This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415

Screen-Shooting Resistant Watermarking with Grayscale Deviation Simulation

Yiyi Li, Xin Liao* and Xiaoshuai Wu

***Abstract*—With the prevalence of electronic devices in our daily lives, content leakages frequently occur, and to enable leakage tracing, screen-shooting resistant watermarking has attracted tremendous attention. However, current studies often overlook a thoughtful investigation of the cross-media screen-camera process and fail to consider the effect of grayscale deviation on the screen. In this paper, we propose screen-shooting distortion simulation (SSDS), which involves a grayscale deviation function for constructing a more practical noise layer. We divide SSDS into screen display and camera shoot. For screen displaying, different viewing angles result in grayscale deviation with distinct intensities, and we simulate the distortions by modeling the relative position of the viewing point and the screen plane. For camera shooting, a series of distortion functions are used to approximate the perturbations in the camera pipeline, including defocus blur, noise and JPEG compression. Furthermore, the gradient-guided encoder is designed to conduct the embedding in the texture region using a modification cost map. Experimental results show that our proposed watermarking framework outperforms the state-of-the-art methods in terms of robustness and visual quality.**

***Index Terms*—Robust watermarking, deep learning, screen-shooting, distortion simulation**

I. Introduction

NOWADAYS, it has become more convenient to capture content on screens with camera phones. Meanwhile, leakage cases are increasing since confidential documents can be easily captured and distributed with the widespread use of mobile devices. In these cases, watermarking technology can embed invisible information within images, which can serve as a way for leakage tracing. In the example of leakage tracing in Fig. 1, an employee can work as usual with the screen embedded with watermark information, which is essentially an identity mark of screen ID, screen user and timestamp. When a leaker captures the secrets from the screen, one can extract the identity mark from the leaked image, thus figuring out which screen was captured. With clues in the extracted watermark, inspectors can find the leaker by checking surveillance videos at the corresponding time, thus realize the leakage tracing. In this scenario, we use the term cross-media to denote the presentation of digital images transmitted in different physical media, such as screen displaying and camera shooting. And the watermark within the leaked image has to confront the cross-media process, which can cause complex distortion or even corruption to the hidden watermark. It comes to be a challenge for the existing digital watermarking methods since they only focus on digital editing for leakage tracing in electronic channels, lacking the prior knowledge for cross-media distortion. Current digital watermarking methods can hardly provide faithful extraction in such cases, while the requirement for leakage tracing under cross-media distortion is increasing. Since it has become common to capture content on screens through mobile phone cameras with the prevalence of camera phones and screen displays, which indicates the demand for resistance to screen-shooting. Therefore, it is urgent to develop a well-designed watermarking algorithm that can resist distortions in screen shooting.

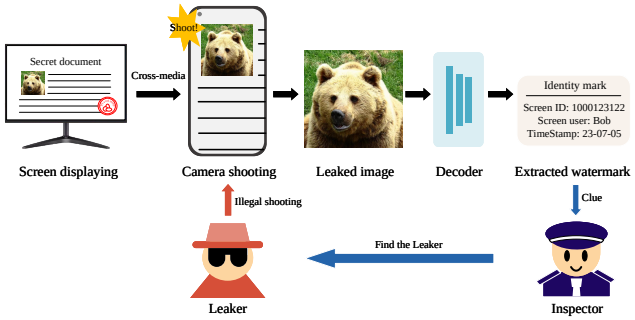


Fig. 1. Illustration of the leakage tracing with our screen-shooting resistant watermarking.

Watermarking algorithms have been extensively studied with various usages, and we will focus on the discussion of robust watermarking. The robustness of traditional watermarking can be realized by manually designed embedding with the prior knowledge of attacks. In the field of deep learning-based watermarking, most existing studies follow a basic encoder-noise layer-decoder structure [1]. The encoder and decoder are responsible for embedding and extracting information, respectively, while the noise layer is aim at introducing noise features into the network training so as to gain the resistance against distortions. HiDDeN [3] is a representative model of this kind of structure, which is equipped with a noise layer composed of dropout, cropping, compression, and so on. More researches are conducted with novel design of noise layer. Some of them follows the encoder-noise layer-decoder structure [3], [20], [22] while the others propose new structures, including two-stage TSDL [21] and De-END architecture [23]. However, given that the noise layer in these methods focus on digital editing distortions, this watermarking

This work is supported by National Natural Science Foundation of China (Grant Nos. U22A2030, 61972142), National Key R&D Program of China (Grant No. 2022YFB3103500), Hunan Provincial Funds for Distinguished Young Scholars (Grant No. 2024JJ2025).

Yiyi Li, Xin Liao and Xiaoshuai Wu are with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China (e-mail: yiyili@hnu.edu.cn, xinliao@hnu.edu.cn, shinewu@hnu.edu.cn).

*Corresponding author: Xin Liao

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415

network is only able to resist against digital operations, lacking a cross-media robustness [4]. Inspired by the success of noise layer in deep learning-based digital watermarking, Tancik *et al.* [4] simulated the cross-media distortion in the same way and proposed StegaStamp, which combined the idea of image perturbation and enriched the components of the noise layer. By compounding a variety of perturbations, the watermark was able to resist physical distortions from printer-scanner, printer-camera, and screen-camera. The idea of distortion simulation has influenced the following researches [7], [9], [26], [27]. Jia *et al.* [7] took into account the influence of lighting by adding its simulation process in training. Thus the watermark was capable of maintaining high decoding accuracy under exposure. Apart from this, some researchers [9] simplified the noise layer by focusing on a few factors that have greater impacts. Among these methods, researchers design noise layer and introduce distortion to the network training [4], [7], [9]. With the learned knowledge for distortion, watermarks can gain robustness to survive in similar distortions.

When a watermark is presented in the real world as a physical entity, such as confidential information on a monitor or printed copyrighted artworks, these digital images undergo a process of physicalization which inevitably introduces information loss, causing the failure of watermark extraction. To address this problem, a proven solution is to enhance robustness by approximating distortion with a well-designed noise layer and explicitly putting it into the network framework during training [4], [7], [9]. By introducing the targeted noise layer, the distortion features are passed through and influence the back propagation, thus the corresponding robustness can be guaranteed.

As analysed above, to maintain robustness in screen-shooting scenarios, the watermarking networks should construct a noise layer to approximate the image perturbation in the real world properly. It requires further insight into cross-media and the ability to model these complex and indescribable processes. We attempted to decompose the screen-shooting process, which consists of the screen displaying and camera shooting. We found that existing works only focus on the distortion of camera shooting and simulate it with blur, noise, light, and so on. They lack further study for screen displaying, especially the grayscale deviation effect caused by different perspectives, which can cause difficulty for watermark extraction. To address above limitations, we study how the screen displays and propose a corresponding distortion simulation method. Our contributions can be summarized into three-fold:

1) We are the first to simulate grayscale deviation with GDF, which represents the imaging principle in distinct viewing angles. GDF along with more distortion simulations construct **SSDS**, introducing the distortion into the networks to enable the resistance to the negative effect in cross-media.

2) We propose the gradient guidance which can guide the encoder to fully utilize the visually imperceptible regions of the image to embed information, spreading the visible shadows to the contour edges.

3) Our method has been demonstrated to be more effective compared with some state-of-the-art methods in extensive experiments of various shooting distances and angels.

The remainder of the paper is organized as follows. In Section II, we discuss the related works. Section III formulates the grayscale deviation issue with analysis and presents the details about the proposed screen-shooting distortion simulation. In Section IV, the structure of the network architecture and corresponding loss function is introduced. In Section V, the evaluation and analysis of the experimental results are provided. Finally, a conclusion is given for this work in Section VI.

II. Related Work

Robust watermarking can be implemented using various kinds of techniques. In this section, we first describe some traditional watermarking methods with manual designing. Then, we describe some digital watermarking algorithms based on deep learning. Finally, some watermarking methods with strong robustness against screen-shooting are introduced.

A. Traditional Watermarking

Compared to deep learning-based watermarking, traditional digital watermarking does not rely on neural networks, but is manually designed based on the prior knowledge of the attack characteristics [10]. For example, for the printing scenario, Kim *et al.* [11] used a rectilinear tiling pattern as a template for robust printing image restoration empirically. They extracted watermarks by detecting associated peaks. In the aspect of human visual system, some empirical or scenario-specific designs also exist. Based on the priori knowledge that human eye is sensitive to font content instead of glyphs, Xiao *et al.* [12] embedded information by continuously changing the glyphs of each character on the font manifold. Similarly, Li *et al.* [13] used the alpha channel to encode the data into the semi-transparent changes of pixels, adjusting the pixel values with the visual impact reducing. As for chart applications, Fu *et al.* [14] designed synchronous markers to embed watermarks into the background of chart images, which does not affect the contents while improving robustness. The performance of traditional watermarks depends on the design of embedding algorithm. For example, Hui *et al.* [15] improved invisibility by embedding watermark based on the spatial-temporal histogram. Fang *et al.* proposed a novel screen-shooting resilient watermarking [8], which we called SSR. They applied an intensity-based scale-invariant feature transform algorithm to find suitable regions for embedding the well-designed template that represents watermark information. Chen *et al.* [16] proposed a feature extraction method for local regions, with Harris-Laplace and SURF corner point detection to achieve feature synchronization. In the frequency domain, Amiri *et al.* [17] analyzed images in the DWT domain and used a genetic algorithm to find suitable locations for watermark embedding. Huang et al. [18] found that spread spectrum and quantization suffer from host signal interference, and they proposed a spread spectrum scheme with adaptive embedding strength. Solanki *et al.* [19] quantized the difference of adjacent frequency locations to embed the information in the phase spectrum of the image. To sum up,

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415

traditional watermarking methods can embed information by modifying pixel values in the spatial domain or coefficients in the frequency domain. However, they only use handcrafted features for embedding and extracting, which is relevant to the targeted task but lacks generalizable.

B. Deep Learning-Based Digital Watermarking

Recently, many researchers have utilize neural networks in watermarking studies. With a suitable training method, the watermarking model can adaptively extract the image features and embed the given information into the image [3], [21]–[24]. HiDDeN [3] is a typical representative of deep learning-based watermarking methods. It consists of three sub-networks that work cooperatively as a whole. The encoder and decoder are responsible for embedding and extracting respectively. The discriminator criticizes the generated images, and the critical result is fed to the encoder, guiding the generation. One of the most significant goals of watermarking is robustness. HiDDeN [3] proposed an encoder-noise layer-decoder structure, and its robustness depends on the design of the noise layer. The noise layer simulates digital editing operations such as cropping and JPEG. By introducing the noise layer in network training, the network learns how to cope with distortions. Chen *et al.* [20] proposed JSNet, a more realistic network for simulating JPEG compression. They provided a differentiable JPEG compression module for watermarking. For non-differentiable features, such as JPEG compression, researchers [20] simulated them in differentiable forms so that they can be involved in the end-to-end training. Ahmadi *et al.* [22], on the other hand, designed a mixture of attacks for robustness. Specifically, they utilized a multi-attack layer to apply different attacks in each iteration and randomly select one of the attacks with a given probability. The robustness of deep learning-based watermarking methods relies heavily on the design of noisy layer attacks, and how to make the watermarking network better learn the features of attacks is the main idea of robustness optimization. These methods [3], [20], [22] are based on attack simulation, which deliver the differentiable features of the attack by applying attacks to the watermarked image and passing it directly to the watermark extraction network. Another idea is to disconnect the training directly, such as the two-stage method TSDL [21], which disconnects the gradient backpropagation of the sub-network. TSDL uses the attacks directly, but the lack of gradient back-propagation may lead to inadequate learning of attack features. Fang *et al.* [23] state that encoders may embed redundant features that are not necessary for decoding, thus they proposed a decoder-encoder-noise layer-decoder architecture. Luo *et al.* [24] used channel coding to improve the robustness, but at the same time, reduced the actual embedding capacity. The methods mentioned above focus on the robustness of digital editing distortion, such as cropping and rotation. However, they can hardly withstand the complex distort in cross-media processes, since they lack corresponding designing for noise layer.

