Camera-Shooting Resilient Watermarking on Image Instance Level

Mingjin He, Bingwen Feng, Yizhi Guo, Jian Weng, *Senior Member, IEEE*, and Wei Lu , *Member, IEEE*

***Abstract*— Capturing displayed images using portable cameras has become familiar among multimedia pirates, necessitating the urgent requirement of camera-shooting resilient watermarking schemes. In this paper, we consider the stealers who only record parts of images, and propose a robust watermarking scheme at the image instance level. This scheme consists of an encoding end, a noise layer, and a decoding end. The encoding end first selects specific watermarking regions associated with segmented image instances. Afterwards, an encoder is employed to embed watermark sequences into the RGB color model of these watermarking regions. At last, templates are embedded to product the final watermarked images. Specifically, our suggested template-based resynchronization comprises a template embedding module at the encoding end and a geometric correction module at the decoding end. The former embeds templates by a correlation-aware multiplicative spread spectrum with an adaptive amplitude, while the latter learns a calibrator to estimate the perspective projection. Experiments on both simulation and real-world scenarios support that the proposed scheme effectively resists camera-shooting attacks with various shooting conditions, regardless of whether the entire displayed images have been captured.**

***Index Terms*— Robust watermarking, camera-shooting attack, image instance, template, resynchronization.**

I. Introduction

THE privacy of multimedia data has attracted more and more attention. It calls for that the content of multimedia data should not be transmitted or distributed illegally. Digital watermarking is a promising technique to track down multimedia pirates. Malicious users may use removable storage devices, email, etc., to steal images. In this case, traditional robust watermarking would be powerful to trace the leak source. It can be designed to resist common image processing [1], [2], [3] or geometric distortions [4], [5], [6] so that the embedded watermark in the stolen images cannot be erased. However, advanced stealers may use portable cameras, e.g., smart phones, to capture displayed multimedia content without leaving any traces. This attack type, commonly referred to as camera-shooting attacks, is demonstrated in Fig. 1. Due to their complexity, simulating these attacks is challenging, making them difficult to effectively address using traditional robust watermarking techniques. Designing a camera-shooting resilient watermarking scheme remains an open problem.

(a) (b)

Fig. 1. Demonstration of camera-shooting attacks. In a) the attacker captures the entire image, while in b) he only captures a part of the image he is interested in.

Typically, robustness against camera-shooting attacks could be achieved by embedding region selection and template/pattern-based synchronization. Fang et al. [7] design intensity-based SIFT to locate embedding regions, and use small-size templates to synchronize watermark sequences. In a subsequent study [8], they design an autocorrelated watermark pattern for precise localization of watermarks in underpainting documents. Alternatively, watermark sequences can be encoded into rotationally orthogonal sinusoidal patterns [9] or directed periodic patterns [10] to facilitate synchronization. In [11], watermark regions are located by enhanced keypoints. An optimization framework is further suggested to unify keypoint enhancement and watermark embedding. Mareen et al. [12] explore the secondary watermark signal created during the compression of watermarked videos, which exhibits high resistance to camcording attacks. While these schemes use traditional robust watermarking skills to counter camera-shooting attacks, the processing of camera shooting remains to be too complex to quantify [7], [8]. Further, the watermark extraction may suffer from various attacks in practice, which is difficult to be predicted and handled by handcrafted modules [13].

Manuscript received 15 August 2023; revised 19 November 2023, 26 February 2024, 10 April 2024, and 25 May 2024; accepted 3 June 2024. Date of publication 10 June 2024; date of current version 27 November 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2022YFB3103100; in part by the National Natural Science Foundation of China under Grant 61802145, Grant 62261160653, Grant 62102101, Grant 62122032, and Grant U23B2023; in part by the Natural Science Foundation of Guangdong Province, China, under Grant 2023A1515011348; in part by the State Archives Administration Science and Technology Program Plan of China under Grant 2023-X-028; and in part by the Fundamental Research Funds for the Central Universities. This article was recommended by Associate Editor J. Gui. *(Corresponding author: Bingwen Feng.)*

Mingjin He, Bingwen Feng, Yizhi Guo, and Jian Weng are with the College of Cyber Security, Jinan University, Guangzhou 510632, China (e-mail: silencehem@stu2020.jnu.edu.cn; bingwfeng@gmail.com; gyzjd@stu.jnu.edu.cn; cryptjweng@gmail.com).

Wei Lu is with the School of Computer Science and Engineering, Guangdong Province Key Laboratory of Information Security Technology, Ministry of Education Key Laboratory of Machine Intelligence and Advanced Computing, Sun Yat-sen University, Guangzhou 510006, China (e-mail: luwei3@mail.sysu.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSVT.2024.3411816.

Digital Object Identifier 10.1109/TCSVT.2024.3411816

With the vigorous development of deep learning techniques, many schemes aim to learn robust watermarking with trainable neural networks. A two-stage extraction network is introduced in [14] to realize camera-shooting resilience. In [15], a comprehensive dataset of camera-display paired images is manually collected to learn the corresponding robustness. However, the achieved robustness is device-specific, and degrades significantly when confronted with another device. General camera-shooting robustness can be achieved by designing a noise layer that simulates the camera-shooting distortions [16], [17], [18], [19], [20], [21]. Jia et al. [18] refine the noise layer by dividing the distortions into camera shooting and environmental components, which better fits the actual shooting scenario. Fang et al. [19] suggest that including only the most influenced distortions in the noise layer suffices to reach strong robustness. Cao et al. [20] design dense block feature maps across different scales to replicate photometric and radiometric effects. Different from these embedding methods, TERA [21] superimposes two complementary message templates onto alternately displayed frames, and suggests an attention guided extraction network to meet robustness requirements. These learning-based approaches have achieved impressive performances. However, they primarily treat the camera-shooting attacks shown in Fig. 1(a), where the entire watermarked image is accessible. In reality, stealers may only record the region he is interested in, as shown in Fig. 1(b) Additionally, sometimes it would be difficult to capture an entire image from a vast display-screen. Most of these schemes would fail in such situation. It is because locating and synchronizing watermarking regions in many schemes relies on the visible markers on the watermarked images, which would not be captured by the stealers. Furthermore, the watermark sequences embedded in the entire image could not survive after a large area of cropping.

The key issues in designing a watermarking scheme resistant to both types of camera-shooting attacks shown in Fig. 1 include robustness, resynchronization, awareness of region of interest, high image quality, and adequate capacity. These issues will be delved in Section III. In this paper, we propose a watermarking network on image instances to address these issues. Watermark embedding and extraction in the scheme are only performed in the watermarking regions associated with image instances. The RGB color model is employed for watermark embedding following previous work such as [16], [17], [18], [19], and [21]. Furthermore, a template-based resynchronization is employed to address the perspective projection during camera shooting. The main contributions of this paper are as follows:

1) An instance-level watermarking method is proposed. The selected watermarking region coincides with the region of interest, meanwhile satisfying the area requirement for the watermark embedding. As a result, it can well resist camera-shooting attacks on local region.
2) A template-based resynchronization method is proposed. Templates are carefully embedded with an adaptive amplitude. Furthermore, a learning-based calibrator is employed to estimate the perspective projection. It can accurately resynchronize the watermarking region without the aid of visible markers.
3) A series of experiments on both simulated distortions and real-world scenarios are conducted. Experimental results support the robustness of the proposed method on real-world camera shooting attacks.

II. Related Work: Learning-Based Robust Watermarking

In recent years, learning-based robust watermarking has been widely studied to resist various distortions.

Common signal processing including noising, blurring, JPEG compression, etc., frequently arises during the acquisition and transmission of images. Thus the robustness on this type of distortions is essential. Inserting a noise layer between the encoder and decoder during training has been widely accepted as the most powerful way to improve robustness [16], [17], [18], [19], [20], [21], [22], [23], [24]. Wang et al. [25] further design an adaptor to balance robustness and imperceptibility. Among these distortions, JPEG compression poses a particular challenge due to its non-differentiability. To address this, various schemes have been proposed that utilize differentiable approximations [22], [24], [26], [27]. Zhang et al. [27] suggest a forward attack simulation layer to simulate JPEG compression as well as other non-differentiable or black-box distortions. We adopt this method in our scheme due to its simplicity and outstanding performance.

Geometric distortions incur desynchronization in watermark information, which behave quite differently from common signal processing. To address this, some schemes fit a quadrilateral into the convex hull of captured images and then twist it back to original shape [16], [18], [19], [20]. Jia et al. [28] impose rectangular data matrices as a watermark sequence and suggest a localization network to help estimate and inverse the perspective transformation. However, this locating method appears vulnerable to JPEG compression and motion blur. In the proposed scheme, we exploit the robust template embedding in traditional watermarking, and suggest a template-based resynchronization method to enhance robustness on geometric distortions.

III. Key Issues in Design

As depicted in Fig. 1, a stealer could capture either the entire image or a part he is interested in. To ensure successful watermark extraction in either case, the designed watermarking scheme must possess the following essential characteristics:

A. Robustness on various distortions.

The camera-shooting process involves digital-to-analog and analog-to-digital conversions in complex environments. This incurs various distortions such as noising, blurring, resampling, color and light changing, compression, and so on [16], [18], and [20]. The watermarking scheme has to resist these distortions, as well as their random combinations.

