016), [Online]. Available: `https://bit.ly/3sQPZga`.

[210] N. Das, M. Shanbhogue, S.-T. Chen, *et al.*, “Shield: Fast, practical defense and vaccination for deep learning using jpeg compression,” in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2018.

[211] Y. Cao, N. Wang, C. Xiao, *et al.*, “Invisible for both camera and LiDAR: Security of multi-sensor fusion based perception in autonomous driving under physical-world attacks,” in *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2021.

[212] T. Sugawara, B. Cyr, S. Rampazzi, D. Genkin, and K. Fu, “Light commands:Laser-based audio injection attacks on voice-controllable systems,” in *Proceedings of the USENIX Security Symposium (USENIX Security)*, 2020.

[213] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, “Synthesizing robust adversarial examples,” in *Proceedings of International Conference on Machine Learning (ICML)*, 2018.

[214] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer, “Adversarial patch,” *arXiv*, 2017.

[215] Y. Zhao, H. Zhu, R. Liang, Q. Shen, S. Zhang, and K. Chen, “Seeing isn’t believing: Towards more robust adversarial attack against real world object detectors,” in *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, 2019.

[216] M. Pautov, G. Melnikov, E. Kaziakhmedov, K. Kireev, and A. Petiushko, “On adversarial patches: Real-world attack on arcface-100 face recognition system,” in *Proceedings of the IEEE International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON)*, 2019.

[217] Z. Zhang, “Image deblurring of camera under display by deep learning,” in *SID Symposium Digest of Technical Papers*, 2020.

[218] B. Liu, Q. Wang, and S. Pollin, “Tais: Transparent amplifying intelligent surface for indoor-to-outdoor mmwave communications,” *IEEE Transactions on Communications*, 2023.

[219] Meta Materials. “Transparent Antennas.” (2024), [Online]. Available: `https://metamaterial.com/solutions/transparent-antennas-2/`.

[220] Meta Materials. “A Sustainable Solution for Improved Network Coverage.” (2024), [Online]. Available: `https://metamaterial.com/industries/5g-communications/`.

[221] K. S. Adu-Manu, N. Adam, C. Tapparello, H. Ayatollahi, and W. Heinzelman, “Energy-harvesting wireless sensor networks (eh-wsns) a review,” *ACM Transactions on Sensor Networks*, 2018.

[222] M. Gulati, F. S. Parizi, E. Whitmire, *et al.*, “Capharvester: A stick-on capacitive energy harvester using stray electric field from ac power lines,” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, 2018.

[223] L. Liu, K. Cao, S. Chen, and W. Huang, “Toward see-through optoelectronics: Transparent light-emitting diodes and solar cells,” *Advanced Optical Materials*, 2020.

[224] T. T. Nguyen, M. Patel, and J. Kim, “All-inorganic metal oxide transparent solar cells,” *Solar Energy Materials and Solar Cells*, 2020.

[225] L. A. Wehner, N. Mittal, T. Liu, and M. Niederberger, “Multifunctional batteries: Flexible, transient, and transparent,” *ACS Central Science*, 2021.

[226] ArchDaily. “Smart Facades: Buildings that Adapt to the Climate Through their Skin.” (2019), [Online]. Available: `https://www.archdaily.com/922537/smart-facades-buildings-that-adapt-to-the-climate-through-their-skin?ad_medium=gallery`.

[227] engadget. “Researchers develop flexible, see-through battery.” (2007), [Online]. Available: `https://www.engadget.com/2007-03-22-researchers-develop-flexible-see-through-battery.html`.

[228] The American Ceramic Society. “Wearable displays go ‘thin as skin’ with novel transparent oxide thin-film transistors.” (2016), [Online]. Available: `https://ceramics.org/ceramic-tech-today/wearable-displays-go-thin-as-skin-with-novel-transparent-oxide-thin-film-transistors/`.

[229] S. Lim, L. Liang, Y. Zhong, N. Emerton, T. Large, and S. Bathiche, “18-3: Free viewpoint teleconferencing using cameras behind screen,” in *SID Symposium Digest of Technical Papers*, 2021.

[230] T. Li, C. An, Z. Tian, A. T. Campbell, and X. Zhou, “Human sensing using visible light communication,” in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2015.

[231] T. Li, Q. Liu, and X. Zhou, “Practical human sensing in the light,” in *Proceedings of the ACM Mobile Systems, Applications, and Services (MobiSys)*, 2016.

[232] D. Ma, G. Lan, M. Hassan, *et al.*, “Solargest: Ubiquitous and battery-free gesture recognition using solar cells,” in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2019.

[233] Z. Liao, Z. Luo, Q. Huang, *et al.*, “Smart: Screen-based gesture recognition on commodity mobile devices,” in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2021.

[234] T3. “Should all TVs come with cameras built-in? I say, "Yes!".” (2023), [Online]. Available: `https://www.t3.com/news/should-all-tvs-come-with-cameras-built-in-i-say-yes`.

[235] Samsung. “Samsung’s Family Hub Brings Food AI and Automation into the Kitchen at CES 2020.” (2022), [Online]. Available: `https://news.samsung.com/global/samsungsfamily-hub-brings-food-ai-and-automationinto-the-kitchen-at-ces-2020`.