C. Screen-Shooting Resistant Watermarking

Widely used electronic devices enable people to conveniently take photos of screens, which leads to social issues in confidential leakage, copyright infringement, and so on. Screen-shooting watermarking is dedicated to solving such problems, and it has a stronger cross-media robustness. To optimize the noise layer, one of the ideas is more realistic simulations, such as the StegaStamp proposed by Tancik *et al.* [4]. This method analyzed the complex physicalization process and decomposed it into a series of image operations. Jia *et al.* [7] followed this idea and added a lighting simulation module that can better approximate the realistic image display. While simulation-based watermarking methods tended to compound multiple image operations, PiMoG [9] simplified the noise layer, retaining only the most significant components, the illumination layer, moiré layer, and Gaussian noise, which were effective and more conducive to model convergence. No simulation is more realistic than real data, and the idea of Wengrowski *et al.* [25] is to train the distortion network with a large amount of real data. They used a 1.9 TB dataset to train so that the network learned the features of screen distortion. Similar to the simulation methods, this method enhanced the robustness by optimizing the noise layer. However, it was based on a huge amount of data, making it difficult to be implemented at scale. Qin et al. [26] designed a deep noise simulation network to simulate the fusion process of real-world noises, which was beneficial in generating robust watermarks. In addition, unsupervised training approaches have also been proposed [27]. Zhong *et al.* [27] did not require any priori knowledge which avoided human intervention and annotation. Overall, most screen-shooting watermarking methods improve the robustness by optimizing the noise layer, including distortions simulation and directly introducing realistic distortion data. They provide approximating methods for camera shooting while limit in the study to the phenomenon of grayscale deviation on screens. It decreases the performance of existing watermarking, and it becomes worse as the viewing angle increases.

III. Screen-shooting Distortion Simulation

In a screen-shooting watermarking task, watermark information is transmitted between the digital and physical worlds, which indicates an essential request for cross-media robustness. To improve this robustness, we designed **SSDS**, which is involved in the network training, introducing the distortion characteristic in the whole network. In this section, we first analyze the impact introduced by screen-shooting, and then we give detailed information about the proposed **SSDS** module, which consists of distortion simulation of screen displaying and camera shooting.

A. Analysis of Screen-Shooting and grayscale deviation

Before presenting the specific method, we provide a detailed analysis of the screen-shooting impact, especially the impact of grayscale deviation. If we tilt our bodies slightly and look sideways at a screen, we will find that the bright images on the screen become darker. With the view angle increasing, the screen sight becomes even darker. The view angle strongly influences the screen displaying, which affects the displayed grayscale and causes darkness in screen corners. In extreme cases, it can result in grayscale inversion. It is a common

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415

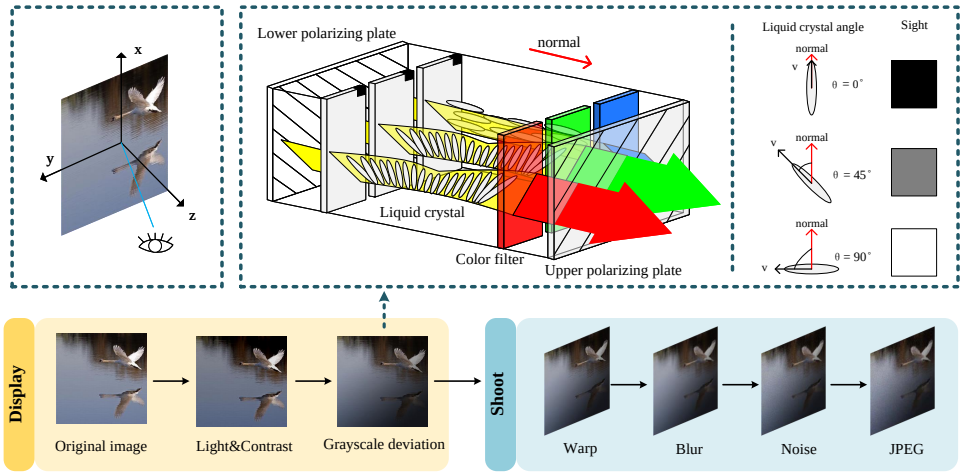


Fig. 2. Illustration of our screen-shooting distortion simulation. To improve robustness, we design simulation methods for screen displaying and camera shooting, thus enabling the watermarking model to learn distortion features during training. The distortion is simulated by several variations, including light and contrast, grayscale deviation, wrap, blur, noise, and JPEG compression.

phenomenon in screen-shooting, but most previous works ignore it and conduct screen-shooting studies in the same way as handling ordinary images. To solve this problem, we need to look inside the screen monitors and figure out the way how screens affect the grayscale.

We provide the structure of a liquid crystal display (LCD) screen in Fig. 2. The backlight provides light for display, and the light-guide plate, which covers with dots, propagates the light. When some of the light rays hit the dots, they emerge out of the front, lighting up the screen. Diffuser film eliminates the dot pattern, and prism film brightens the light. A polarizing plate only allows light waves of a specific polarization to pass through. In an LCD screen, two polarizing plates are settled 90 degrees to each other. Straight light comes out from the inner plate, but it can't pass through the other, since the plates have different directions. To build a light path, liquid crystals are filled between the polarizing plates, which line up in a helix, so the light rays twist and emerge out of the outer polarizing plate. If the liquid crystals are applied with an electric field, they will be parallel to the normal direction Fig. 2, thus forming a straight path instead of twisting. In this case, the light rays can't pass through, resulting in dark sight. The magnitude of crystal inclination can be controlled by the voltage, so that different intensities of light are displayed, which is the observed grayscale. We let θ denote the tilt angle of the liquid crystal. When the electric field is off, the crystals are in perfect horizontal helix, representing a grayscale of 255 with $\theta = 90°$. With the max electric field on, liquid crystals tilt up with $\theta = 0°$, representing a grayscale of 0. We can simply discuss grayscales on color screens, because the colors come from color films, such as RGB film, which turns the white light into red, green, and blue [38]. Every pixel point on screens contains lights of three colors, and their various intensities contribute to the colors, as shown in Fig. 2. The discussed grayscale is actually the intensity of light. We will formulate the effect of grayscale deviation in Section III-B.

In addition, the screen-shooting process can involve other distortions. Without loss of generality, the watermarked image

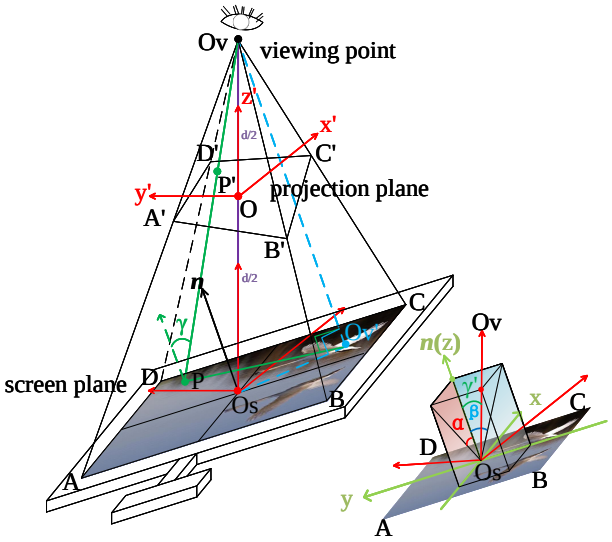


Fig. 3. Illustration of the 3D spatial screen projection. The shape of the projection plane and the pixel-wise viewing angle are calculated by the coordinates of the observation point and the screen plane. The pixel-wise viewing angle is the basic of grayscale deviation simulation.

is first saved in a digital format. When it is displayed on a screen, it is regarded as a physical pattern secondly. The pattern finally transforms back into a digital image after being caught by a camera. Cameras can also involve complex distortions, and we will discuss them in Section III-C. To enhance the resistance to cross-media distortion, we design a screen-shooting distortion simulation module **SSDS**, which is composed of screen displaying and camera shooting. Watermarked images outputted from the encoder will be applied distortion by **SSDS**. The decoder learns the distortion features from the distorted images, and the evaluation of its decoding accuracy will guide the encoder via backpropagation. The watermark information is then embedded in areas that are not easily disturbed by screen-shooting distortions, thus the decoder extraction is robust to these distortions.

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415

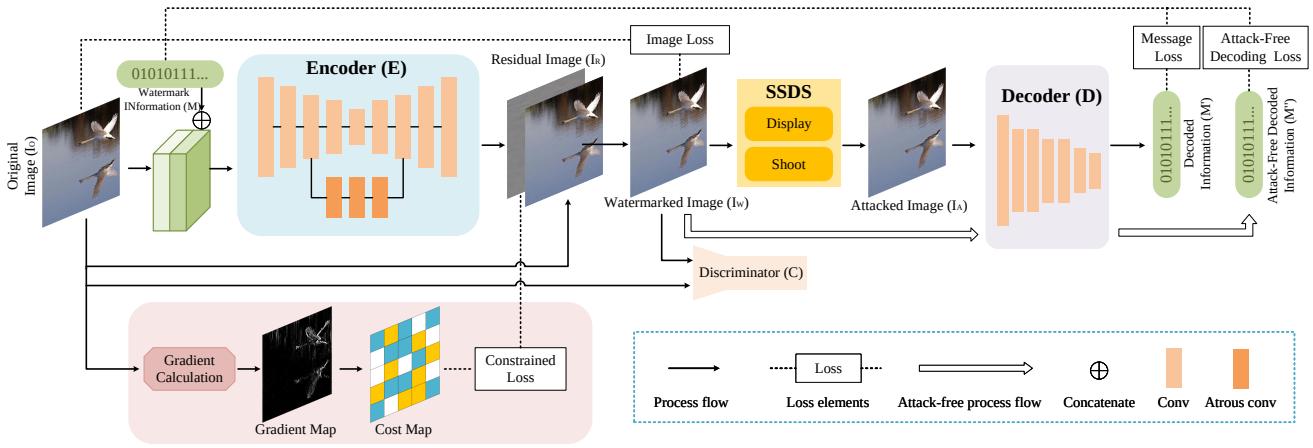
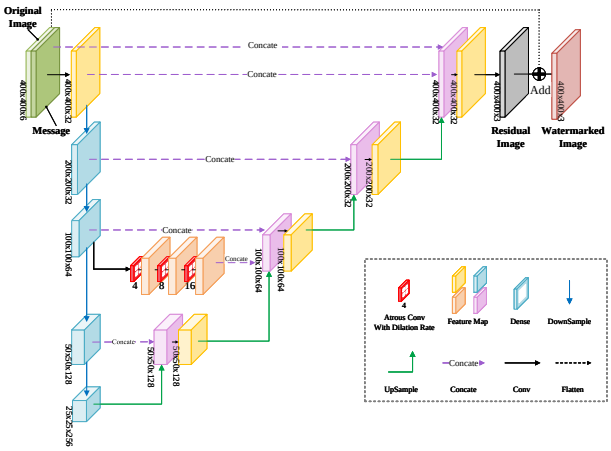
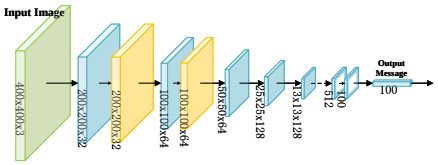


Fig. 4. The overview of the proposed screen-shooting resistant watermark. The original image I_O and the watermark message M are concatenated as the input. The encoder $\mathbf{E}$ extracts features from the input data, and outputs a watermarked image I_W. The watermarked image I_W undergoes distortion simulation and is fed to the decoder $\mathbf{D}$ for extraction. The discriminator $\mathbf{C}$ classifies images and feeds back the result to supervise the generation quality.