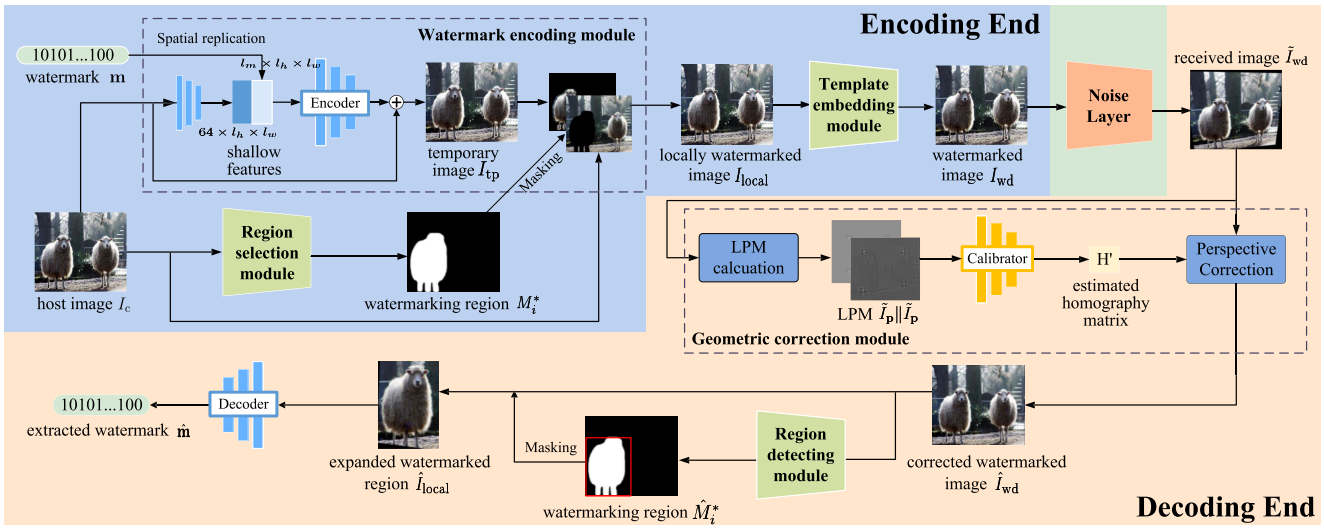


Fig. 2. Pipeline of the proposed method.

B. Automatic Resynchronizing

Different shooting angles, distances, and other factors can introduce geometric distortions like perspective projections and cropping in captured images [7], [8]. These geometric distortions will bring synchronization errors on the embedded watermark sequences. As a result, the watermarking scheme must have the ability to automatically resynchronize watermarking regions.

C. Awareness of Region of Interest (RoI)

If a stealer only captures specific areas of images he is interested in, watermark information in other regions will be excluded. Consequently, it is crucial to concentrate watermark information in the regions of interest to increase its survival rate in captured images.

D. High Quality of Watermarked Image

The quality of watermarked images should be sufficient for normal applications. Furthermore, the imperceptibility of the embedded watermark can also help evade the stealer's attention, otherwise he would skate over the watermarking region.

E. Adequate Capacity

The length of the watermark sequences to be embedded should be sufficient to carry tracking information.

IV. Proposed Method

A learning-based watermarking network is presented herein to fulfill all the aforementioned requirements. Drawing inspiration from work in [16], [17], [18], [19], [20], and [21], we use adversarial learning with a noise layer to obtain robustness on various distortions. Additionally, a template-based geometric correction method is suggested to facilitate automatic resynchronization. To ensure awareness of region of interest, we embed watermark sequences at the instance level. At last, the loss functions are carefully designed to balance imperceptibility and capacity. The framework of the proposed scheme is illustrated in Fig. 2. It can be partitioned into the encoding end, noise layer, and decoding end. Detailed descriptions of these components are provided in subsequent subsections.

A. Encoding End

The encoding end embeds a watermark sequence $\mathbf{m} \in \{0,1\}^{l_m \times 1}$ into a host image $I_c \in [0,1]^{3 \times l_h \times l_w}$ by using the following steps:

1) Given host image I_c, the watermarking regions are first selected by the region selection module. It generates several binary mask $M_i^* \in \{0,1\}^{l_h \times l_w}$, where a value 1 indicates the corresponding pixel is selected for watermarking.
2) The watermark encoding module then embeds $\mathbf{m}$ into the host image guided by M_i^*. It outputs a locally watermarked image $I_{local} \in [0,1]^{3 \times l_h \times l_w}$. where only the selected regions undergo modifications to carry watermark information.
3) At last, the template embedding module embeds templates at prefix points in I_{local} to provide a reference for the watermark resynchronization. It gives the final watermarked image $I_{wd} \in [0,1]^{3 \times l_h \times l_w}$.

The following are the implementation details of each module.

1) Region Selection Module: This module selects watermarking regions that coincide with RoIs. We utilize an off-the-shelf Mask R-CNN [29] to identify and segment RoIs within the image. This method builds upon Faster R-CNN [30] by adding a branch for predicting segmentation masks on each RoI. It is worth noting that while there are numerous instance segmentation methods available, we have chosen a relatively straightforward approach one the sake of convenience.

Mask R-CNN segments the host image to give several candidate masks $M_1, M_2, \cdots$ that corresponded to each RoI. Suppose only the first n_M candidate RoIs are required to be protected, where the value of n_M is determined according to practical applications. Ideally, each of them should carry an entire watermark sequence independently. However, embedding a large watermark sequence into a small region could be

Fig. 3. Demonstration of watermarking region selection.

difficult. As a result, M_i should extend to neighboring regions if it is smaller than a threshold τ_M. This is achieved as follows.

1) Start with the smallest M_i that has not been covered by any watermarking region. Set it as the prime watermarking region $M_i^* = M_i$.
2) Goto Step 3 if the area of M_i satisfies $|M_i| < \tau_M$, otherwise goto Step 6.
3) Find a mask M_j satisfying $M_j = \arg\min_{M'_j \text{ adjacent to } M_i} \left|M'_j\right|$ and update the watermarking region as $M_i^* = M_i^* \cup M_j$.
4) Goto Step 2 if $M_j \neq \varnothing$, otherwise goto Step 5.
5) Find a rectangle R centered at M_i^* that satisfies $|R \cap I_c| = \tau_M$, and update the watermarking region by $M_i^* = R \cap I_c$. Goto Step 6. Note that if M_i lies close to the boundary, not all sections of the rectangle contain image data. Consequently, only the section of the rectangle that contains image data is considered as the effective region.
6) Output M_i^* as an acceptable watermarking region, and goto Step 1 to process the next region, until all the n_M regions have been processed.

Figure 3 demonstrates an example of watermarking region selection. Initially, Mask R-CNN detected three potential regions as candidates. We set $n_M = 3$, indicating that all of them need to be watermarked. The middle instance region is smaller than the threshold, therefore it is merged with the adjacent left one. Similarly, the right instance region is not sufficiently large. Given the absence of any other suitable instance region, it is expanded to encompass a rectangular area. Finally, this refinement process results in two watermarking regions.

Following the above procedure we obtain several watermarking region $M_1^*, M_2^*, \cdots \in \{0,1\}^{l_h \times l_w}$. Each of them is a binary mask, where value 1 indicates the corresponding pixel falls within the region, and 0 otherwise. These masks are then used to guide the watermark embedding.

2) Watermark Encoding Module: The encoding module treats one watermarking region each time. For the i-th region M_i^*, the encoding module first embeds the watermark sequence $\mathbf{m}$ into the host image I_c to produce a temporary watermarked image $I_{tp} \in [0,1]^{3 \times l_h \times l_w}$. Then pixels in I_c covered by M_i^* are replaced with those in I_{tp} to get the instance-level watermarked image $I_{\text{local},i} \in [0,1]^{3 \times l_h \times l_w}$ associated with M_i^*. Repeating the above processing can give a set of $I_{\text{local},1}, I_{\text{local},2}, \cdots$, each of which only has watermark sequences in one watermarking region. Finally, by replacing each M_i^* in I_c with its corresponding $I_{\text{local},i}$, a locally watermarked image I_{local} is obtained, where all the watermarking regions having carried watermark sequences.

We use an encoder with a residual-dense architecture similar to SteganoGAN [31] to generate I_{tp}. It receives I_c and $\mathbf{m}$ as the inputs. Note that, in order to provide adequate resolution for the image instances, the host image is first cropped if its background area is over large.

The encoder begins with extracting shallow features of I_c using a convolution layer. To ensure that the complete watermark sequence is accessible across all latent representations, we spatially replicate $\mathbf{m}$ and concatenate it with the shallow features. This operation enhances robustness against common signal processing and cropping. Then, through several convolutional layers, a residual map with $\mathbf{m}$ having been embedded in it is generated. By combining it with the host image and applying a sigmoid function, we obtain the temporary watermarked image I_{tp}.

In addition, we introduce PatchGAN [32] as the discriminator for adversarial training. This could improve the visual quality of the generated watermarked images.

3) Template Embedding Module: To allow the captured watermarked image having a reference for the watermark resynchronization, templates are embedded at prefixed n_p points around each M_i^* in I_{local}. After that, we obtain the final watermarked image $I_{wd} \in [0,1]^{3 \times l_h \times l_w}$.

The template energy and watermarked image quality should be carefully balanced. We employ correlation-aware multiplicative spread spectrum (CMSS) [33] to embed templates. Firstly, a 7×7 sized pattern T is generated by randomly selecting each element from a binary set $\{-1, +1\}$ with equal probability. This pattern is then embedded into the coefficient blocks beginning at points $\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_{n_p}$. We use the correlation to detect the existence of templates, defined as

$$z = \sum_{i,j} \Big(X[i,j]^2 \times T[i,j] \Big) \tag{1}$$

where X denotes a coefficient block. To guarantee reliable template detection, the template energy is determined such that the absolute correlation $|z|$ calculated by the coefficient block containing the template is ρ times greater than those calculated by the blocks without the template. The template embedding procedure is as follows.