[236] Furbo. “Furbo Dog Camera.” (2022), [Online]. Available: `https://shopcn.furbo.com/products/furbo`.

[237] J. Bugeja, D. Jönsson, and A. Jacobsson, “An investigation of vulnerabilities in smart connected cameras,” in *Proceedings of the IEEE PerCom workshops*, 2018.

[238] The Intercept. “Ring data can be view by staff.” (2022), [Online]. Available: `https://theintercept.com/2019/01/10/amazon-ringsecurity-camera/`.

[239] Security Boulevard. “Xiaomi IoT Cameras Leak Private Stills via Google Home Hub.” (2020), [Online]. Available: `https://securityboulevard.com/2020/01/xiaomi-iot-cameras-leak-privatestills-via-google-home-hub/`.

[240] Y. Hua, S. Nakamura, M. S. Asif, and A. C. Sankaranarayanan, “Sweepcam—depth-aware lensless imaging using programmable masks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[241] Y. Zheng, Y. Hua, A. C. Sankaranarayanan, and M. S. Asif, “A simple framework for 3d lensless imaging with programmable masks,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021.

[242] T. Van Ede, R. Bortolameotti, A. Continella, *et al.*, “Flowprint: Semi-supervised mobile-app fingerprinting on encrypted network traffic,” in *Proceedings of Network and distributed system security symposium (NDSS)*, 2020.

[243] J. Li, H. Zhou, S. Wu, *et al.*, “{Foap}:{fine-grained}{open-world} android app fingerprinting,” in *Proceedings of the USENIX Security Symposium (USENIX Security)*, 2022.

[244] Y. Liu, G. W. Wornell, W. T. Freeman, and F. Durand, “Imaging privacy threats from an ambient light sensor,” *Science Advances*, 2024.

[245] Z. Xie, X. Ouyang, X. Liu, and G. Xing, “Ultradepth: Exposing high-resolution texture from depth cameras,” in *Proceedings of the ACM Embedded Networked Sensor Systems (SenSys)*, 2021.

[246] Z. Xie, X. Ouyang, L. Pan, W. Lu, G. Xing, and X. Liu, “Mozart: A mobile tof system for sensing in the dark through phase manipulation,” in *Proceedings of the ACM Mobile Systems, Applications, and Services (MobiSys)*, 2023.

[247] Statista Research Department. “Number of spycam related crimes in South Korea from 2011 to 2021.” (2022), [Online]. Available: `https://www.statista.com/statistics/1133121/south-korea-number-of-spycam-crimes/`.

[248] Courtney Linder. “There Might Be Secret Surveillance Equipment in Your Vacation Rental.” (2020), [Online]. Available: `https://www.popularmechanics.com/technology/security/a32983077/vacation-rental-surveillance-spying/`.

[249] T. Liu, Z. Liu, J. Huang, R. Tan, and Z. Tan, “Detecting wireless spy cameras via stimulating and probing,” in *Proceedings of the ACM Mobile Systems, Applications, and Services (MobiSys)*, 2018.

[250] R. A. Sharma, E. Soltanaghaei, A. Rowe, and V. Sekar, “Lumos: Identifying and localizing diverse hidden {iot} devices in an unfamiliar environment,” in *Proceedings of the USENIX Security Symposium (USENIX Security)*, 2022.

[251] Z. Yu, Z. Li, Y. Chang, S. Fong, J. Liu, and N. Zhang, “Heatdecam: Detecting hidden spy cameras via thermal emissions,” in *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, 2022.

[252] S. Sami, S. R. X. Tan, B. Sun, and J. Han, “Lapd: Hidden spy camera detection using smartphone time-of-flight sensors,” in *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2021.

[253] D. Lee, S.-B. Kim, T. Kim, *et al.*, “Stretchable oleds based on a hidden active area for high fill factor and resolution compensation,” *Nature Communications*, 2024.

[254] M. Gopakumar, G.-Y. Lee, S. Choi, *et al.*, “Full-colour 3d holographic augmented-reality displays with metasurface waveguides,” *Nature*, 2024.

LIST OF PUBLICATIONS

The list of publications accomplished during PhD.

1. **Hanting Ye**, Guohao Lan, Jinyuan Jia, Qing Wang, *"Unicorn: Securing USC with Universal Screen Perturbation"*, **(Under Submission)**, 2025.

2. **Hanting Ye**, Niels van der Kolk, Qing Wang, *"NIRF: Detecting Cameras That Hide Behind Screen"*, Conditionally Accepted by the ACM Conference on Mobile Computing and Networking **(MobiCom'25)**, 2025.

3. **Hanting Ye**, Qing Wang, *"Computing behind Transparent Screen"*, In Proceedings of the International Conference on Embedded Wireless Systems and Networks **(EWSN'24)**, 2024.

4. **Hanting Ye**, Guohao Lan, Jinyuan Jia, Qing Wang, *"Screen Perturbation: Adversarial Attack and Defense on Under-Screen Camera"*, In Proceedings of the ACM Conference on Mobile Computing and Networking **(MobiCom'23)**, 2023.