(a) Encoder



(b) Decoder

Fig. 5. The network structure of the proposed method. (a) and (b) are the structure of Encoder and Decoder respectively.

B. Screen Displaying Distortion

Even if the viewing point is perpendicular to the screen plane at its center, the viewing angle of points outside the center will not be 0 degrees exactly. The greatest viewing angle will appear at the four corners of the screen. Therefore, the grayscale usually deviates even though it is shot vertically. Besides, the images fed into the Decoder are usually the perspective-calibrated images as shown in the second row of Fig. 6, which tries to ease the impact of perspective warping. However, grayscale deviation still exists. It is necessary to handle it for robustness enhancement. To simulate the effect of grayscale deviation, we need to obtain the viewing angle for each pixel point first. Thus we set up a three dimensional (3D) spatial model to calculate the viewing angle pixel-wise, and then match them with the corresponding points on the projection plane.

Even if the viewing point is perpendicular to the screen plane at its center, the viewing angle of points outside the center will not be 0 degrees exactly. The greatest viewing angle will appear at the four corners of the screen. Therefore, the grayscale usually deviates even though it is shot vertically. To simulate the effect of grayscale deviation, we need to obtain the viewing angle for each pixel point first. Thus we set up a three dimensional (3D) spatial model to calculate the viewing angle pixel-wise, and then match them with the corresponding points on the projection plane.

In the 3D model, point O_v is the viewing point, while plane *ABCD* and plane $A'B'C'D'$ represent the screen and its projection plane respectively (see Fig. 3). Denote the center of the screen as O_s. Line O_vO_s intersects the projection plane $A'B'C'D'$ at point O, and will always be perpendicular to plane $A'B'C'D'$ no matter how the viewing point changes position, because it's the projection. Thus we can set a coordinate system at O, with O_vO as the z'-axis and plane $A'B'C'D'$ as the plane $x'Oy'$. To facilitate the calculation, we restrict line *AB* to be always parallel to the plane $y'Oz'$, as well as ensure an arbitrarily tilted viewing angle. According to these demands, the red coordinate system $x'y'z'$ is set up. When the viewing angle is 0°, plane *ABCD* will be vertical to O_vO_s and parallel to plane $x'Oy'$. Besides, another coordinate system of green is set in Fig. 3, whose origin is O_s with x-axis parallel to BC and y-axis parallel to AB. The red coordinate system can be moved from O to O_s, thus the two coordinate systems are overlapped. It's easy to find out that the angle between the two coordinate systems is γ', which is the viewing angle of the screen. To standardize the representation of x and y throughout the paper, we discuss the green coordinate system with x, y, and z, while the red one uses x', y', and z'. As shown in Fig. 3, the viewing angle of the screen γ' can be decomposed into α and β in the x and y directions. We could draw a vertical line overlapped with the normal direction on plane *ABCD*, which is denoted as $\boldsymbol{n}$. If we keep this line fixed

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415

on O_s and tilt it in xO_sz plane towards the negative direction of x-axis, it will finally meet the projection point of O_v on XoZ plane, and this tilt angle is α, which is negative in this case. Similarly, the viewing angle in the y direction is denoted as β. The viewing angle of the screen γ' is in the range of $(-90°, 90°)$ because the maximum angle at which we can observe a screen is $\pm 90°$. Thus the range of α and β are also $(-90°, 90°)$. For pixel-wise viewing angle γ, its absolute value increases as the pixel point is away from the projection point O'_v. Its decomposed pixel-wise angle $\alpha_{x,y}$ and $\beta_{x,y}$ vary with it, thus $\alpha_{x,y}$ and $\beta_{x,y}$ are different on the diagonal of *ABCD*.

In addition, the distance d from the center of the screen to the viewing point can be set in the 3D model. Let O be the midpoint of O_vO_s. Denote the width and height of the screen as *w* and *h*. It's easy to obtain the coordinates of A, B, C, D, O_v with α, β, *w*, *h*, *d* in red coordinate system. The projection plane can be represented as *z'* $= 0$. A' and B' will be obtained by associating the coordinates of A, B, and O_v with the projection plane. Similarly, C' and D' can be calculated, thus the shape of the projection is obtained. Let O_v be perpendicular to plan *ABCD* at O'_v, whose coordinate can be figured out easily. For every point *P* on the screen plane, its pixel-wise viewing angle γ can be figured out with the coordinates of P, O_v, and O'_v. Thus, the pixel-wise viewing angle is related to the distance to the project point O'_v. The job of the red coordinate system is done after gaining $A'B'C'D'$ and γ. Point *P* can be represented with the green coordinate system as (x, y), and the viewing angles $\gamma_{x,y}$ can be decomposed into $\alpha_{x,y}$ and $\beta_{x,y}$ in x and y directions. The angles are recorded into an angle matrix with the corresponding coordinate. Then, the angle matrix is projected on the projection plane with its new 2D coordinates. The projecting function can be formulated as Eq. (1).

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} e_1 & e_2 & e_3 \\ e_4 & e_5 & e_6 \\ e_7 & e_8 & e_9 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{1}$$

where e_i can be calculated by four pairs of original and corresponding projected point coordinates (x_i, y_i) and (x'_i, y'_i). The coordinates of $ABCD$ and $A'B'C'D'$ are used for it.

As discussed above, the grayscale varies with the viewing angle, and we can formulate its effects in x and y directions separately, because their variance trends are different (see Appendix for more details). For the x direction of shooting top and down at a screen, its grayscale deviation can be formulated by Eq. (2):

$$F_{TD}\left(\alpha_{x,y}, \mathrm{ori}_{x,y}\right) = \frac{NH(\alpha_{x,y})}{2}\left[\cos\left(\omega\alpha_{x,y} + \varphi(ori_{x,y})\right)\right] + \frac{N}{2} \tag{2}$$

$$H\left(\alpha_{x,y}\right) = -\frac{4{\alpha_{x,y}}^2}{\pi^2} + 1 \tag{3}$$

$$\varphi\left(\mathrm{ori}_{x,y}\right) = -\frac{\pi}{N^2} ori_{x,y}^2 + \pi \tag{4}$$

where ω and φ are radian frequency and phase respectively. N represents the max value of pixels, which usually equals 255 for digital images, but we set $N = 1$ because images are normalized to (0,1) for network training. Since grayscale is non-negative, the cosine function is added with $N/2$. Factor H is used to ensure the first term in F_{TD} will tend to 0 at $\pm 90°$, to ensure the simulation for contrast drop as the viewing angle increases. In addition, the value of $\pm 90°$ is based on the viewing angle of screens, which cannot distinguish between white and black at $\pm 90°$. Since the period T is $180°$, the radian frequency factor ω should be $\omega = T/2\pi = 2$. Besides, we try to simulate the observed grayscale in screen-shooting, which should relate to the original grayscale of its digital image. Factor φ is used to introduce the impact of the original grayscale $ori_{x,y}$, which is the phase that could translate the cosine shape along the angle-axis. Based on related theory and data analysis, we find that pixel points with distinct original grayscales follow a similar variance trend like a translation [31], and their distance is represented by φ. The distance becomes larger as the original grayscale increases, thus we use a quadratic to $ori_{x,y}$ in Eq. (4), aiming to speed up the translation with φ.

When shooting a screen in the left-right direction, the variance trend is more simple because the viewing angle is perpendicular to the robbing direction [31]. The observed grayscales converge at large angles in both positive and negative directions, resulting in a mixed gray style of contrast dropping. Based on this discussion, the grayscale deviation in the left-right direction is formulated as Eq. (5).

$$F_{LR}\left(\beta_{x,y}, \mathrm{ori}_{x,y}\right) = S_1(ori_{x,y}) \cdot \beta_{x,y}^2 + S_2(ori_{x,y}) \tag{5}$$

$$\begin{aligned} S_1(ori_{x,y}) =& \frac{4(G_{min} - G_{max})ori_{x,y}}{\pi^2 N} \\ &+ \frac{4\left(G_{mid} - G_{min}\right)}{\pi^2} \end{aligned} \tag{6}$$

$$S_2\left(\mathrm{ori}_{x,y}\right) = \mathrm{ori}_{x,y} \tag{7}$$

For factor S_2, it is the observed grayscale when $\beta_{x,y} = 0$, thus it equals the original grayscale in theory. In Eq. (6) of S_1, G_{min} and G_{max} are the minimum and maximum of grayscale, and G_{mid} represents the target converge value at large angles. The constant set of G_{min}, G_{mid} and G_{max} can be chosen randomly from prepared groups in every step, which is beneficial for generalization. For example, G_{mid} can set as 0.4 with G_{min}=0 and G_{max}=N=1, thus the minimum and maximum of S_1 are roughly equal to -0.24 and 0.16. More information about the equations is provided in the Appendix.