1) Shift the host image so that M_i^* is located at the center of the polygon formed by $\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_{n_p}$.
2) The **cb** channel of Ycbcr color space is employed to embed template, because it is experimentally not sensitive to the modification. Convert the color space of the host image from RGB to Ycbcr, then apply Haar wavelet transform to the **cb** channel of the host image, and use the **LL** subband as the cover $\mathbf{X}$ to embed templates.
3) Use a 7×7 sized window to scan $\mathbf{X}$ in a zigzag order, and measure the correlation between T and each obtained coefficient block X_k. Denote the result as z_k. Then, the correlation threshold of template embedding is defined as:

$$t = \rho \times \max_k(|z_k|) \tag{2}$$

Recall that ρ controls the template energy, and is experimentally set with 2.

4) For the k-th coefficient block X_k, whose top left corner is $\mathbf{p}_o, o = 1, \cdots, n_p$. Firstly, calculate the correlation between T and X_k, saying z_k. Then the pattern T is embedded according to the sign of z_k, formed as:

$$Y_k[i,j] = \begin{cases} X_k[i,j] \times (1 + \alpha_1 T[i,j]) & \texttt{if } z_k \geq 0 \\ X_k[i,j] \times (1 - \alpha_2 T[i,j]) & \texttt{if } z_k < 0 \end{cases} \tag{3}$$

where the amplitudes α_1 and α_2 are set so that

$$\begin{cases} (1+\alpha_1^2) z_k + 2\alpha_1 \sum_{i,j} \left(X_k[i,j]^2 \right) \geq t \\ (1+\alpha_2^2) z_k - 2\alpha_2 \sum_{i,j} \left(X_k[i,j]^2 \right) \leq -t \end{cases} \tag{4}$$

5) Use the **LL** subband that has been embedded with templates to reconstruct the **cb** channel, and convert the color space of the host image to RGB again.
6) Shift the host image to the original location.

B. Noise Layer

We insert a noise layer between the encoding and decoding ends to simulate the perturbations that may incurred by camera shooting. The perturbations and their parameters are set as follows:

- **Scaling**. We randomly scale the watermarked image I_{wd} with a scaling factor in $[0.5, 2]$. After that, the scaled image is resized to its original size.
- **Blurring**. Both defocusing and motion blurring are considered. We use Gaussian blur to filter I_{wd} with kernel size varying in $\{3, 5, 7\}$ and the standard deviation randomly sampled between 1 and 3 for defocusing blurring. To simulate motion blurring, we apply a random angle from 0 to 2π and a motion blur kernel of size ranging from 3 to 7. During training, we randomly apply Gaussian blur, motion blur, and a combination of them for iterative training.
- **Color transformation**. We use the simulation proposed by [28], formed as:

$$\tilde{I}_{wd} = (1.0 + f_{con}) \times (I_{\text{wd}} + f_{col}) + f_{bri} \tag{5}$$

where $\tilde{I}_{wd}$ is the distorted image, f_{con}, f_{col} and f_{bri} are the offsets of contrast, color, and brightness respectively. They are set as $f_{con} \in [-0.3, 0.3]$, $f_{col} \in [-0.1, 0.1]$, and $f_{bri} \in [-0.3, 0.3]$ during training.
- **Noising**. We add either Gaussian noise or random noise to I_{wd} with a probability of 50% for each type of noise. The mean of the Gaussian noise is set to 0, while the standard deviation is uniformly sampled in $[0, 0.02]$. The range of the random noise is set between $[-0.1, 0.1]$.
- **JPEG compression**. we use the differentiable approximation proposed by [27] to simulate JPEG compression. The quality factor is sampled uniformly within $[50, 100]$.
- **Perspective warping**. To sample a homography, we randomly sample the perspective scale within $[0, 0.2]$ to perturb the four corner locations of the image.

After applying all the above perturbations, the pixel values of the distorted image are clipped to $[0, 1]$. These distorted images are used in the adversarial training to gain robustness on the corresponding distortions.

C. Decoding End

The goal of the decoding end is to recover the embedded watermark sequences from the received image $\tilde{I}_{\text{wd}} \in [0,1]^{3\times\tilde{l}_h\times\tilde{l}_w}$ which has suffered from various distortions. Note that sometimes $\tilde{l}_h \neq l_h, \tilde{l}_w \neq l_w$ due to geometric distortions. The decoding procedure is as follows.

1) Use the geometric correction module to geometrically correct $\tilde{I}_{\text{wd}}$ with the aid of prefixed templates, yielding a geometrically corrected image $\hat{I}_{\text{wd}}$.
2) Use the region detecting module to detect and clip each watermarked region $\hat{I}_{\text{local},i}$.
3) The watermark decoding module takes $\hat{I}_{\text{local},i}$ as input and outputs the extracted watermark sequence $\hat{\mathbf{m}}$.

Each module is detailed as follows.

1) Geometric Correction Module: This module first detects the embedded templates in the received image $\tilde{I}_{\text{wd}}$, then uses a calibrator to inverse the perspective projection.

Similarly to the template embedding module, $\tilde{I}_{\text{wd}}$ is first converted to Ycbcr color space. The **LL** subband of Haar wavelet transformed **cb** channel, denoted as $\tilde{\mathbf{X}}$, is then used to detect the embedded templates. It is scanned in a zigzag pattern with a 7×7 sized window to calculate a location probability map (LPM) $\tilde{I}_{\mathbf{p}}$ by using:

$$\tilde{I}_{\mathbf{p}}[i,j] = \left| \tilde{X}[i,j]^2 \times T[i,j] \right| \tag{6}$$

where $\tilde{X}$ is the scanned coefficient block. This LPM reflects the probability of the locations of each template.

In addition to the above LPM, another reference LPM $I_{\mathbf{p}}$ is generated to estimate the homography with the original template locations. This reference LPM is calculated by using an ideal image, one that is free from any geometrical attacks. Further, the detection of its template should remain unaffected by the content of the image itself. Therefore, we embed the template to an image with pixel values all equal to 1 following the method outlined in Section IV-A.3. This constructed image serves as the ideal image to calculate $I_{\mathbf{p}}$. Then both $\tilde{I}_{\mathbf{p}}$ and $I_{\mathbf{p}}$ are fed into a calibrator to estimate the homography between them. Subsequently, we inverse the perspective projection with the estimated homography, yielding a geometrically corrected image $\hat{I}_{\text{wd}}$.

We retrain the geometric correction network suggested in DHN [34] as the calibrator. It accepts the concatenated $I_{\mathbf{p}} \| \tilde{I}_{\mathbf{p}}$ as input and learns to estimate the offsets between the detected templates. Its training strategy is stated in Section IV-D.2. We have tried other methods such as spatial transformer network (STN) for perspective projection correction on the received image. However, these attempts have not been successful due to the diversity of image contents.

2) Region Detecting Module: This module is similar to the region selection module described in IV-A.1. Upon acquiring multiple watermarking regions $\hat{M}_1^*, \hat{M}_2^*, \cdots$, each of them

is expanded to the smallest rectangle region that covers it. These expanded regions subsequently serve as the input $\hat{I}_{\text{local},1}, \hat{I}_{\text{local},2}, \cdots$ for the subsequent watermark decoding module.

3) Watermark Decoding Module: Each time the watermark decoding module inputs one $\hat{I}_{\text{local},i}$ into the decoder to extract the embedded watermark sequence. Note that some regions may be not used to carry watermark information. As a result, only those regions with extraction accuracies larger than 60% are designated as watermarked.

In order to focus more attention on regions where features can be accurately extracted, we adopt an attention-guided extraction network similar to that in [21] as the decoder. Additionally, a global average pooling layer and two fully connected layers are incorporated near the output, enabling the decoder to accommodate input images of different sizes.

D. Training Strategy and Loss Function

There are four networks that require training, an encoder, a decoder, the corresponding discriminator, and a calibrator. We do not train them simultaneously due to their competing goals. Instead, the calibrator is trained separately from the other networks.

1) Training Strategy of Encoder, Decoder, and Discriminator: The perturbations described in Section IV-B are imposed sequentially during training. Experimentally we find that the size of M_i^* has a substantial impact on the network's convergence. To alleviate the variance in size among various images, the rectangular convex hulls enclosing the watermarked regions within the same batch are resized to a unified size $l_s \times l_s$. Then they are concatenated together to form a new tensor, which is subsequently fed into the decoder for training. The encoder, the decoder, and the discriminator are trained via the following stages.

a) Stage 1. optimizing decoder: We only train the decoder first to gain the watermark extraction ability. In this stage, binary cross entropy (BCE) between $\mathbf{m}$ and $\hat{\mathbf{m}}$ is employed as the sole loss L_{sec} to optimize the decoder.

b) Stage 2. optimizing encoder, decoder, and discriminator jointly: Both the encoder and the discriminator are taken into account in this stage. In order to guarantee the visual quality of watermarked images, multiple loss functions related to image quality are introduced: the perceptual loss L_{vgg} defined in [35], the LPIPS perceptual loss L_{lpips} defined in [36], the L2 loss L_2, and the SSIM loss L_{ssim} defined in [37]. Taken altogether, the total loss for this stage is formed as

$$L_t = \lambda_1 L_{vgg} + \lambda_2 L_{lpips} + \lambda_3 L_2 + \lambda_4 L_{ssim} + \lambda_5 L_{sec} + \lambda_6 L_{dis} \tag{7}$$

where L_{dis} is the least squared adversarial loss suggested in [38], and λ_1, λ_2, λ_3, λ_4, λ_5, and λ_6 are the loss weight factors.