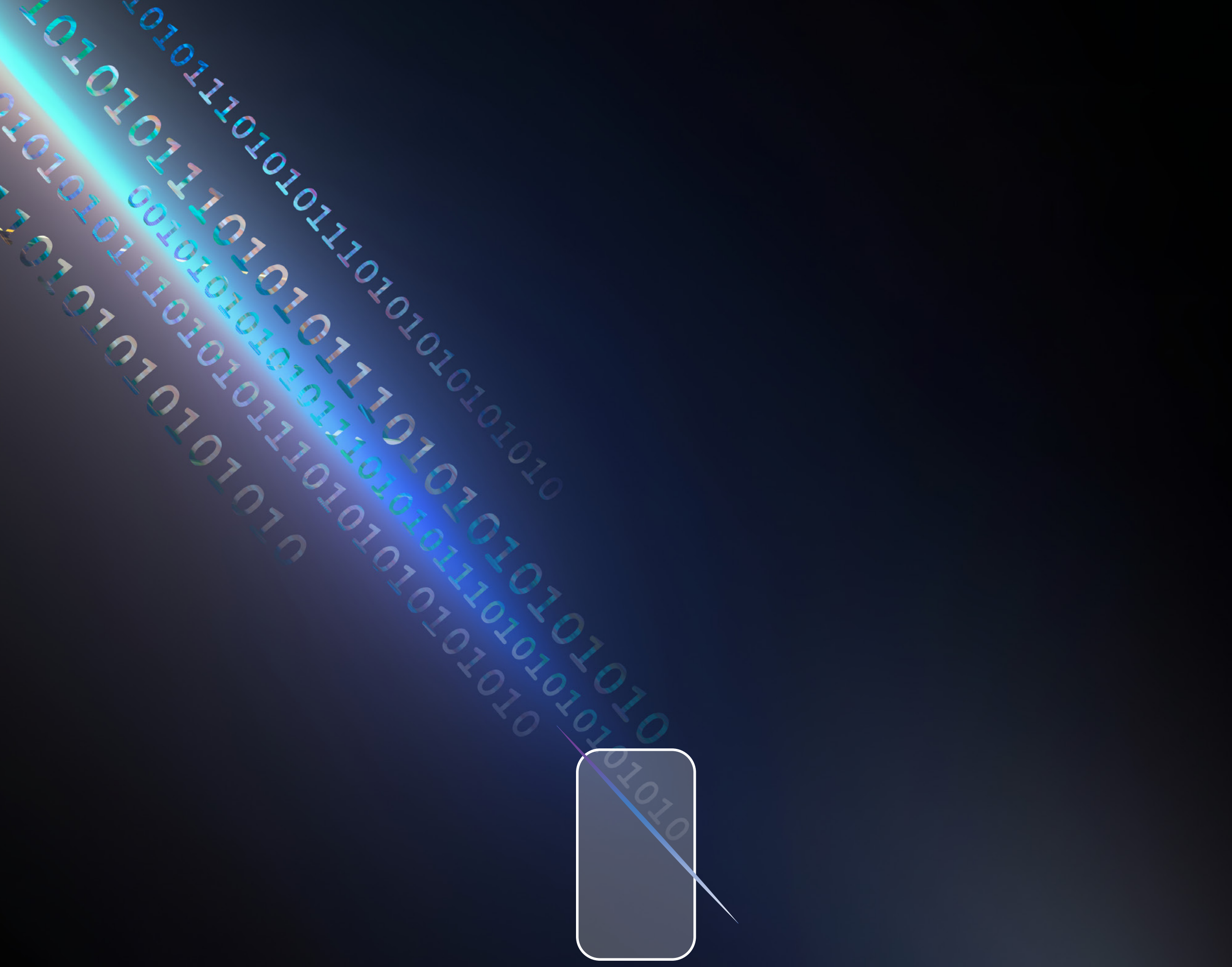
5. **Hanting Ye**, Jie Xiong, Qing Wang, *"When VLC Meets Under Screen Camera"*, In Proceedings of the ACM Conference on Mobile Systems, Applications, and Services **(MobiSys'23)**, 2023.

6. **Hanting Ye**, Qing Wang, *"SpiderWeb: Enabling Through-Screen Visible Light Communication"*, In Proceedings of the ACM Conference on Embedded Network Sensor Systems **(SenSys'21)**, 2021.

—

7. Hao Liu, **Hanting Ye**, Xiangrui Zhang, Jie Yang, Qing Wang, *"Fingertip Air-Writing with Ambient Light"*, In Proceedings of the EAI Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services **(MobiQuitous'23)**, 2023.

8. Hao Liu, **Hanting Ye**, Jie Yang, Qing Wang, *"Through-Screen Visible Light Sensing Empowered by Embedded Deep Learning"*, In Proceedings of ACM SenSys Workshop on Challenges in Artifiicial Intelligence and Machine Learning for Internet of Things **(AIChallengeIoT'21)**, 2021.


1010111010101010
1010111010101010
0010101011101010101010
01010111010101010
1010101010

2024