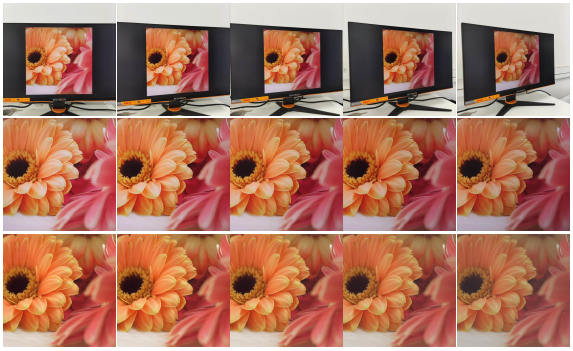
Combining the formulas of both directions, we can get the integrated GDF:

$$\begin{aligned} GDF(\alpha_{x,y}, \beta_{x,y}, ori_{x,y}) =& \delta F_{TD}(\alpha_{x,y}, ori_{x,y}) \\ &+ (1-\delta) F_{LR}(\beta_{x,y}, ori_{x,y}) \end{aligned} \tag{8}$$

where δ represents the weight, ranging in (0.45, 0.55) for generalization. We provide some examples of the GDF-generated images in Fig. 6, in which the first two rows represent the raw data and the perspective-calibrated data of shot images, while the third row is generated by GDF. The similarity between the shot images and generated images indicates the effectiveness of GDF. In addition, we compare the variation trend of GDF and the shot data with Fig. A3 and Fig. A5 in the Appendix. In general, their variance trends are similar to each other with slight differences. In fact, it is hard to quantitatively formulate the exact same distortions within screen-shooting. Introducing

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415



(a) 0° (b) 15° (c) 30° (d) 45° (e) 60°

Fig. 6. Effect of shooting screen at different angles. It shows the visual impact of grayscale deviation, which increases as the viewing angle increases.

the most influenced parts of distortions is enough to gain strong robustness [9].

In addition, there are differences in brightness and contrast between the image displayed on the screen and itself. We simulate the change in brightness by randomly adding or subtracting pixel values in the range of -20% to 20%. For contrast simulation, it is the blending of the original color image and its corresponding gray image at a random ratio. The screen displaying distortion in **SSDS** is composed of the effect of GDF, brightness and contrast.

C. Camera Shooting Distortion

The instant shooting process converts physical patterns into digital images, introducing a severe and complex distortion, which is a compound effect [4], [7]. We decompose the complex distortion and use a set of differentiable distortions to model the same effects. The shooting position is unfixed which introduces perspective warp. The 3D to 2D physical transformation, which is mentioned in grayscale deviation, is the basic of warp simulation. The warp angle can be calculated by the viewing angle of the four corners. Handhold camera shaking causes defocus blur and motion blur. Defocus blur caused by handhold camera shaking is simulated by a Gaussian blur kernel with random standards ranging from 1 to 3 pixels. For motion blur, a straight-line blur kernel is utilized in the convolution layer. Inside the camera, electron components introduce unavoidable noise. The noise is simulated by Gaussian noise with a random standard deviation ranging from 0 to 0.02. The camera senses the light and converts it to electrical signals. The resulting RAW data will be compressed into a JPEG image. The rounding step in JPEG compression is not differentiable, so we use a differentiable JPEG simulation function from Shin and Song [33]. With the above simulation progress, the comprehensive effect of camera shooting can be decomposed into perspective warp, blur, noise, as well as JPEG compression.

IV. Watermark Network

As shown in Fig. 4, the structure of our watermark network consists of four components: (1) the encoder $\mathbf{E}$ takes the concatenated original image I_O and watermark information M as input and outputs the residual image I_R, which is added to I_O to generate I_W; (2) the screen-shooting distortion simulation module **SSDS** receives I_W and distort it to generate the attacked image I_A; (3) the decoder $\mathbf{D}$ recovers watermark information from I_A and I_W, which are denoted as M' and M'' respectively; (4) the discriminator $\mathbf{C}$ evaluates the performance of generation via calculating the probability whether the given image contains watermark or not. In this section, we will describe in detail the composition of each component and corresponding training strategy.

A. Encoder

For the Encoder $\mathbf{E}$ in general, the feature maps are first scaled down and then scaled up in the U-shaped downsampling and upsampling as shown in Fig. 5(a). With skip connections, the shallow and deep features are combined, aiming to make full use of different levels of features. Besides, we use three atrous convolutions with dilation rates of 4, 8, and 16, respectively, to form a pyramid network structure. It does not scale down the feature map while providing a larger field of perception, compensating for the information loss due to downsampling in the U-shaped structure. The output of the last atrous convolution is concatenated to the shallow and deep features in the same shape, which forms a new feature map in the shape of (100, 100, 64). The new map is passed to the following convolutions to generate the residual image, which is finally added to the original image to obtain the watermarked image.

It is beneficial to improve the visual quality of the watermarked image by giving a large weight to embed in regions with boundaries and complex textures [35]. Therefore, we propose a constraint to the encoder based on gradient. Specifically, an image is represented as pixel values and the degree of pixel variation is the gradient, which means the regions where the pixel value varies drastically correspond to a large gradient. For gradient calculation, we use the Sobel operator to calculate the horizontal and vertical gradients so as to obtain the variation gradient map Map_{VG}, which marks texture regions with higher values. Compared to other operators, the Sobel operator does not suppress or filter out non-edge regions, which results in less effective edge extraction but allows the gradient to transition from texture regions to plain ones smoothly. Thus the variation gradient map can be smoother and more stable. After that, the variation gradient map is normalized to the range of 0 to 1 and subtracted from 1 to generate the modification cost map $Map_{MC} \in (0, 1)$. Map_{MC} gives a larger weight to embed into texture regions by applying smaller modification costs to the corresponding areas. Finally, this cost map will work on the outputted residual image I_R with the constrained loss L_{CT}, which will be discussed in Section IV-E. We refer to this whole process as gradient guidance, which aims to constrain the embedding areas to eliminate the visible shadow and optimise the image quality of watermarked images (see Fig. 8).

B. Distortion Module

As we mentioned before, the robustness improvement mainly depends on the noise layer, thus the design of **SSDS**

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415

(a) Origin (b) SSR [8] (c) RIHOOP [7] (d) StegaStamp [4] (e) w/o Guidance (f) Ours

Fig. 7. Qualitative comparisons on the visual quality of watermarked images. The gradient guidance leads the information to be embedded in the edge region, so that visible shadows are spread, enhancing the image quality.

is crucial. Screen-shooting cannot be described directly in digital and is therefore difficult for machines to understand. We try to decompose screen-shooting into two processes, one of which is the screen display of digital images, and the other is the process of camera shooting. For screen displaying, it involves changes in grayscale, contrast and brightness, which can be different on different monitors. We simulate them with a series of image operations. In addition, the grayscale deviation is a special phenomenon in the monitor screen that varies with the viewing angle (see Fig. 6). In this regard, we analyze the principle of screen displaying and model the angle between the observation point and the screen in 3D space, as well as designing GDF to simulate the grayscale deviation. For camera shooting, we analyze the process from pressing the shutter to producing an electronic picture. At the moment the shutter is pressed, the human hand will shake unconsciously and the light-sensitive components will feel the light shift slightly, resulting in a blurred image. The light is then converted into electrical signals that propagates through the electronic components, introducing electronic noise. The raw sampled data represented by the electrical signal is called RAW data, which needs to be digitized and compressed before being outputted as an image. Finally, the camera outputs a compressed JPEG format image. This complex camera process can be divided into several image operations including defocus blur, noise and JPEG compression.

C. Decoder

The Decoder $\mathbf{D}$ is used for the watermark information extraction, as shown in Fig. 5(b). We first use a seven-layer convolution structure to extract image features, in which the feature map is gradually scaled down as the convolution processing with a stride of 2. Then the feature map is flattened and the dimensionality is reduced by two dense layers, so that the output of the last dense layer is the same length as the original watermark information M.

In training, our embedded watermark information is a random bit string of 0 and 1, so extracting the watermark is essentially a binary classification task, i.e., each bit needs to be classified as 0 or 1. Therefore, we use a sigmoid activation function to make the data more aggregated to 0 or 1, then round and output the results. Intuitively, a cross-entropy loss function is used to calculate the difference between the decoded watermark information M' and the original watermark information M. When the difference between the two gradually decreases as the training processes, it means that the decoder's accuracy is gradually improving.

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415

TABLE I
ABLATION STUDY: THE IMAGE QUALITY AND BIT ACCURACY.

Modules		Image Quality		Bit Acc		
GDF	Guidance	PSNR	SSIM	0°	30°	60°
×	×	29.41	0.9657	0.9730	0.9230	0.8323
×	✓	31.76	0.9630	0.9840	0.9429	0.8375
✓	×	30.33	0.9696	**0.9940**	**0.9804**	**0.9059**
✓	✓	**34.17**	**0.9821**	0.9870	0.9607	0.8692

TABLE II
ABLATION STUDY: THE IMAGE QUALITY AND BIT ACCURACY WITH AND WITHOUT ATROUS COMVOLUTIONS.

Module	Image Quality		Bit Acc		
	PSNR	SSIM	0°	30°	60°
w/o atrous conv	33.29	0.9806	0.9860	0.9597	0.8577
w/ atrous conv	**34.17**	**0.9821**	**0.9870**	**0.9607**	**0.8692**

D. Discriminator

The discriminator $\mathbf{C}$ is used to evaluate the quality of the generated watermarked images, and its role is to assist in training the generative model. The performance of the discriminator should be within a moderate range. On the one hand, it can simply distinguish whether the input image is generated or not, thus guiding the encoder to generate watermarked images with better quality. On the other hand, it should not be so powerful that the discriminator continuously negates the output of the encoder, resulting in a lack of positive feedback for the encoder and leaving the encoder with no idea how to optimize. So we introduce the Wasserstein distance [29] in the discriminator training. The superiority of the Wasserstein distance is that it can reflect the distance between two distributions even if they do not have any overlap. Thus no matter how far away the two distributions are, the gradient still exists. It means that the parameters are updatable during training, which overcomes the problem of vanishing gradients.

E. Loss Function

The training process can be divided into three phases with different tasks. In the first phase, the distortion is introduced to train the model in a targeted manner, which only focuses on improving the robustness of the model with the message decoding loss L_M based on cross-entropy in Eq. (9).

$$L_M = -\sum_{i=1}^{n}[M_i \log(\mathbf{D}(I_A)) + (1-M_i)\log(1-\mathbf{D}(I_A))] \quad (9)$$

where n indicates the training batch size, and M_i is the inputted watermark information. $\mathbf{D}(I_A)$ represents the decoded information of I_A, which is the attacked image output by **SSDS**.

Through the first training phase, the trained model is robust but poor in image quality. Thus the primary target of the second phase is to improve the performance in image quality. We add several visual-related losses to fully exploit the potential of the encoder, including L_V, L_{CT} and L_C. The image visual loss L_V is based on L2 and LPIPS [30] while the constrained loss L_{CT} is obtained by multiplying the modification cost map with the residual image (see Eq. (11)). These two loss functions are added to the training along with the Wasserstein distance [29] discriminator loss L_C.

$$L_V = \frac{\sum_{i=1}^{n}(I_W - I_O)^2}{n} \times 1.5 + LPIPS(I_W, I_O) \quad (10)$$

$$L_{CT} = Map_{MC} \times I_R \quad (11)$$

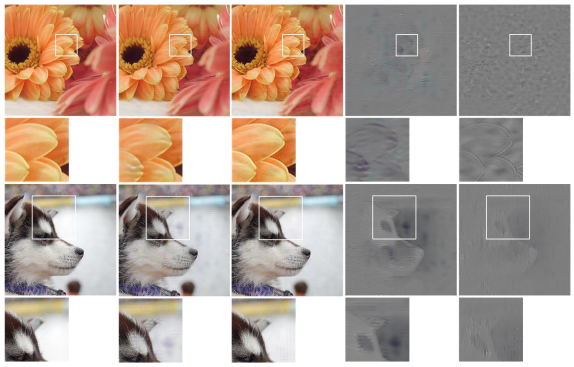
$$L_C = Wasserstein(I_W, I_O) \quad (12)$$

where I_R represents the residual image, which will be added to the original image I_O to generate watermarked image I_W. Map_{MC} is the modification cost map, which ranges from 0 to 1. With these losses, the embedded information gradually incorporates with the image in the textures until it is visually invisible, thereby optimizing the visual quality.