It is noteworthy that managing the visual quality of watermarked images can be challenging due to the intricate nature of the imposed perturbations. Consequently, it is crucial to integrate various loss functions related to visual quality. Although there may be other effective combinations of loss functions, experimental outcomes demonstrate that the utilized combination is effective for the training process.

2) Training Strategy of Calibrator: The calibrator outputs $2 \times n_p$ real-valued offsets between the detected templates in $\tilde{I}_{\mathbf{p}}$ and those in $I_{\mathbf{p}}$. Therefore, we employ an L2 loss between the real and estimated offsets as the geometrical correction loss L_{geo}. Furthermore, to ensure the accuracy of the calibrator in real-world scenarios, the noise layer described in Section IV-B is incorporated into the training process.

V. Experimental Results and Analysis

A. Experimental Settings

1) Dataset and Hyper-Parameter Setting: We randomly select 30,000 images from the MS-COCO dataset [39] to constitute our training dataset, and randomly select 100 images from the rest for the evaluation. All the selected images are resized to $3 \times 400 \times 400$, that is, $l_h = 400$ and $l_w = 400$. The area threshold of the watermarking region is set as $\tau_M = l_h \times l_w/8$. It should be noted that the length of watermark sequences should not be too large, considering that the watermark region is typically smaller than half of the host image. Consequently, two values are established in the experiments: $l_m = 50$ and $l_m = 100$. Experimental results indicate that the network with longer watermark sequences encounters difficulties in converging. Nevertheless, this length of watermark sequences remains useful for indicating copyright ownership [12], [40], or labeling the output device. For each experiment, the test watermark sequence is randomly sampled from the Bernoulli distribution $Ber(0.5)$.

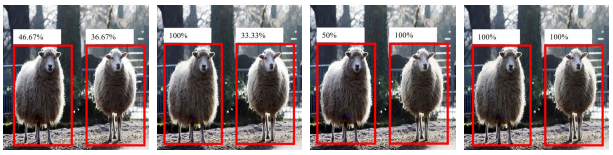
The experiments are conducted on a GeForce RTX 3090 24G GPU in the environment of Python 3.8.8 and Pytorch 1.10.1+cu111. During training, the hyper-parameters are set as $\lambda_1 = 1$, $\lambda_2 = 1.5$, $\lambda_3 = 5$, $\lambda_4 = 0.5$, $\lambda_5 = 2$, $\lambda_6 = 0.005$. The Adam optimizer [41] with hyper-parameters $\beta_1 = 0.5$ and $\beta_2 = 0.999$ is employed for gradient descent. The learning rate is set to 2×10^{-4} for the baseline network and 1×10^{-3} for the adversarial network. All models are trained with a batch size of 10. Additionally, we set $n_M = 2$, $l_s = 256$, and $n_p = 4$.

2) Evaluation Metrics: We use the bit error ratio (BER) to measure robustness. Specifically, it is the ratio $\sum |\mathbf{m}[i] - \hat{\mathbf{m}}[i]|/l_m$. Consequently, the watermark extraction accuracy (ACC) can be defined as: $\text{ACC} = (1 - \text{BER}) \times 100\%$.

For evaluating visual quality, PSNR, SSIM [37], and LPIPS [36] are adopted as the metrics. In addition, we use Average Corner Error (ACE) to assess the performance of the geometric correction module. It is defined as the L_{geo} computed within a single image.

B. Reliability Evaluation

We first assess the reliability of the proposed scheme in accurately distinguishing between watermarked and non-watermarked images. Since the proposed scheme focuses on watermarking only the selected watermarking regions, rather than the entire image, images that contain at least one watermarked region are identified as watermarked. In the process of watermark extraction, only regions with extraction accuracies



(a) (b) (c) (d)

Fig. 4. Demonstration of the accuracy of watermark extraction when embedding watermarks in different regions. In (a), neither of the image instances was used for watermark embedding. In (b), only the left instance was used, in (c) only the right instance was used, and in (d) both instances were employed.

TABLE I

TEST ON DISTINGUISHING BETWEEN WATERMARKED AND NON-WATERMARKED IMAGES

	Extraction accuracy	Watermarked regions (identified/total)	Watermarked images (identified/total)
host images	49%	0/0	0/0
images with one watermarked regions	100%	10/10	10/10
images with two watermarked regions	100%	20/20	10/10

TABLE II

IMAGE QUALITY COMPARISON AVERAGED OVER THE EVALUATION IMAGES

Method	$l_m = 50$ PSNR↑	SSIM↑	LPIPS↓	$l_m = 100$ PSNR↑	SSIM↑	LPIPS↓
Stegastamp	30.12	0.954	0.093	28.40	0.935	0.075
RIHOOP	**35.90**	0.971	**0.013**	28.72	0.937	0.017
Ours	35.47	**0.978**	0.016	**29.19**	**0.942**	**0.018**

exceeding 60% are considered as watermarked. Therefore, the determination of whether an image is watermarked hinges solely on whether regions with extraction accuracies exceeding 60% can be identified.

We select 10 test images, each containing multiple potential watermarking regions, from the test image set. These images are evaluated by successively using 0, 1, and 2 potential watermarking regions each time. Specifically, a pseudorandom binary watermark sequence of length 50 was embedded into each test image. In the first scenario, neither instance was used for watermark embedding. In the second scenario, one instance was employed, while in the third scenario, both instances were used. Figure 4 illustrates the extraction accuracies of a test image across these scenarios. It can be observed that the proposed scheme can accurately identify the watermarking regions. The averaged experimental results are presented in Table I. It demonstrates that our proposed scheme can effectively distinguish between watermarked and non-watermarked images.

C. Image Quality Evaluation

1) Visual Quality Test: We first assess the visual quality of watermarked images. Figure 6 shows the differences in the RGB channel between the original and watermarked images. It can be observed that watermarking energy appears to be evenly spread across all three channels. To comprehensively assess the effectiveness of our approach, we compare it with two state-of-the-art learning-based approaches, Stegastamp [16] and RIHOOP [18]. These methods are specifically designed to resist camera-shooting attacks. The watermark sequences' lengths in them are set as same as those in our scheme. Furthermore, they are retrained on our training set for the sake of fair comparison.

TABLE III

TEST ON CHANGES IN IMAGE FILE SIZE (KB) CAUSED BY WATERMARK EMBEDDING

	"Flower"	"Zebra"	"Elephant"	"Train"	Averaged
Host	106	168	147	178	149.75
Watermarked	112	169	155	185	155.25

TABLE IV

PARAMETERS USED IN EACH TYPE OF PERTURBATIONS

parameter \ strength		1	2	3
Scaling	factor	0.5	1.25	2
Defocusing blurring	ks σ	3 1	5 2	7 3
Motion blurring	ks σ angle	3 1 $[0°, 360°]$	5 2 $[0°, 360°]$	7 3 $[0°, 360°]$
Color Transform	f_{con} f_{col} f_{bri}	$[-0.1, 0.1]$ $[-0.03, 0.03]$ $[-0.1, 0.1]$	$[-0.2, 0.2]$ $[-0.06, 0.06]$ $[-0.2, 0.2]$	$[-0.3, 0.3]$ $[-0.1, 0.1]$ $[-0.3, 0.3]$
Gaussian noise	μ σ	0 $[0, 0.007]$	0 $[0, 0.014]$	0 $[0, 0.020]$
Random noise	factor	$[-0.033, 0.033]$	$[-0.066, 0.066]$	$[-0.1, 0.1]$
JPEG	QF	90	70	50
Perspective	str.	$[0, 0.07]$	$[0, 0.14]$	$[0, 0.20]$
Cropping	radio	0.9	0.6	0.3
Combined		combination of all the above		

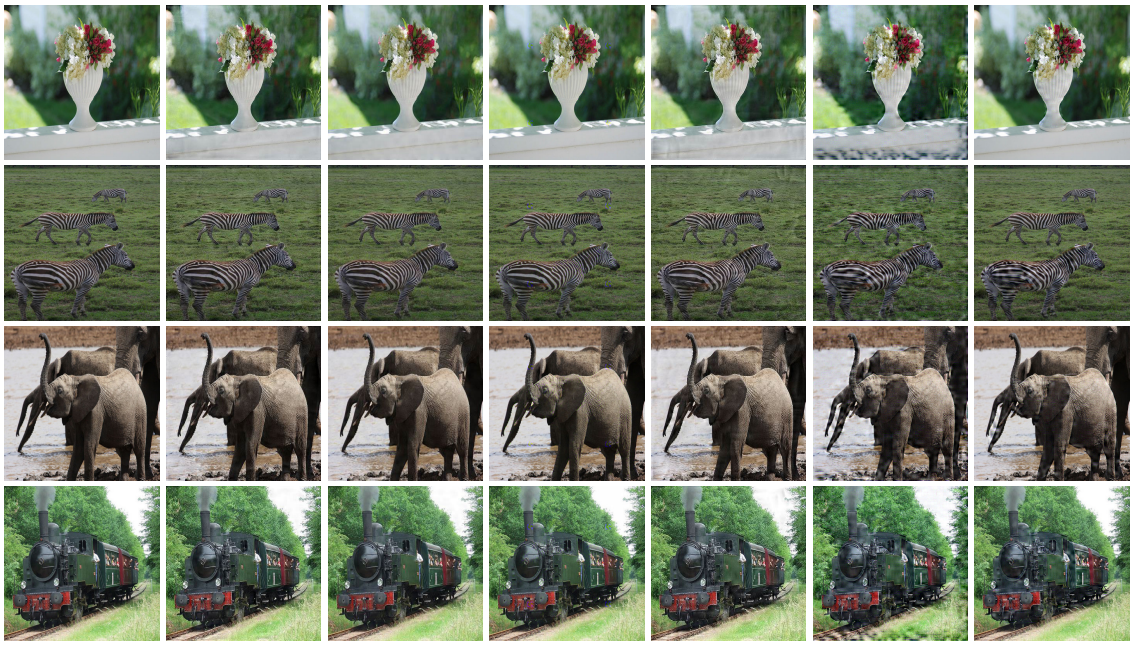
Figure 5 illustrates some watermarked images generated by different schemes. It can be observed that the watermarked images generated by our scheme closely resemble the host images. This may be because, compared with the RoIs, the backgrounds in images are generally smoother, which is not suitable for embedding watermark sequences. Therefore, only using RoIs allows the proposed scheme to reach better visual quality. Additionally, Table II compares the averaged PSNR, SSIM, and LPIPS values of the watermarked images generated by different schemes. It confirms the high visual quality of our approach.