Finally, we find that the distortion simulation module designed in pursuit of high robustness might lead to overfit the distortion, making the extract accuracy decline when decoding watermarked images without attacks. To solve this problem, the attack-free decoding loss L_{AF} is introduced in the last phase. We add a branch to directly decode the output watermarked image from the encoder while maintaining the original decoding branch. The two branches work together in the third phase to guarantee robustness in both attack and attack-free decoding. They share a similar loss function of

TABLE III
ABLATION STUDY: THE BIT ACCURACY WITH AND WITHOUT GDF UNDER DIFFERENT SHOOTING DISTANCE.

Module	Distance (cm)					
	50	60	70	80	90	100
w/o GDF	0.9890	0.9900	0.9930	0.9900	0.9845	0.9840
w/ GDF	**0.9920**	**0.9959**	**0.9981**	**0.9960**	**0.9905**	**0.9870**



(a) Origin (b) Watermarked image (c) Residual image

Fig. 8. Qualitative results for comparison with the ablation result of gradient guidance on image quality. (a) is the original image. (b) and (c) are the watermarked image and the residual image respectively. The first column in both (b) and (c) is generated by the model without gradient guidance, and the second column is generated by the model with gradient guidance.

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415

TABLE IV
ABLATION STUDY: THE IMAGE QUALITY OF THE MODEL TRAINED WITH DIFFERENT LOSSES.

Losses			Image Quality	
L_V	L_C	L_{CT}	SSIM	PSNR
✓	×	×	0.9538	25.29
✓	✓	×	0.9696	30.33
✓	×	✓	0.9819	33.82
✓	✓	✓	**0.9821**	**34.17**

TABLE V
COMPARISONS OF THE IMAGE QUALITY AND BIT ACCURACY WITH REPLACING THE EDGE DETECTION METHOD IN GRADIENT GUIDANCE.

Method in Gradient Guidance	Image Quality		Bit Acc		
	PSNR	SSIM	0°	30°	60°
DexiNed	32.41	0.9785	0.9534	0.8334	0.6929
Canny	31.91	0.9771	0.9830	0.9227	0.8181
Sobel	**34.17**	**0.9821**	**0.9870**	**0.9607**	**0.8692**

cross-entropy, but the attack-free decoding loss L_{AF} inputs the unattacked watermarked image I_W (see Eq. (13)).

$$L_{AF} = -\sum_{i=1}^{n}[M_i \log(\mathbf{D}(I_W)) + (1 - M_i)\log(1 - \mathbf{D}(I_W))] \tag{13}$$

The entire loss function consists of the following parts in Eq. (14): 1) the message decoding loss L_M. 2) the image visual loss L_V. 3) the constrained loss L_{CT}. 4) the discriminator loss L_C. 5) the attack-free decoding loss L_{AF}.

$$Loss = \lambda_1 L_M + \lambda_2 L_V + \lambda_3 L_{CT} + \lambda_4 L_C + \lambda_5 L_{AF} \tag{14}$$

The parameter λ is used to control sub-losses in different phases, including whether to add sub-losses in the current training phase, as well as adjusting for a tradeoff between cross-media robustness, image quality and attack-free decoding accuracy. In the first phase, the parameter λ_1 increases from 0 to 2 uniformly as the training steps increase, while the other weighting coefficients are set to 0. Thus, the first phase only contains supervised training for robustness, the watermarking model will focus on improving the decoding capability. In the second phase, the parameter λ_1, which has finished increasing, is now fixed. And the parameter λ_2, λ_3, and λ_4 start to increase uniformly to 1, 1, and 1.5 respectively. These image quality related loss functions are given increased weights, which enables the whole network to work on optimizing the visual quality. The third phase builds upon the first two, allowing the parameter λ_5 increasing uniformly from 0 to 1.5, thus the attack-free decoding loss is added to the supervised training. The robustness is first supervised by L_M in the first phase, and then influenced by both L_M and L_{AF} in the third phase, in order to guarantee the correct extraction under attack or not. Overall, the targets of the three phases are different, thus we need to ensure a smooth transition between the phases. In this case, the weighting coefficients are increased uniformly from 0 to λ_i in each phase. With this training strategy, the performance of the corresponding tasks can be targeted to improve at different phases.

V. EXPERIMENTAL RESULTS AND ANALYSIS

In this section, we first provide the details of the experimental setup. Secondly, the effectiveness of the grayscale deviation function and gradient guidance is verified by ablation experiments. Then, we compare the proposed method with some state-of-the-art watermarking methods, including image quality and robustness under different shooting distances and viewing angles. Finally, we perform more tests with different device combinations.

A. Experimental Setting

To compare with the previous methods, we use the Mirflickr database [34], which was used by StegaStamp [4]. Mirflickr contains a variety of natural images, such as landscapes, near objects, plants, animals, and people. We split the training and testing set with a ratio of 4:1, in which the images are center cropped and resized to the size of 400 × 400. The embedded watermark information is a 100 bits string, which is generated randomly for training and testing. In addition, the training progress is conducted with NVIDIA GeForce RTX 2080 Ti GPU. For experiments involving manual shooting, we randomly select 100 images from the test set and shoot them under different test conditions, such as different distances and angles. For the same test condition, we fix the shooting position with a tripod and shoot with the help of Bluetooth to avoid moving the position when touching the camera. Moreover, the images are shot continuously under the same test condition in comparison experiments. These shooting controls ensure a fair comparison. The experiments are conducted with the screen of LG 22M37A and the phone camera of Mi 10 Lite. In addition, more combinations of devices were tested in Section V-D to verify the generalization of the proposed method, including screens of AOC 215LM00020 and ENVISION G249G as well as cameras of vivo x7 and iPhone XR. For test metrics that do not involve manual shooting, such as image quality evaluation, we use the full test set which contains 5,000 images.

Generally, watermarking methods focus on two properties: the watermarked image's visual quality and the watermark's robustness. For visual quality, we perform a qualitative analysis by providing the visualizations of watermarked images and residual images, as well as utilizing PSNR and SSIM as quantitative evaluation metrics. And for robustness, we measure it with bit accuracy and decoding accuracy. Bit accuracy is the rate of correctly decoded bits over all bits in one extraction. Decoding accuracy represent the rate of completely correct extractions to all extractions.

B. Ablation Experiments

To verify the effectiveness of the scheme, we conducted ablation experiments on different modules. We first give ablation results for the designed modules in Table I and II. Then we conduct more ablation experiments under different situations to validate the robustness effect of GDF and the invisibility effect of gradient guidance respectively.

Table I shows the individual effect of GDF and gradient guidance. Training with gradient guidance gets better performance in visual quantity than training without it. Meanwhile,

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415

TABLE VI
ABLATION STUDY: THE BIT ACCURACY WITH AND WITHOUT GDF UNDER DIFFERENT SHOOTING ANGELS.

Module	Angle (°)										
	left&down70	left&down60	left&down45	left&down30	left&down15	vertical	right&up15	right&up30	right&up45	right&up60	right&up70
w/o GDF	0.7460	0.7865	0.8658	0.9060	0.9543	0.9840	0.9418	0.9400	0.9408	0.8780	0.8148
w/ GDF	**0.8178**	**0.8433**	**0.8840**	**0.9428**	**0.9880**	**0.9870**	**0.9798**	**0.9785**	**0.9640**	**0.8950**	**0.8338**

TABLE VII
COMPARISON OF THE IMAGE QUALITY AND BIT ACCURACY IN DIFFERENT SHOOTING CONDITIONS WITH THE STATE-OF-THE-ART WATERMARKING METHODS.

Methods	Image Quality		Bit Acc		
	PSNR	SSIM	0°	30°	60°
StegaStamp [4]	28.21	0.9290	0.9345	0.8874	0.7151
RIHOOP [7]	31.14	0.9622	0.9700	0.9464	0.8263
SSR [8]	26.99	0.8719	0.9523	0.7785	0.5836
PIMoG [9]	**36.19**	0.9808	0.9283	0.8054	0.6263
Ours	34.17	**0.9821**	**0.9870**	**0.9607**	**0.8692**

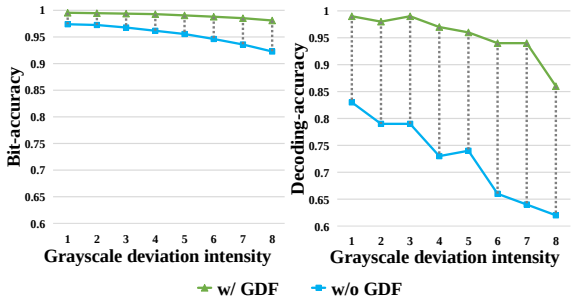


Fig. 9. Ablation study: The bit accuracy and decoding accuracy with and without GDF under different grayscale deviation intensity.

the results indicate that GDF contributes to the portion of the model's robustness, which enhance the bit accuracy at all angles. Even though GDF is designed for robustness enhancement, it also involves the texture features by calculating the original grayscales, which improves the invisibility slightly. In addition, comparing rows 2∼4, the increased PSNR value of 34.17 attributes to the combined effects of GDF and gradient guidance, which are connected by grayscale calculation, making a greater impact together than individual efforts. In Table II, the method with atrous convolutions reaches higher scores, indicating that our proposed method can achieve great performance with atrous convolutions.

For the robustness, we focus on the GDF module design and give more detailed information in Table III and Table VI. GDF is beneficial to improve the robustness of the model, and the model with GDF can better resist the effects from distortion at different angles and distances. The model with GDF outperforms the model without GDF in all cases. Robustness is of great importance in the screen-shooting resistant task. To further verify the effectiveness of GDF, we conducted tests under different grayscale deviation intensities, shown in Fig. 9. It can be seen that the model without GDF has a significant decrease in both the bit correct rate and decoding correct rate. In worst cases, it can drop to a decoding accuracy of 0.6, while the model with GDF maintains high performance. As the distortion intensity increases, the model with GDF gains a larger advantage over the one without GDF, which indicates the effectiveness of GDF. Besides, the reason why a single model maintains satisfactory performance under multiple angles is that its angles are taken randomly in the training, which enhances its generalization.

In addition, visual quality evaluations are conducted in the ablation study. The first columns in Fig. 8(b) and Fig. 8(c) are generated by models without gradient guidance, while their second columns are generated by models with gradient guidance. The residual images generated by the model without gradient guidance show irregular visible shadows, which are likely caused by watermark embedding. Suppose the watermark is embedded in areas that are not easily visible to the human eye, such as boundary regions. In that case, the visual quality of the generated watermarked images will be better. On the other hand, the residual images generated by the model with gradient guidance have sharper edges. Comparing the last two images in the fourth row, the residual image of the puppy without gradient guidance shows gray shadows on the wall and the edges of its hair are ignored as smooth lines. We note that the puppy residual image with gradient guidance avoids visible shadows. Furthermore, it correctly identifies the hair edges and diffusely embeds the information into the boundary texture. This shows that the gradient guidance module indeed helps to improve the visual quality of the images. Besides, we conduct ablation study for the losses in the second phase, which aim at improving invisibility. The results are shown in Table IV. Loss L_V provide basic ability for invisibility, while L_{CT} and L_C fine tune it. Compared with L_C, the proposed L_{CT} have higher score in both SSIM and PSNR.