2) File Size Test: Next, The impact of watermark embedding on the size of image files is evaluated. Table III presents the experimental results on the test images depicted in Fig. 5, as well as the results averaged over all the test images. It can be observed that the proposed scheme only marginally alters the image size, thereby avoiding a substantial escalation in storage or transmission burden.

D. Robustness Evaluation

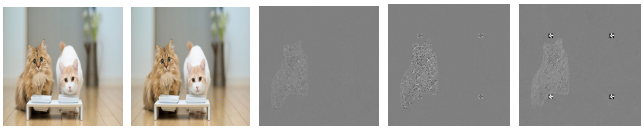
In this section, we conduct experiments in both simulation and real-world scenarios to verify the robustness of the proposed scheme.

1) Simulation-Based Robustness Test: We first evaluate the robustness on synthetic perturbations. Besides the perturbation described in Section IV-B, cropping attacks are also



(a) (b) (c) (d) (e) (f) (g)

Fig. 5. Demonstration of host and watermarked images. The images in (a) are the host images, in (b), (c), and (d) are the watermarked images generated by Stegastamp, RIHOOP, and the proposed scheme, respectively, with $l_m = 50$, and in (e), (f), and (g) are the watermarked images generated by Stegastamp, RIHOOP, and the proposed scheme, respectively, with $l_m = 100$.



(a) (b) (c) (d) (e)

Fig. 6. RGB channel difference between the original and watermarked images. (a) is the original image, (b) is the watermarked image, (c), (d), and (e) are the differences between (a) and (b) on the R, G, and B channels, respectively.

included because they usually occur during camera shooting. The strength of each type of perturbation is parameterized from 0 (weakest) to 3 (strongest). Table IV outlines the specific parameters employed for each perturbation. These effects are visually demonstrated in Fig. 7. We compare the accuracy of watermark extraction between our scheme and two methods, Stegastamp and RIHOOP, under these perturbations. The comparison results are depicted in Fig. 8. It can be observed all the schemes can well resist common signal processing due to the elaborate noise layer. However, their performance begins to deteriorate when facing geometric attacks such as perspective warping, cropping, and combined perturbations. Nevertheless, the proposed scheme presents steadier robustness on these geometric attacks than the compared schemes. It may be attributed to the spatially concentrated watermark information in the proposed scheme, which results in reduced susceptibility to geometric attacks compared to other schemes. Given that camera shooting often introduces such geometric distortions, it implies that our scheme will offer superior robustness against real-world camera-shooting attacks.

2) In-the-Wild Robustness Test: Real-world camera shooting involves various shooting conditions. To assess the effectiveness of our proposed method across a range of real-world scenarios, we captured numerous photos under different shooting angles, distances, and light conditions. Note that only parts of watermarked images that contain regions of interest can be captured when the shooting distances are too close, which corresponds to the scenario shown in Fig. 1(b). Furthermore, we intentionally refrained from utilizing error correction codes in all the schemes in order to only evaluate the effectiveness of different models.

In the experiments, we utilized a Lenovo LT2223wA monitor and a DELL E2417H monitor as our primary display monitors. Photographs were captured using three camera-equipped mobile devices: an 8-megapixel iPhone 6, a 12-megapixel iPhone X, and a 13-megapixel Lenovo Legion Tab y700. Subsequently, watermarked images are roughly cropped out from the captured photos and resized to fit different schemes.

a) Robustness under different shooting angles: We use a bracket to fix the mobile device (30 cm from the monitor) and adjust the orientation of the monitor accordingly. Then, the shooting distance is adjusted to 8 cm, which causes that only parts of images can be captured.

Figure 9 demonstrates several captured images as well as the detection results. They show that the embedded watermark can be successfully detected. We further list the averaged watermark extraction accuracies of different methods in Table V. It can be seen that the proposed scheme exhibits comparable accuracy to RIHOOP and outperforms Stegastamp when the shooting distance is 30 cm. Since RIHOOP introduces a more realistic shooting model, it performs slightly better than the proposed scheme. However, when the shooting distance is reduced to 8 cm, our method outperforms the compared ones by a large margin. The scores obtained by Stegastamp and RIHOOP are basically below 60%. In contrast, the proposed scheme is sufficiently robust to different shooting angles whatever the entire image is captured. This resilience can be attributed to the follows. Given that RoIs in an image

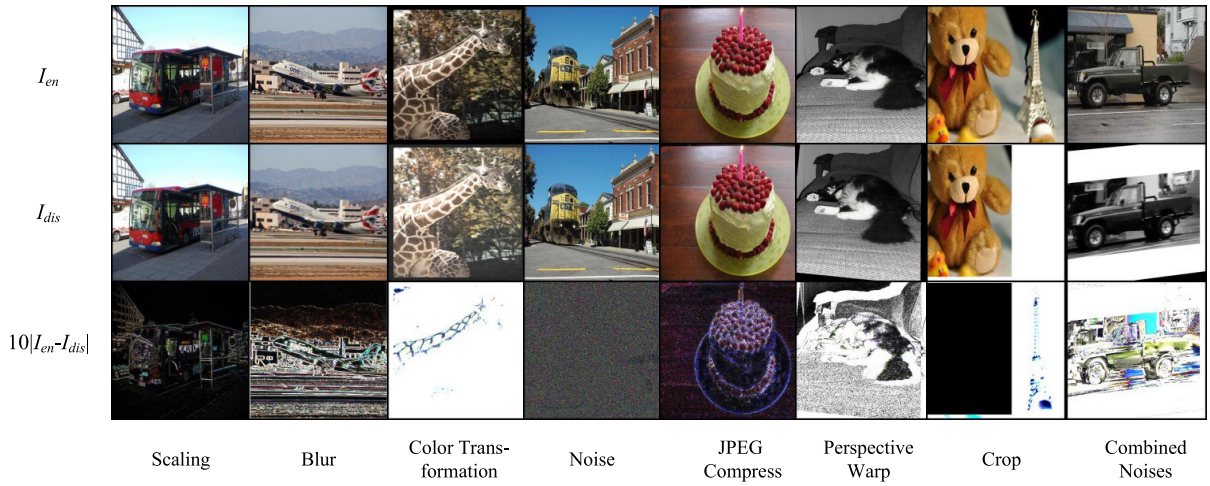


Fig. 7. Demonstration of different perturbations. Top: watermarked image I_{wd}. Middle: distorted image $\tilde{I}_{\text{wd}}$. Bottom: magnified difference $|I_{\text{wd}} - \tilde{I}_{\text{wd}}|$. The employed perturbations in the figures are: scaling (scaling factor= 2), blurring (Gaussian blur: kernel size = 7, motion blur: kernel size = 7, angle = 35°), color transformation ($f_{con} = 0.3$, $f_{col} = 0.1$, $f_{bri} = 0.3$), noising (random noise: range in $[-0.1, 0.1]$, Gaussian noise: $\mu = 0, \sigma = 0.02$), JPEG compression (QF = 50), perspective distortion (distortion strength = 0.2), and cropping (cropping ratio= 60%).

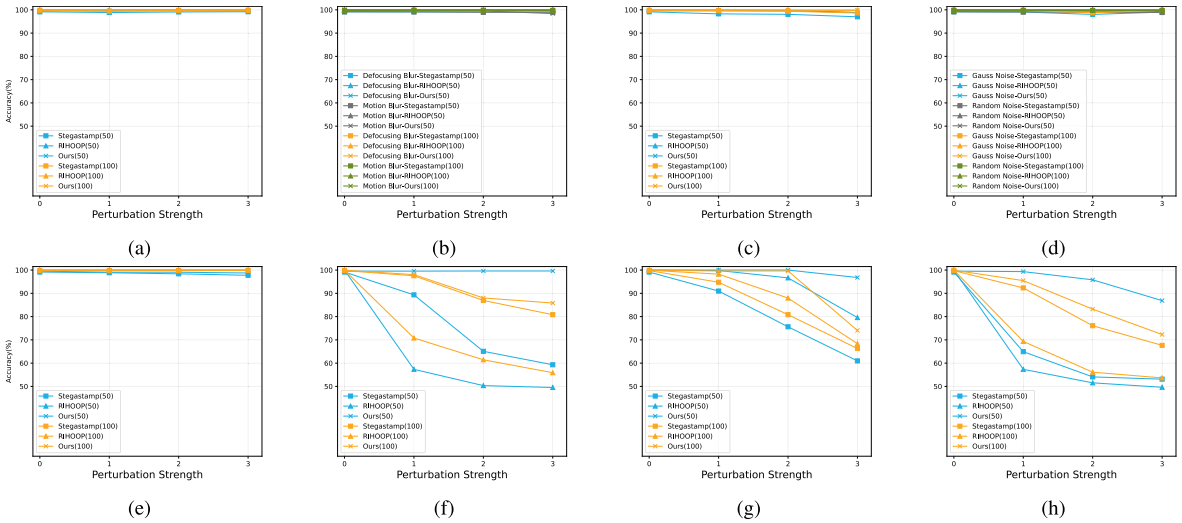


Fig. 8. Watermark extraction accuracy under (a) scaling, (b) blurring, (c) color transformation, (d) noising, (e) JPEG compression, (f) perspective warping, (g) cropping, and (h) combined perturbations.