C. Comparison Experiments

To verify the effectiveness of our method, we performed comparison experiments with some state-of-the-art methods, including StegaStamp [4], RIHOOP [7], SSR [8] and PIMoG [9]. We have experimentally tested the visual quality and robustness under different conditions. The experimental results show that our method has better performance.

1) Image Quality Comparison: In this experiment, we fairly embed watermarks on 5000 images and compare the invisibility of watermarked images generated with different methods. We give the qualitative comparison results in Fig. 7, where the last column shows our watermarked images. Compared to the other methods, our watermarking method reduces the visible shadows. This is because the gradient guidance allows the embedding information to diffuse into

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415

TABLE VIII
COMPARISON OF BIT ACCURACY FOR EXTRACTED WATERMARK MESSAGE UNDER DIFFERENT SHOOTING ANGELS.

Methods	Angle (°)										
	left&down70	left&down60	left&down45	left&down30	left&down15	vertical	right&up15	right&up30	right&up45	right&up60	right&up70
StegaStamp [4]	0.5975	0.6233	0.7503	0.8327	0.9143	0.9345	0.9610	0.9420	0.8860	0.8068	0.7263
RIHOOP [7]	0.7318	0.7753	**0.8863**	0.9173	0.9563	0.9700	**0.9875**	0.9755	0.9418	0.8773	0.8263
SSR [8]	0.5871	0.5621	0.6524	0.7211	0.7496	0.9523	0.8848	0.8359	0.7207	0.6051	0.5957
PIMoG [9]	0.5867	0.6067	0.7042	0.7900	0.8325	0.9283	0.9083	0.8208	0.7483	0.6458	0.5858
Ours	**0.8178**	**0.8433**	0.8840	**0.9428**	**0.9880**	**0.9870**	0.9798	**0.9785**	**0.9640**	**0.8950**	**0.8338**

TABLE IX
COMPARISON OF BIT ACCURACY FOR EXTRACTED WATERMARK MESSAGE UNDER DIFFERENT SHOOTING DISTANCE.

Methods	Distance (cm)					
	50	60	70	80	90	100
StegaStamp [4]	0.9711	0.9787	0.9890	0.9721	0.9785	0.9345
RIHOOP [7]	**0.9945**	0.9934	0.9975	0.9925	0.9885	0.9700
SSR [8]	0.9323	0.9656	0.9633	0.9398	0.8781	0.9523
PIMoG [9]	0.9533	0.9567	0.9683	0.9467	0.9467	0.9283
Ours	0.9920	**0.9959**	**0.9981**	**0.9960**	**0.9905**	**0.9870**

TABLE X
BIT ACCURACY OF EXTRACTED WATERMARK WITH SCREEN SHOOTING UNDER DIFFERENT SCREENS AND CAMERAS.

Screen	Camera	Min	Median	Max	Mean
LG 22M37A	Mi 10 Lite	0.97	0.99	1.00	0.9925
	vivo X7	0.97	1.00	1.00	0.9950
	iPhone XR	0.98	0.99	1.00	0.9942
ENVISION G249G	Mi 10 Lite	0.97	0.99	1.00	0.9937
	vivo X7	0.98	1.00	1.00	0.9960
	iPhone XR	0.98	1.00	1.00	0.9953
AOC 215LM00020	Mi 10 Lite	0.97	1.00	1.00	0.9947
	vivo X7	0.98	1.00	1.00	0.9965
	iPhone XR	0.98	1.00	1.00	0.9952

the boundary texture, so that they are not easily observed by the human eye. We give the quantitative comparison results in Table VII. Our method has a high bit accuracy at different shooting angles, which is recorded as an average. We achieve the best performance of robustness in this comparison. Since our method focuses on improving robustness, there is a compromise in terms of visual quality. Our method is sight inferior to PIMoG [9] in PSNR, but it outperformed the other methods and is the best in the measurement of SSIM.

2) Robustness Comparison Under Different Perspective Angles: The result of bit accuracy under different shooting angles is shown in Table VIII. We have shot from the bottom left to the top right corner in the range of 0 to 70 degrees. The experimental results show the advantage of our proposed method, especially that it can maintain a higher bit accuracy though the larger shooting angles. This outstanding performance can be attributed to the grayscale deviation function proposed in the scheme. GDF combines the deviation of the grayscale with the observation angle and therefore better fits the effect of the angle change. Besides, we notice that the proposed scheme is slightly worse than RIHOOP [7] in a few results. However, our scheme gains larger advantages in most cases, especially at large angles. It is because at smaller tilt angles, the image is less grayscale deviated and more sensitive to luminance, and the light simulation module in RIHOOP may play a role in it.

3) Robustness Comparison Under Different Distances: The shooting distance is also an important evaluation for screen shooting watermarking. However, this experiment is related to a number of factors, such as the size of the monitor screen. If the screen is larger, then the image obtained by the camera from the same distance is also larger, and the information loss is relatively smaller. The screen we used in the distance test is in the size of 39cm $\times$ 29cm. Although there are differences in size and resolution between different screens, we can still simply compare with other methods under the same conditions to evaluate the performance. The relevant test results are shown in Table IX. As can be seen, our method has certain advantages. In the experiments of distance, we keep the camera vertical to screen while shooting. All methods perform well and their difference is not significant. At closer distances, e.g. 50cm, the RIHOOP method [7] performs slightly better. But in most cases, our method performs the best bit accuracy.

Gradient guidance plays an important role in enhancing image quality. Within it, the Sobel operator is used to calculate the gradient, which is uncomplicated but suitable for this work. In Table V, we provide comparison experiments by replacing the Sobel operator with other edge detection methods, including the Canny operator, and an edge detection network named DexiNed [39]. The method with Sobel achieves higher scores in bit accuracy, because of its guidance to significant areas. Specifically, advanced arts gain finer boundaries by connecting subtle and intermittent textures [35], [39], but the detected edges involve tiny edges that can be corrupted by the high strength distortion in SSDS. In this case, the Sobel operator is enough for our task with less complexity.

D. Evaluation With Different Devices

We conducted more tests with different cameras and screen display devices. The results of the tests are shown in Table X. It can be seen that our method has high decoding accuracy for different device combinations. It verifies that our method has good generalizability and can be widely used on different devices, indicating well application prospects. We used a specially designed positioner to locate and crop images, which is also used to other methods for fair comparisons. In order

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415

to present the performance objectively, we did not apply additional image enhancement methods to the captured images. The extraction rate of the designed method can reach a speed of 50 frames per second on a normal laptop (NVIDIA GeForce 940M GPU), which is sufficient to achieve a satisfactory real-time interaction speed in practical applications. Generally, the decoding accuracy decline for watermarking methods may cause mistakes in leakage tracing. Thus researchers take measures to enhance robustness. For the proposed method, we design GDF which improves robustness, especially at large angles, and utilize BCH to enable the message to be recovered exactly as long as the bit accuracy is above 95%. There is no method that guarantees complete decoding in any case, and we are just trying to improve the accuracy as much as possible.

VI. Conclusion

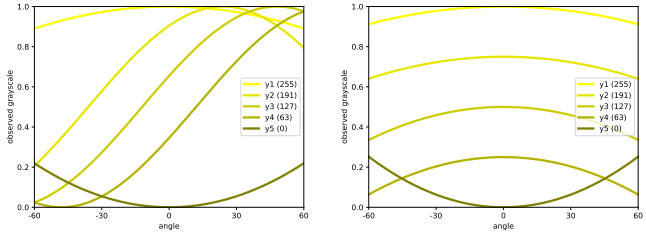
In this paper, we perform an analysis of watermarking under the cross-media progress, especially the grayscale deviation that is typically encountered in screens. Based on this analysis, GDF is proposed to approximate the distortion, which is calculated by the optical imaging principle in distinct viewing angles. GDF, together with a series of image operations, contributes to the SSDS module. By training with SSDS in the watermarking framework, the robustness towards screen displaying and camera shooting can be guaranteed. In addition, for the visible shadow caused by embedding, we design a gradient-guided encoder to spread the concentrated shadow to the texture region. It's not only beneficial to the image quality but also optimizes the cross-media robustness. The experimental results demonstrate that our method is effective and performs well in comparisons with the state-of-the-art methods. For screen-shooting resistant watermarking, the moiré pattern problem is gradually gaining attention, and although the model does not provide special treatment for moiré, it can still resist slight moiré patterns. Our future work is to analyze moiré to further enhance the moiré-resistant performance.

Appendix
Detailed explanation for GDF Equations

This appendix provides more information about the proposed grayscale deviation function. In Section A, we describe the relationship between grayscale and the viewing angle with the theoretical knowledge of screens and data analysis of real-shot images. Section B formulates the grayscale deviation effects based on the features summarized in Section A.

A. The relationship between grayscale and the viewing angle

According to the birefringence effect of liquid crystal in screens, the grayscale will vary with the view angle. There is a rubbing step in the fabrication of the screens. The variation trend of grayscale is different in the perpendicular and parallel rubbing direction [32], as shown in Fig. A1(a) and Fig. A1(b). The abscissa is viewing angles, and the ordinate is the observed grayscale. There are 5 lines of gradient colors, representing the original grayscale of 0, 63, 127, 191, 255. We can simply discuss grayscales on color screens, because the colors come from color films, such as RGB film, which turns the white light into red, green, and blue light. Every pixel on a screen contains three lights, and their various intensities contribute to the colors. The discussed grayscale is actually the intensity of light.



(a) top-down direction (b) left-right direction

Fig. A1. The theoretical variance trend of the observed grayscale that varies with the viewing angle. (a) and (b) are the variance trend in the top-down direction and left-right direction respectively.

Take the second bright line of 191 for example, its grayscale declines as the angle increases on both sides in Fig A1(b). Meanwhile, in the parallel direction of Fig A1(a), its grayscale declines on one side, but increases on the other. It matches the following examples: 1) the grayscale observed on the left or right always decreases as the viewing angle increases. 2) the grayscale always declines when observed from the bottom, and it increases when observed from the top. We provide a few photos of this situation in Fig. A2(a).

(a) LG (b) ENVISION

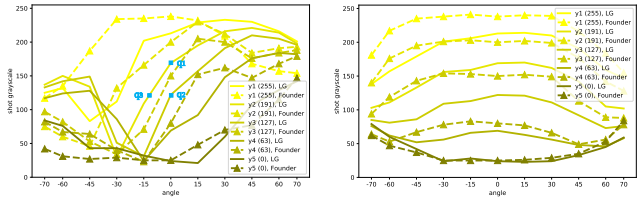
Fig. A2. The phenomenon of grayscale deviation when shooting screens at distinct viewing angles.

As technology advances, special films, in-plane switching (IPS), multidomain vertical alignment (MVA), and optically compensated bend (OCB) are proposed to avoid light leakage for wider viewing angles. Their grayscale variation may be slightly different from the theory value in Fig. A1, but they still fit into a similar mold of liquid crystal tilt [31]. We shoot on the left-right and bottom-top direction with different screens, including LG 22M37A and Founder F201HV. With new technologies, the grayscale of these screens no longer changes intensely, enjoying a better viewing experience in wider angles, as shown in Fig. A2(b). The real-shot data are likely to be different from the theoretical data, thus we collect real-shot data and provide their variance trend in Fig. A3.

By analyzing the variance trend in Fig. A3(a), we can summarize the features of grayscale deviation in the parallel direction as follows:

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415



(a) top-down direction (b) left-right direction

Fig. A3. The variance trend of the observed grayscale in real-shot images that varies with the viewing angle. (a) and (b) are the variance trend in the top-down direction and left-right direction respectively.

(a) 80° (b) 85°

Fig. A4. The phenomenon of contrast drop at large angles. (a) and (b) are the shot images at 80° and 85° respectively.

a-1 The lines of different original grayscale converge at large angles.

a-2 The shape of lines is similar to sine or cosine.

a-3 Different grayscales of lines seem obtainable by translating one of them.

a-4 The distance between bright lines is greater than that of dark lines.

By analyzing the variance trend in Fig. A3(b), we can summarize the features of grayscale deviation in the perpendicular direction as follows:

b-1 The lines of different original grayscale converge at large angles.

b-2 The bright lines decline as the angle increases on both sides, while the dark lines increase as the angle increases.

b-3 The gradient of the brightest line (pixel value=255) is greater than that of the darkest line (pixel value=0).

For features a-1 and b-1, the convergence is due to contrast dropping in large angles. If we observe screens at large angles, such as 89°, the bright and dark pixels will become indistinguishable, ending in a mixed gray style. Fig. A4 shows examples of shot images at large angles, which suffer from contrast dropping.

For feature a-2, the cosine shape is caused by both the liquid crystal angle and viewing angle, since the liquid crystal path twists under an electronic-like helical structure [32]. The strength of the twist corresponds to the pixel value. In Fig. 2, the liquid crystals with a pixel value of 255 will own a liquid crystal angle θ of 90°, which enjoys the most light transmittance. Theoretically, for a point with a pixel value of 127 ($\theta = 45°$), if we shoot the screen at a proper viewing angle like 45°, it will meet most of the light, leading to grayscale increasing. If shooting on the other side, the grayscale will decline as it is away from the best angle. This matches a sine or cosine style. Oppositely, the darkest line meets the least light at a viewing angle of 0°, so the grayscale increases as the viewing angle increases. The dark line also follows the sine or cosine style, but it is half a period away from the bright line.

In addition, we will take an example to explain feature a-3. If we fix the viewing angle in 0° while decreasing the pixel value from 191 to 127, it's like drawing a vertical line in Fig. A3(a), and the observed grayscale changes from q_1 to q_2. However, the latter value can be obtained by the same line of 191 at another angle actually (see q_3 in Fig. A3(a)). It shows that the bright and dark lines can obtained by translating one of them, indicating that a proper phase position in the sine or cosine style can change a line from bright to dark.

As for feature a-4, it means the phase between line-0 and line-63 is smaller than that between line-255 and line-191. We believe that new technology, such as special films [36], helps avoid light leakage, which makes the variance of dark lines flatter and shortens the distance of dark lines. It also explains feature b-3.

B. Simulating Grayscale Variation with Equations

The variation mold of grayscale is called grayscale deviation, and we try to simulate it with equations based on the findings in Part A. The effect of grayscale deviation can be decomposed into parallel and perpendicular rubbing directions, and we represent them with α and β in x and y directions, as shown in Fig. 3. The tilt angle of α and β are also the viewing angles of point O_s. For every point (x, y) on screen plane, we can calculate their $\alpha_{x,y}$ and $\beta_{x,y}$ with given x, y, α, β and length of O_vO_s. Thus, we have viewing angles for every point in both parallel and perpendicular rubbing directions. Later in the material, we will introduce how we design the equations in both directions respectively, and then describe the combined formula as well as its effects.

1) Mathematical derivation of grayscale deviation in parallel direction of α

We have analyzed the features of Fig. A3 in Part 1, and try to design equations based on them. The horizontal axis in Fig. A3(a) is $\alpha_{x,y}$ and Fig. A3(b)'s is $\beta_{x,y}$, which are known as they can be calculated. The ordinate axis is the observed grayscale, and we use cosine as the basic function according to feature a-2:

$$F_{TD}\left(\alpha_{x,y}, \mathrm{ori}_{x,y}\right) = \frac{NH(\alpha_{x,y})}{2}\left[\cos\left(\omega\alpha_{x,y} + \varphi(ori_{x,y})\right)\right] + \frac{N}{2} \tag{A1}$$

where ω and φ are radian frequency and phase respectively. N represents the max value of pixels, which usually equals 255 for digital images. However, we normalize the input images to range (0,1) for network training, so $N = 1$ in this case. The $N/2$ added here is used to move up the horizontal center line of the cosine function from 0 to $N/2$ in Fig. A3. Factor H is based on feature a-1:

$$H\left(\alpha_{x,y}\right) = -\frac{4{\alpha_{x,y}}^2}{\pi^2} + 1 \tag{A2}$$

Since the observed grayscale converges at large angles at both sides in the parallel direction, factor H is used to ensure the first term in F_{TD} will converge to 0 at $\pm 90°$. In addition, the value of $\pm 90°$ comes from the viewing angle of screens, which cannot distinguish between white and black at $\pm 90°$. Since the period T is 180°, the factor ω should be $\omega = T/2\pi = 2$.

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415

For factor φ, it presents the impact of the original grayscale $ori_{x,y}$ based on features a-3 and a-4, which translate the cosine function. Factor φ can be formulated by:

$$\varphi\left(\mathrm{ori}_{x,y}\right) = -\frac{\pi}{N^2} ori_{x,y}^2 + \pi \tag{A3}$$

Ideally in Fig. A1(a), the brightest line of 255 has $ori_{x,y} = N$ and $\varphi = 0$, while the darkest line of 0 has $ori_{x,y} = 0$ and $\varphi = \pi$, which matches the figure shape. The quadratic to $ori_{x,y}$ aims to speed up the decline of φ, so that bright lines have greater distance than the dark lines, which satisfies feature a-4.

2) Mathematical derivation of grayscale deviation in perpendicular direction of β

In the perpendicular direction, its figure is much more flatter with slight grayscale variation at large angles. Due to feature b1 and its rugby-like shape, we utilize a power function to simulate it:

$$F_{LR}\left(\beta_{x,y}, \mathrm{ori}_{x,y}\right) = S_1(ori_{x,y}) \cdot \beta_{x,y}^2 + S_2(ori_{x,y}) \tag{A4}$$

For factor S_2, it is the observed grayscale when $\beta_{x,y} = 0$, thus it equals the original grayscale in theory, like the theory figure in Fig. A1(b). However, the grayscales do not range from (0, N) in the shot data due to the phenomenon of contrast drop in screens, which is shown in Fig. A3(b). We decide to handle this drop with the contrast simulation in **SSDS**, and still use the original grayscale as S_2 for GDF:

$$S_2\left(\mathrm{ori}_{x,y}\right) = \mathrm{ori}_{x,y} \tag{A5}$$

According to features b-2 and b-3, factor S_1 determines the curvature of lines, which should match the following requirements:

r-1 $S_1 > 0$ when $ori_{x,y} = 0$.

r-2 $S_1 < 0$ when $ori_{x,y} = N$.

r-3 $abs(S_1)$ is smaller when $ori_{x,y} = 0$ compare to that when $ori_{x,y} = N$.

Besides, grayscales of the lines converge at large angles (see feature b-1). Let G_{mid} be the target converge value. Thus:

r-4 F_{LR} will pass through point $(-\pi/2, G_{mid})$ and point $(\pi/2, G_{mid})$ in the limit angles of $\pm 90°$.

According to requirement r-4, factor S_1 follows $S_1 = 4(G_{mid} - S_2)/\pi^2$. As discussed above, the range of S_2 is $[0, N]$. Let this range be $[G_{min}, G_{max}]$. Thus:

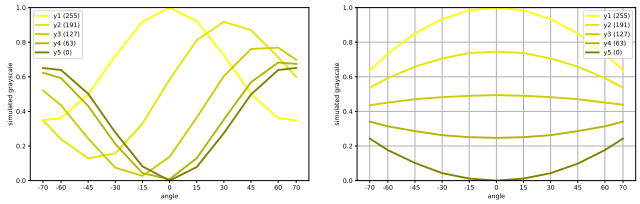
r-5 The range of S_1 should be $[4(G_{mid} - G_{max})/\pi^2, 4(G_{mid} - G_{min})/\pi^2]$.

Based on the requirements above, we use a linear equation to simulate S_1:

$$S_1\left(\mathrm{ori}_{x,y}\right) = k \cdot ori_{x,y} + c \tag{A6}$$

With the discussion of the requirements, factor S_1 is regarded as a line in the simulation, and it could be easy to get its factors:

$$c = \max\left(S_1\right) = \frac{4\left(G_{mid} - G_{min}\right)}{\pi^2} \tag{A7}$$



(a) top-down direction (b) left-right direction

Fig. A5. The variance trend of the simulated grayscale that varies with the viewing angle. (a) and (b) are the variance trend in the top-down direction and left-right direction respectively.

$$\begin{aligned} k &= \frac{min(S_1) - max(S_1)}{N - 0} \\ &= \frac{[4(G_{mid} - G_{max})/\pi^2] - [4(G_{mid} - G_{min}]/\pi^2)}{N} \end{aligned} \tag{A8}$$

Organize these formulas, and we can get:

$$\begin{aligned} S_1(ori_{x,y}) =& \frac{4(G_{min} - G_{max})ori_{x,y}}{\pi^2 N} \\ &+ \frac{4\left(G_{mid} - G_{min}\right)}{\pi^2} \end{aligned} \tag{A9}$$

$$\begin{aligned} &F_{LR}(\beta_{x,y}, ori_{x,y}) \\ &=[\frac{4(G_{min} - G_{max})ori_{x,y}}{\pi^2 N} + \\ &\frac{4(G_{mid} - G_{min})}{\pi^2}]\beta_{x,y}^2 + ori_{x,y} \end{aligned} \tag{A10}$$

In F_{LR}, factor $\beta_{x,y}$ and $ori_{x,y}$ are the variables, while the other factors are constants. We can find some groups of values for the constants in training. Concretely, the value of G_{mid} can be 2/5 according to Fig. A3(b). With $G_{min} = 0$ and $G_{max} = N = 1$, the minimum and maximum of S_1 are roughly equal to -0.24 and 0.16, which matches all the requirements. Other proper groups of values are allowed of cause, and we suggest training by randomly choosing a constant group from the prepared groups in every step, which is beneficial for generalization.