TABLE V
WATERMARK EXTRACTION ACCURACY UNDER CAMERA-SHOOTING WITH DIFFERENT HORIZONTAL ANGLES

Method \ Angle		Left 60°	Left 30°	Left 10°	0°	Right 10°	Right 30°	Right 60°
Stegastamp ($l_m = 50$)	30 cm	95.60%	97.05%	97.25%	99.45%	97.40%	96.30%	98.10%
RIHOOP ($l_m = 50$)	30 cm	**99.10%**	99.35%	**99.75%**	99.85%	**99.75%**	99.65%	**99.70%**
Ours ($l_m = 50$)	30 cm	98.25%	**99.65%**	99.50%	**99.88%**	99.55%	**99.70%**	98.60%
Stegastamp ($l_m = 50$)	8 cm	52.18%	53.45%	56.95%	56.43%	55.55%	53.78%	54.35%
RIHOOP ($l_m = 50$)	8 cm	53.55%	51.68%	50.68%	52.78%	50.35%	51.90%	50.08%
Ours ($l_m = 50$)	8 cm	**98.55%**	**99.23%**	**98.68%**	**99.68%**	**99.38%**	**99.23%**	**98.60%**
Stegastamp ($l_m = 100$)	30 cm	98.00%	98.10%	98.23%	99.65%	99.00%	99.00%	98.65%
RIHOOP ($l_m = 100$)	30 cm	**99.38%**	99.15%	99.45%	**99.75%**	99.55%	**99.53%**	**99.70%**
Ours ($l_m = 100$)	30 cm	99.25%	**99.35%**	**99.53%**	**99.75%**	**99.63%**	99.40%	99.40%
Stegastamp ($l_m = 100$)	8 cm	48.00%	45.50%	48.35%	59.36%	45.00%	41.78%	48.50%
RIHOOP ($l_m = 100$)	8 cm	50.50%	50.10%	56.50%	45.82%	54.50%	48.00%	49.50%
Ours ($l_m = 100$)	8 cm	**98.45%**	**99.20%**	**99.15%**	**99.33%**	**99.20%**	**98.98%**	**98.35%**

often occupy only a portion of the image and are typically situated away from the periphery, it becomes imperative to exclude a significant portion of the image in order to eliminate specific segments of the watermarking region. When such a large portion of an image is cropped, the compared schemes that distribute watermarks throughout the entire image are likely to lose a considerable amount of embedded watermark information. Furthermore, these alternative schemes rely on image boundaries for resynchronizing the captured watermarked image, but such boundaries are often absent in

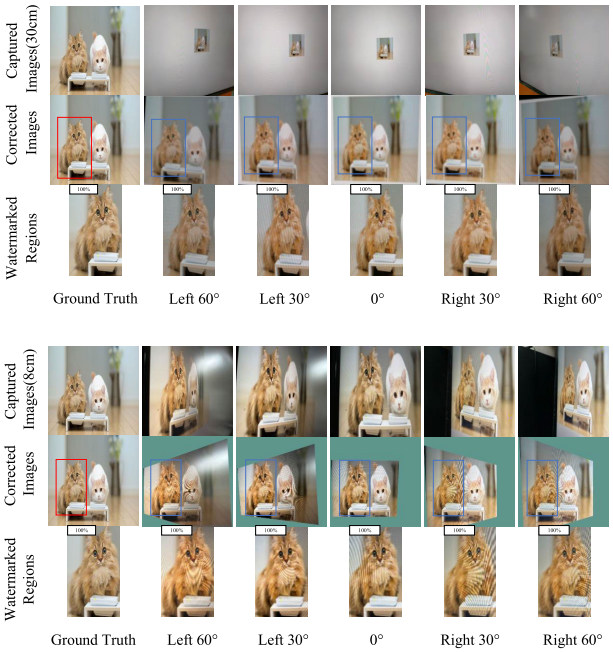


Fig. 9. Demonstration of watermark detection of the proposed scheme under different shooting angles. In the first column are the captured images with shooting distance 30cm. In the fourth column are the captured images with shooting distance 8cm. In the second and fifth columns are their corrected images, and in the third and last column are the detection results.

TABLE VI

WATERMARK EXTRACTION ACCURACY UNDER CAMERA-SHOOTING WITH DIFFERENT DISTANCES

Method \ Distance	10 cm	20 cm	30 cm	40 cm	50 cm
Stegastamp ($l_m = 50$)	54.35%	90.75%	99.45%	99.85%	99.03%
RIHOOP ($l_m = 50$)	52.78%	90.85%	99.85%	**99.93%**	**99.43%**
Ours ($l_m = 50$)	**99.13%**	**99.68%**	**99.88%**	98.75%	96.13%
Stegastamp ($l_m = 100$)	48.50%	98.50%	99.65%	99.45%	99.50%
RIHOOP ($l_m = 100$)	44.75%	98.98%	**99.75%**	99.50%	**99.65%**
Ours ($l_m = 100$)	**99.00%**	**99.60%**	**99.75%**	**99.60%**	99.55%

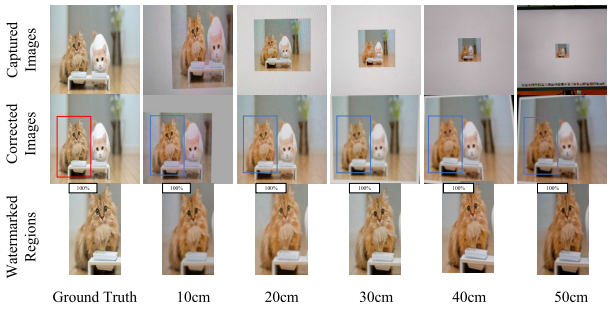
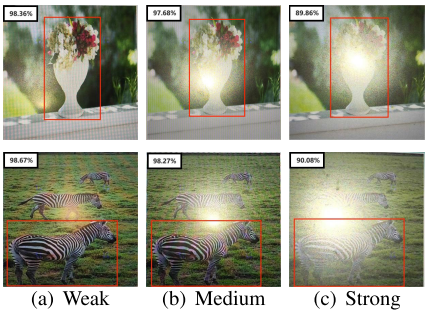


Fig. 10. Demonstration of watermark detection of the proposed scheme under different shooting distances. In the first column are the captured images, in the second column are the corrected images, and in the last column are the detection results.

partially captured images. Consequently, these schemes forfeit their ability of resynchronization. As a result, the proposed approach demonstrates greater robustness against partial image capturing compared to the other schemes.

b) Robustness under different shooting distances: We fix the position of the monitor and vary the distance between the shooting device and the monitor in {10cm, 20cm, 30cm, 40cm, 50cm}. Visualization results shown in Fig. 10 demonstrate the scheme's efficacy across all distances. The averaged watermark extraction results are



(a) Weak (b) Medium (c) Strong

Fig. 11. Demonstration of watermark detection of the proposed scheme under different lighting conditions.

TABLE VII

WATERMARK EXTRACTION ACCURACY UNDER CAMERA-SHOOTING WITH DIFFERENT LIGHTING CONDITIONS

Method \ Angle	Weak	Medium	Strong	Mean
Stegastamp($l_m = 50$)	96.35%	95.75%	84.58%	92.23%
RIHOOP($l_m = 50$)	**99.30%**	**99.58%**	89.65%	**96.18%**
Ours($l_m = 50$)	98.30%	97.15%	**89.75%**	95.07%
Stegastamp($l_m = 100$)	97.30%	95.05%	84.50%	92.28%
RIHOOP($l_m = 100$)	98.18%	**99.50%**	**90.23%**	**95.97%**
Ours($l_m = 100$)	**98.30%**	98.13%	89.55%	95.33%

compared in Table VI. It indicates that the performances of Stegastamp and RIHOOP decrease noticeably in close shots. It may be due to the effect of unconsidered noises, such as moire patterns. On the other hand, the proposed scheme does not perform very well with increasing the shooting distance. It may be because the watermarking energy in the proposed scheme is concentrated in image instances, which is more sensitive to the loss of image details. The compared schemes often fail to extract watermark sequences when only parts of images are available. In contrast, our scheme maintains a high level of accuracy (above 95%). As a result, the proposed scheme is robust to various shooting distances.

c) Robustness under different light conditions: We evaluate the robustness of our scheme against different light intensities. Specifically, half of the images are displayed on a DELL E2417H monitor, and the remaining half are shown on a Lenovo LT2223wA monitor. Subsequently, an external light source is used to illuminate the screen at three different intensities: weak, medium and strong. Figure 11 illustrates the captured images under these light intensities. The experimental results are reported in Table VII. It can be observed that all three schemes exhibit an accuracy exceeding 90% under weak and medium light intensity. However, as the light intensity increases, the accuracy of Stegastamp decreases below 85%, whereas our scheme and RIHOOP can still maintain an accuracy close to 90%. These results indicate the remarkable robustness of our scheme to illumination variations.

d) Variations of different devices: Recognizing the utilization of two types of display monitors and three types of camera-equipped mobile devices in our experiments, we assess the impact of these devices on robustness. We record the monitor and camera used for capturing each photo, and compare the performance of all the combinations. Table VIII details the robustness comparison among these combinations, where the shooting distance is 30 cm and the shooting

TABLE VIII
COMPARISON OF WATERMARK EXTRACTION ACCURACY USING DIFFERENT DEVICES

Method \ Device	DELL E2417H			Lenovo LT2223wA		
	iPhone 6	iPhone X	Legion Tab	iPhone 6	iPhone X	Legion Tab
Stegastamp(0°)	99.70%	100.00%	99.31%	98.53%	100.00%	99.76%
RIHOOP(0°)	99.72%	100.00%	99.67%	99.47%	99.92%	99.74%
Ours(0°)	99.86%	99.83%	99.84%	99.70%	99.85%	99.82%
Stegastamp(30°)	94.07%	98.47%	99.00%	95.13%	99.00%	99.48%
RIHOOP(30°)	98.90%	100.00%	98.89%	98.90%	99.82%	100.00%
Ours(30°)	99.57%	99.66%	99.57%	99.42%	99.54%	99.60%

TABLE IX
WATERMARK EXTRACTION ACCURACY UNDER PRINTING-SHOOTING ATTACKS

Method \ Scene	Indoor (entire)	Outdoor (entire)	Mean (entire)	Indoor (part)	Outdoor (part)	Mean (part)
Stegastamp($l_m = 50$)	97.55%	96.23%	96.89%	52.48%	51.13%	51.81%
RIHOOP($l_m = 50$)	**99.68**%	98.45%	99.06%	49.65%	53.30%	51.48%
Ours($l_m = 50$)	99.35%	**98.70**%	**98.44**%	**98.15**%	**98.73**%	**97.63**%
Stegastamp($l_m = 100$)	98.05%	97.13%	97.59%	66.23%	69.13%	67.68%
RIHOOP($l_m = 100$)	**99.88**%	99.35%	99.62%	51.43%	46.18%	48.81%
Ours($l_m = 100$)	99.73%	**99.68**%	**99.71**%	**98.83**%	**96.13**%	**97.48**%

angles vary in $\{0°, 30°\}$. In the table, the best two combinations for each method are underlined for emphasis. It can be observed that the three schemes present considerable robustness regardless of the monitor-camera combination. Furthermore, we observe a positive correlation between the number of camera pixels and the accuracy of watermark extraction. However, the minor exception to this is the Lenovo Legion Tab y700, which does not perform very well when the shooting angle is 0°. This may be attributed to the fact that the camera's larger vision field unexpectedly compromises the resolution of the captured watermarked image.

e) Robustness on printed images: At last, we test the robustness on another type of challenge distortions, printing-shooting attacks. To conduct this test, we printed the watermarked images using a TOSHIBA STUDIO457 color consumer printer and captured them using camera equipment in both indoor and outdoor environments. The settings for the camera equipment were identical to those used for the camera-shooting attacks.

We captured 40 images for each camera under each condition. The results of the watermark detection are visually presented in Fig. 12. Table IX reports the averaged accuracies, where "entire" and "part" indicate whether an entire image is captured. These results show that the proposed scheme effectively resists printing-shooting attacks, regardless of whether the entire image or just a portion is captured, and is not influenced by the camera model used.

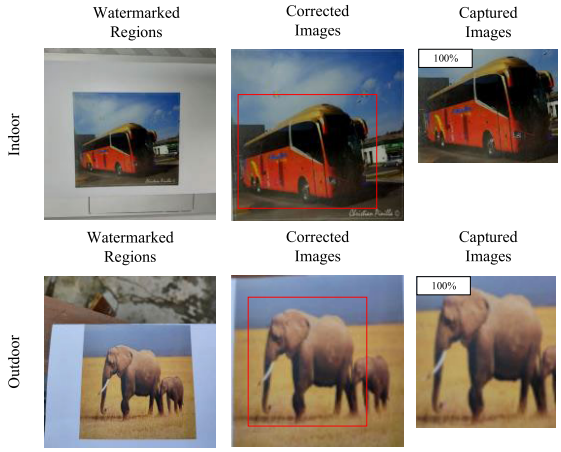


Fig. 12. Demonstration of watermark detection of the proposed scheme under printing-shooting attacks. In the first column are the captured images, in the second column are the corrected images, and in the last column are the detection results. These examples include different lighting conditions.

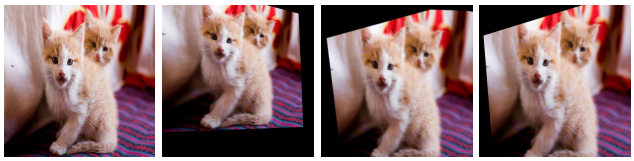
E. Ablation Study

In this section, we evaluate the fundamental design of the proposed scheme through ablation studies, particularly focusing on the geometric correction module and the loss function. It is worth noting that, for the sake of simplicity, we utilize a consistent value of $l_m = 50$ for all the test schemes presented in this section.

1) Geometric Correction: This subsection investigates three cases: no geometric correction, manual correction, and geometric correction module. In the first case, we directly decode the geometrically distorted images. In the second case, we manually align the distorted and original watermarked images through SIFT keypoint matching, and inverse the geometric distortion. Finally, in the last case, the geometric correction module described in Section IV-C.1 is used. Figure 13 demonstrates these cases. The comparison results are listed in Table X, where, besides a perspective warping with a strength of 0.1, distortion 1 is accompanied by defocusing and motion blurring, and distortion 2 is accompanied by Gaussian noising and JPEG compression. The computation time is assessed by using an Intel(R) Xeon(R) Gold 6226R

TABLE X
WATERMARK EXTRACTION RESULTS IN CASES OF DIFFERENT GEOMETRIC CORRECTION

No geometric correction	Manual Correction	Geometric correction module	Method	Distortion 1	Distortion 2	Cost time
✓	✗	✗	Stegastamp	74.19%	77.02%	0s
✓	✗	✗	RIHOOP	51.16%	52.34%	0s
✓	✗	✗	Ours	94.14%	93.62%	0s
✗	✓	✗	Stegastamp	96.16%	96.88%	0.076s
✗	✓	✗	RIHOOP	99.65%	**99.81**%	0.067s
✗	✓	✗	Ours	95.85%	98.27%	0.071s
✗	✗	✓	Ours	**99.73**%	99.49%	0.022s



(a) (b) (c) (d)

Fig. 13. Demonstration of geometric correction effect. (a) is the watermarked image, (b) is the geometric distorted version, (c) is the image after manual correction, and (d) is the image corrected with the geometric correction module.

TABLE XI
ABLATION RESULTS OF COMBINATIONS OF LOSS FUNCTIONS

Loss	PSNR↑	SSIM↑	LPIPS↓	ACE↓	ACC↑
L_{t1}	29.83	0.940	0.100	2.27	98.31%
L_{t2}	30.62	0.907	0.027	2.21	98.58%
L_{t3}	35.47	0.978	0.016	2.06	99.88%

CPU operating at 2.90GHz with 24GB of RAM, alongside a GeForce RTX 3090 GPU with 24GB of memory. This evaluation was conducted within an environment utilizing Python 3.8.8 and the Pytorch framework version 1.10.1+cu111.

It suggests that geometric correction is necessary, because the watermark extraction accuracy decreases seriously when no geometric correction is applied for all the schemes. However, surprisingly, the influence is less in the proposed scheme. It may be because watermarking regions are usually located at the center of the captured images, rendering them less susceptible to distortion than the surrounding regions. On the whole, the geometric correction module can provide better robustness with less computation time than manual correction. It confirms the effectiveness of the proposed geometric correction module.

2) Combination of Loss Function: In this subsection, we validate the effectiveness of the combined loss function described in Section IV-D.1 by training three models. The first model employs a loss function of $L_{t1} = L_{vgg} + L_2 + L_{sec}$, the second model uses a loss function of $L_{t2} = L_{lpips} + L_{vgg} + L_2 + L_{sec}$, and the third model uses a loss function of $L_{t3} = L_{ssim} + L_{dis} + L_{lpips} + L_{vgg} + L_2 + L_{sec}$. Experimental results of these three models are compared in Table XI. It demonstrates that the employed combination of loss functions can simultaneously enhance the performance of the geometric correction module (ACE reduces from 2.27 to 2.06), the watermark extraction accuracy of decoder (ACC increases from 98.31% to 99.88%), and the visual quality of watermarked images (PSNR increases from 29.83 to 35.47, SSIM increases from 0.940 to 0.978, and LPIPS reduces from 0.100 to 0.016). Figure 14 gives the corresponding visualization results.

(a) Host (b) L_{t1} (c) L_{t2} (d) L_{t3}

Fig. 14. Watermarked images obtained by the encoding end supervised with different combined loss functions.

VI. CONCLUSION

In this paper, we propose an instance-level watermarking scheme that can resist various distortions, including both camera-shooting and printing-shooting attacks. It is composed of an encoding end, a noise layer, and a decoding end. The encoding end consists of three modules: region selection module, watermark encoding module, and template embedding module. The region selection module selects watermarking regions according to image instances. The watermark encoding module uses a residual-dense structure-based encoder to generate a locally watermarked image, guided by the selected watermarking regions. Additionally, the template embedding module embeds templates at prefixed points for the watermark resynchronization. The decoding end also consists of three modules: geometric correction module, region detecting module, and watermark decoding module. The geometric correction module calculates location probability maps from the received watermarked image and a reference image. It uses

a calibrator to detect and inverse the perspective projection. The region detecting module detects watermarking region, and the watermark decoding module extracts watermark sequences from these regions. Furthermore, to address the geometric distortions that arise from camera shooting, we propose a novel template-based resynchronization method that effectively corrects such distortions without relying on visible markers. Experimental results demonstrate that our method exhibits strong resistance to camera-shooting attacks under various shooting conditions, accurately extracting embedded watermark sequences.