3) The final formula of both directions and its effects

Combining the formulas of both directions, we can get the integrated GDF:

$$\begin{aligned} GDF(\alpha_{x,y}, \beta_{x,y}, ori_{x,y}) =& \delta F_{TD}(\alpha_{x,y}, ori_{x,y}) \\ &+ (1 - \delta) F_{LR}(\beta_{x,y}, ori_{x,y}) \end{aligned} \tag{A11}$$

where δ represents the weight, ranging in (0.4, 0.6) for generalization. The intensity of grayscale deviation in left-right direction and top-down direction can be different among various screens. For example, TN screens suffer the most serious light leakage in the top-down direction, while IPS screens perform equally well in top-down and left-right directions with similar variance trends. With the random selection for δ, G_{mid}, and the intensity factor of the entire GDF, the trained model can resist distortion from different kinds of screens as shown in Table X.

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2024.3415415

In fact, it is hard to quantitatively formulate the exact same distortions within screen-shooting. Introducing the most influenced parts of distortions is enough to gain strong robustness [9]. During the training, we randomly choose α and β in every step with a proper range of $(-80°, 80°)$. It improves the generalization of the model at different angles, to achieve robustness to multiple angles with a single model. In fact, the chosen range of $(-80°, 80°)$ is different from the real viewing angle of $(-90°, 90°)$. However, the observed area shrinks as the angle grows. In the extreme angle of 90°, the observed area could turn into a line. To obtain steady training, we determine the range of α and β as $(-80°, 80°)$ in training, which is the viewing angle to the center point. The pixel-wise viewing angle $\alpha_{x,y}$ and $\beta_{x,y}$ are calculated by α and β, so that the observed grayscale can be worked out with GDF. We provide some examples of the GDF-generated images in Fig. 6, in which the first two rows represent the raw data and the perspective-calibrated data of shot images. The last row is the simulated images generated by GDF, which are similar to the shot images. In addition, the variation trend of simulated images is shown in Fig. A5.

REFERENCES

[1] O. Evsutin and K. Dzhanashia, "Watermarking schemes for digital images: Robustness overview," *Signal Process. Image Commun.*, vol. 100, pp. 1-22, Jan. 2022.

[2] S. Mellimi, V. Rajput, I. A. Ansari and C. W. Ahn, "A fast and efficient image watermarking scheme based on deep neural network," *Pattern Recognit. Lett.*, vol. 151, pp. 222-228, Nov. 2021.

[3] J. Zhu, R. Kaplan, J. Johnson, and F. Li, "HiDDeN: Hiding data with deep networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Munich, Germany, Sep. 2018, pp. 657-672.

[4] M. Tancik, B. Mildenhall and R. Ng,"StegaStamp: Invisible hyperlinks in physical photographs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Seattle, WA, USA, Jun. 2020, pp. 2114-2123.

[5] W. Sun, J. Zhou, Y. Li, M. Cheung and J. She, "Robust high-capacity watermarking over online social network shared images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 3, pp. 1208-1221, Mar. 2021.

[6] C. Yu, "Attention based data hiding with generative adversarial networks," in *Proc. AAAI*, New York, NY, USA, Apr. 2020, pp. 1120-1128.

[7] J. Jia, Z. Gao, K. Chen, M. Hu, X. Min, G. Zhai and X. Yang, "RIHOOP: Robust invisible hyperlinks in offline and online photographs," *IEEE Trans. Cybern.*, vol. 52, no. 7, pp. 1-13, Dec. 2021.

[8] H. Fang, W. Zhang, H. Zhou, H. Cui and N. Yu, "Screen-shooting resilient watermarking," *IEEE Trans.Inform.Forensic Secur.*, vol. 14, no. 6, pp. 1403-1418, Jun. 2019.

[9] H. Fang, Z. Jia, Z. Ma, E. Chang and W. Zhang, "PIMoG: An effective screen-shooting noise-layer simulation for deep-learning-based watermarking network," in *Proc. ACM Int. Conf. Multimedia*, Oct. 2022, pp. 2267-2275.

[10] N. S. Narawade, "Robust reversible watermarking using integration of histogram shifting and patchwork algorithm to improve quality and capacity," in *Proc. Int. Conf. Inform. Process.*, Dec. 2015, pp. 625-630.

[11] W. Kim, S. Lee and Y. Seo, "Image fingerprinting scheme for print-and-capture model," in *Proc. Pacific-Rim Conf. Multimedia (PCM)*, Hangzhou, China, Nov. 2006, 106-113

[12] C. Xiao, C. Zhang and C. Zheng, "FontCode: Embedding information in text documents using glyph perturbation", *ACM Trans. Graph.*, vol. 37, no.2, pp. 1-16, Apr. 2018.

[13] T. Li, C. An, X. Xiao, A. T. Campbell and X. Zhou, "Real-time screen-camera communication behind any scene," in *Proc. Annual Int. Conf. Mobile Syst. Appl. Services*, Florence, Italy, May. 2015, pp. 197-211.

[14] J. Fu, B. Zhu, W. Cui, S. Ge, Y. Wang, H. Zhang, H. Huang, Y. Tang, D. Zhang and X. Ma, "Chartem: Reviving chart images with data embedding," *IEEE Trans. Vis. Comput. Graph.*, vol.27, no.2, pp. 337-346, Feb. 2021.

[15] C. Hui, S. Liu, W. Cui, J. Zeng, F. Jiang and D. Zhao, "Adaptive flexible 3D histogram watermarking," in *Proc. IEEE Int. Conf. Multimedia Expo*, Shenzhen, China, Jul. 2021, pp. 1-6.

[16] W. Chen, N. Ren, C. Zhu, Q. Zhou, T. Seppänen and A. Keskinarkaus, "Screen-cam robust image watermarking with feature-based synchronization," *Applied Sciences*, vol. 10, no.21, pp. 7494, Oct. 2020.

[17] S. Hamid Amiri, M. Jamzad, "Robust watermarking against print and scan attack through efficient modeling algorithm," in *Signal Process.: Image Commun.*, vol. 29, no. 10, pp. 1181-1196, Nov. 2014.

[18] Y. Huang, B. Niu, H. Guan and S. Zhang, "Enhancing Image Watermarking With Adaptive Embedding Parameter and PSNR Guarantee," in *IEEE Trans. Multimedia*, vol. 21, no. 10, pp. 2447-2460, Oct. 2019.

[19] K. Solanki, U. Madhow, B. S. Manjunath, S. Chandrasekaran and I. El-Khalil, "Print and scan resilient data hiding in images," *IEEE Trans. Inf. Forensics Security*, vol. 1, no. 4, pp. 464-478, Dec. 2006.

[20] B. Chen, Y. Wu, G. Coatrieux, X. Chen and Y. Zheng, "JSNet: A simulation network of JPEG lossy compression and restoration for robust image watermarking against JPEG attack," *Comput. Vis. Image Understand.*, vol. 197, pp. 103015, Aug. 2020.

[21] Y. Liu, M. Guo, J. Zhang, Y. Zhu and X. Xie, "A novel two-stage separable deep learning framework for practical blind watermarking," in *Proc. ACM Int. Conf. Multimedia*, Nice, France, Oct. 2019, pp. 1509-1517.

[22] M. Ahmadi, A. Norouzi, N. Karimi, S. Samavi and A. Emami, "ReDMark: Framework for residual diffusion watermarking based on deep networks," *Expert Syst. Appl.*, vol. 146, pp. 113157, May. 2020.

[23] H. Fang, Z. Jia, Y. Qiu, J. Zhang, W. Zhang and E. C. Chang, "De-END: Decoder-driven watermarking network," *IEEE Trans. Multimedia*, vol. 25, pp. 7571-7581, 2023.

[24] X. Luo, R. Zhan, H. Chang, F. Yang and P. Milanfar, "Distortion agnostic deep watermarking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Seattle, WA, USA, Jun. 2020, pp. 13545-13554.

[25] E. Wengrowski and K. Dana, "Light field messaging with deep photographic steganography," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Long Beach, CA, USA, Jun. 2019, pp. 1515-1524.

[26] C. Qin, X. Li, Z. Zhang, F. Li, X. Zhang and G. Feng, "Print-camera resistant image watermarking with deep noise simulation and constrained learning," *IEEE Trans. Multimedia*, vol. 26, pp. 2164-2177, 2024.

[27] X. Zhong, P. Huang, S. Mastorakis and F. Y. Shih, "An automated and robust image watermarking scheme based on deep neural networks," *IEEE Trans. Multimedia*, vol. 23, pp. 1951-1961, Jul. 2021.

[28] L C. Chen, G. Papandreou, F. Schroff and H. Adam, "Rethinking atrous convolution for semantic image segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Hawaii, USA, Jun. 2017.

[29] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. Int. Conf. Machine Learning (ICML)*, Sydney, Australia, Jul. 2017, pp. 214-223.

[30] R. Zhang, R. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Salt Lake City, USA, Jun. 2018, pp. 586-595.

[31] C. H. Park, S. H. Lee, J. Jeong, K. J. Kim and H. C. Choi, "Control of gray scale inversion in a film compensated twisted nematic liquid crystal display using beam steering optical film," *Appl. Phys. Lett.*, vol. 89, no. 10, pp. 101119, Sep. 2006.

[32] M. Shibazaki, T. Ishinabe, T. Miyashita, T. Uchida and T. Sunata, "New concept of the wide viewing angle liquid crystal display without grayscale inversion: optics and quantum electronics," *Jpn. Journal Appl. Phys.*, vol. 40, no. 12B, pp. L1373, Dec. 2001.

[33] R. Shin and D. Song, "JPEG-resistant adversarial images," in *Proc. NeurIPS Workshop*, Long Beach, California, USA, Dec. 2017, pp. 1-6.

[34] M. J. Huiskes and M. S. Lew, "The mir flickr retrieval evaluation," in *Proc. Int. Conf. Multimedia Inf. Retrieval*, Vancouver, British Columbia, Canada, Oct. 2008, pp. 39-43.

[35] C. Kuo and S. Cheng, "Fusion of color edge detection and color quantization for color image watermarking using principal axes analysis," *Pattern Recognition*, vol. 40, no. 12, pp. 3691-3704, Dec. 2007.

[36] X. Zhu, Z. Ge and S. Wu, "Analytical solutions for uniaxial-film-compensated wide-view liquid crystal displays," *Journal of Display Technol.*, vol. 2, no. 1, pp. 2-20, Mar. 2006.

[37] H. Chen, J. He and S. Wu, "Recent Advances on Quantum-Dot-Enhanced Liquid-Crystal Displays," *IEEE Journal Selected Topics Quant. Electron.*, vol. 23, no. 5, pp. 1-11, Sep. 2017.

[38] Z. Luo, D. Xu and S. Wu, "Emerging Quantum-Dots-Enhanced LCDs," *Journal Display Technol.*, vol. 10, no. 7, pp. 526-539, Jul. 2014.

[39] X. Soria, E. Riba and A. Sappa, "Dense Extreme Inception Network: Towards a Robust CNN Model for Edge Detection," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Snowmass, CO, USA, Mar. 2020, pp. 1912-1921.

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on June 24,2024 at 01:21:55 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.