The proposed scheme offers a solution for tracking the unauthorized distribution of private digital media. In the event that watermark is detected within the distributed image, we can leverage it to trace its source of leakage. Furthermore, it can also be used for content authentication and copyright protection. For instance, we can ascertain the authenticity of image content if an image exhibits regions with conflicting watermark, lacking the intended watermark, or containing watermark associated with another image. The watermarked region can also serve as an invisible stamp, facilitating granular copyright protection for image content. Nevertheless, it is noteworthy that since the watermark information is not distributed across the entire image, our current scheme cannot be employed for integrity verification.

A limitation of our current scheme is that the length of the watermark sequence cannot exceed 100 bits due to the restricted watermarking region area. Increasing the watermarking capacity remains a focus for future work. Additionally, we are exploring other color models and resynchronization methods, such as binocular stereo, to further enhance robustness against camera-shooting attacks.

REFERENCES

[1] 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.

[2] A. Kamili, N. N. Hurrah, S. A. Parah, G. M. Bhat, and K. Muhammad, "DWFCAT: Dual watermarking framework for industrial image authentication and tamper localization," *IEEE Trans. Ind. Informat.*, vol. 17, no. 7, pp. 5108–5117, Jul. 2021.

[3] Z. Huang, B. Feng, and S. Xiang, "Robust reversible image watermarking scheme based on spread spectrum," *J. Vis. Commun. Image Represent.*, vol. 93, May 2023, Art. no. 103808.

[4] K. M. Hosny and M. M. Darwish, "Resilient color image watermarking using accurate quaternion radial substituted Chebyshev moments," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 15, no. 2, pp. 1–25, May 2019.

[5] Y. Tang, S. Wang, C. Wang, S. Xiang, and Y.-M. Cheung, "A highly robust reversible watermarking scheme using embedding optimization and rounded error compensation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 4, pp. 1593–1609, Apr. 2023.

[6] Z. Ma, W. Zhang, H. Fang, X. Dong, L. Geng, and N. Yu, "Local geometric distortions resilient watermarking scheme based on symmetry," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 12, pp. 4826–4839, Dec. 2021.

[7] H. Fang, W. Zhang, H. Zhou, H. Cui, and N. Yu, "Screen-shooting resilient watermarking," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 6, pp. 1403–1418, Jun. 2019.

[8] H. Fang et al., "A camera shooting resilient watermarking scheme for underpainting documents," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 11, pp. 4075–4089, Nov. 2020.

[9] T. Nakamura, A. Katayama, M. Yamamuro, and N. Sonehara, "Fast watermark detection scheme from analog image for camera-equipped cellular phone," *IEICE Trans. Inf. Syst.*, vol. 87, no. 12, pp. 2145–2155, 2004.

[10] A. Pramila, A. Keskinarkaus, and T. Seppänen, "Toward an interactive poster using digital watermarking and a mobile phone camera," *Signal, Image Video Process.*, vol. 6, no. 2, pp. 211–222, Jun. 2012.

[11] L. Dong, J. Chen, C. Peng, Y. Li, and W. Sun, "Watermark-preserving keypoint enhancement for screen-shooting resilient watermarking," in *Proc. IEEE Int. Conf. Multimedia Expo. (ICME)*, Jul. 2022, pp. 1–6.

[12] H. Mareen et al., "Camcording-resistant forensic watermarking fallback system using secondary watermark signal," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 9, pp. 3403–3416, Sep. 2021.

[13] X. Zhong, P.-C. Huang, S. Mastorakis, and F. Y. Shih, "An automated and robust image watermarking scheme based on deep neural networks," *IEEE Trans. Multimedia*, vol. 23, no. 1, pp. 1951–1961, Jul. 2020.

[14] H. Fang et al., "Deep template-based watermarking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 4, pp. 1436–1451, Apr. 2021.

[15] E. Wengrowski and K. Dana, "Light field messaging with deep photographic steganography," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1515–1524.

[16] M. Tancik, B. Mildenhall, and R. Ng, "StegaStamp: Invisible hyperlinks in physical photographs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2117–2126.

[17] S. Ge, J. Fei, Z. Xia, Y. Tong, J. Weng, and J. Liu, "A scree-shooting resilient document image watermarking scheme using deep neural network," *IET Image Process.*, vol. 17, no. 2, pp. 323–336, Feb. 2023.

[18] J. Jia et al., "RIHOOP: Robust invisible hyperlinks in offline and online photographs," *IEEE Trans. Cybern.*, vol. 52, no. 7, pp. 7094–7106, Jul. 2022.

[19] H. Fang, Z. Jia, Z. Ma, E.-C. Chang, and W. Zhang, "PIMoG: An effective screen-shooting noise-layer simulation for deep-learning-based watermarking network," in *Proc. 30th ACM Int. Conf. Multimedia*, Oct. 2022, pp. 2267–2275.

[20] F. Cao, T. Wang, D. Guo, J. Li, and C. Qin, "Screen-shooting resistant image watermarking based on lightweight neural network in frequency domain," *J. Vis. Commun. Image Represent.*, vol. 94, Jun. 2023, Art. no. 103837.

[21] H. Fang et al., "TERA: Screen-to-Camera image code with transparency, efficiency, robustness and adaptability," *IEEE Trans. Multimedia*, vol. 24, pp. 955–967, 2022.

[22] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, "Hidden: Hiding data with deep networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 657–672.

[23] Z. Jia, H. Fang, and W. Zhang, "MBRS: Enhancing robustness of DNN-based watermarking by mini-batch of real and simulated JPEG compression," in *Proc. 29th ACM Int. Conf. Multimedia*, Oct. 2021, pp. 41–49.

[24] 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.*, vols. 197–198, Aug. 2020, Art. no. 103015.

[25] B. Wang, Y. Wu, and G. Wang, "Adaptor: Improving the robustness and imperceptibility of watermarking by the adaptive strength factor," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 11, pp. 6260–6272, Nov. 2023.

[26] R. Shin and D. Song, "JPEG-resistant adversarial images," in *Proc. Workshop Mach. Learn. Comput. Secur.*, vol. 1, 2017, p. 8.

[27] C. Zhang, A. Karjauv, P. Benz, and I. S. Kweon, "Towards robust deep hiding under non-differentiable distortions for practical blind watermarking," in *Proc. 29th ACM Int. Conf. Multimedia*, Oct. 2021, pp. 5158–5166.

[28] J. Jia, Z. Gao, D. Zhu, X. Min, G. Zhai, and X. Yang, "Learning invisible markers for hidden codes in offline-to-online photography," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jul. 2022, pp. 2273–2282.

[29] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2961–2969.

[30] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 1–11.

[31] K. Alex Zhang, A. Cuesta-Infante, L. Xu, and K. Veeramachaneni, "SteganoGAN: High capacity image steganography with GANs," 2019, *arXiv:1901.03892*.

[32] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1125–1134.
[33] X. Zhang and Z. J. Wang, "Correlation-and-bit-aware multiplicative spread spectrum embedding for data hiding," in *Proc. IEEE Int. Workshop Inf. Forensics Security*, Nov. 2013, pp. 186–190.
[34] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Deep image homography estimation," 2016, *arXiv:1606.03798*.
[35] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Proc. Eur. Conf. Comput. Vis.*, Amsterdam, Netherlands, 2016, pp. 694–711.
[36] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 586–595.
[37] W. Zhou, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Jul. 2004.
[38] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2794–2802.
[39] T. Lin et al., "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.
[40] T. Furon, "A constructive and unifying framework for zero-bit watermarking," *IEEE Trans. Inf. Forensics Security*, vol. 2, no. 2, pp. 149–163, Jun. 2007.
[41] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic gradient descent," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–15.

Mingjin He received the M.S. degree from Jinan University, Guangzhou, China, in 2023.

His research interests include multimedia forensics and security.

Bingwen Feng received the B.E. and Ph.D. degrees from Sun Yat-sen University, Guangzhou, China, in 2008 and 2014, respectively.

He is currently an Associate Professor with the College of Cyber Security, Jinan University, Guangzhou. His research interests include multimedia security, AI security, and privacy protection.

Yizhi Guo received the B.S. degree from Guangzhou University, Guangzhou, China, in 2023. He is currently pursuing the M.S. degree with the College of Cyber Security, Jinan University, Guangzhou.

His research interests include multimedia security protection.

Jian Weng (Senior Member, IEEE) received the Ph.D. degree in computer science and engineering from Shanghai Jiao Tong University, Shanghai, China, in 2008.

From 2008 to 2010, he was a Post-Doctoral Fellow with the School of Information Systems, Singapore Management University, Singapore. He is currently a Professor and the Vice-Chancellor with Jinan University, Guangzhou, China. He has published over 300 papers in cryptography and security conferences and journals, such as CRYPTO, EUROCRYPT, ASIACRYPT, CCS, Usenix Security, NDSS, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, and IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING. His research interests include public key cryptography, cloud security, and blockchain.

Prof. Weng served as the PC co-chair or a PC member for more than 30 international conferences. He serves as an Associate Editor for IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY.

Wei Lu (Member, IEEE) received the B.S. degree in automation from Northeast University, China, in 2002, and the M.S. and Ph.D. degrees in computer science from Shanghai Jiao Tong University, China, in 2005 and 2007, respectively.

He was a Research Assistant with The Hong Kong Polytechnic University from 2006 to 2007. He is currently a Professor with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China. His research interests include multimedia forensics and security, data hiding and watermarking, and privacy protection. He is an Associate Editor of *Signal Processing* and *Journal of Visual Communication and Image Representation*.