

# Gaussian Shading++: Rethinking the Realistic Deployment Challenge of Performance-Lossless Image Watermark for Diffusion Models

Zijin Yang, Xin Zhang, Kejiang Chen, Kai Zeng, Qiyi Yao, Han Fang, Weiming Zhang, Nenghai Yu

**Abstract**—Ethical concerns surrounding copyright protection and inappropriate content generation pose challenges for the practical implementation of diffusion models. One effective solution involves watermarking the generated images. Existing methods primarily focus on ensuring that watermark embedding does not degrade the model performance. However, they often overlook critical challenges in real-world deployment scenarios, such as the complexity of watermark key management, user-defined generation parameters, and the difficulty of verification by arbitrary third parties. To address this issue, we propose Gaussian Shading++, a diffusion model watermarking method tailored for real-world deployment. We propose a double-channel design that leverages pseudorandom error-correcting codes to encode the random seed required for watermark pseudorandomization, achieving performance-lossless watermarking under a fixed watermark key and overcoming key management challenges. Additionally, we model the distortions introduced during generation and inversion as an additive white Gaussian noise channel and employ a novel soft decision decoding strategy during extraction, ensuring strong robustness even when generation parameters vary. To enable third-party verification, we incorporate public key signatures, which provide a certain level of resistance against forgery attacks even when model inversion capabilities are fully disclosed. Extensive experiments demonstrate that Gaussian Shading++ not only maintains performance losslessness but also outperforms existing methods in terms of robustness, making it a more practical solution for real-world deployment.

**Index Terms**—Image watermark, Diffusion models, Performance-lossless.

## I. INTRODUCTION

Diffusion models [1]–[5] signify a noteworthy leap forward in image generation. These well-trained diffusion models, especially commercial models like Stable Diffusion (SD) [6], Glide [7], and Muse AI [6], enable individuals with diverse backgrounds to create high-quality images effortlessly. However, this raises concerns about intellectual property and whether diffusion models will be stolen or resold twice.

On the other hand, the ease of generating realistic images raises concerns about potentially misleading content generation. For example, on May 23, 2023, a Twitter-verified user

Z. Yang, X. Zhang, K. Chen, Q. Yao, W. Zhang, N. Yu are with University of Science and Technology of China, Hefei 230026, China and Anhui Province Key Laboratory of Digital Security (email: {bsmhmmf@mail., XinZhang1999@mail., chenkj@, qyyao@mail., zhangwm@, ynh@}ustc.edu.cn).

K. Zeng is with the Department of Information Engineering and Mathematics, University of Siena, Siena, Italy (email: kai.zeng@unisi.it).

H. Fang is with the School of Computing, National University of Singapore, 117417, Singapore (email: fanghan@nus.edu.sg).

Zijin Yang and Xin Zhang contributed equally to this work.

Corresponding author: Kejiang Chen.

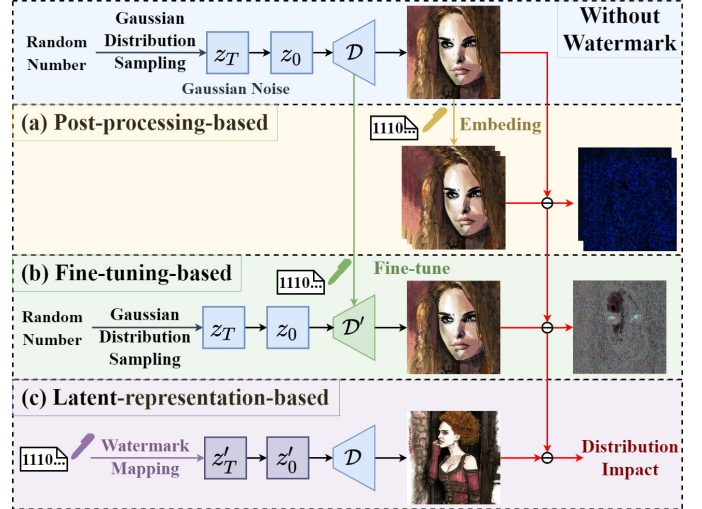


Fig. 1. Existing watermarking frameworks can be divided into three categories: (a) post-processing-based, (b) fine-tuning-based, and (c) latent-representation-based. Since methods (a) and (b) either introduce watermark residuals or require additional computational overhead, method (c) has emerged as the mainstream approach by overcoming these two drawbacks. Their performance is primarily evaluated based on the impact on the distribution.

named Bloomberg Feed posted a tweet titled “Large explosion near the Pentagon complex in Washington DC-initial report,” along with a synthetic image. This tweet led to multiple authoritative media accounts sharing it, even causing a brief impact on the stock market<sup>1</sup>. On June 13, 2024, the European Union enacted the Artificial Intelligence Act, which mandates the implementation of technical safeguards including watermarks to prevent AI-generated content from misleading the public, ensuring transparency and credibility in the information ecosystem<sup>2</sup>. The urgency of labeling generated content for copyright authentication and prevention of misuse is evident.

Watermarking is highlighted as a fundamental method for labeling generated content, as it embeds watermark information within the generated image, allowing for subsequent copyright authentication and the tracking of false content. Existing watermarking methods for the diffusion model can be divided into three categories, as shown in Fig. 1. Post-processing-based methods [8]–[20] adjust robust image features to embed watermarks, thereby directly altering the image

<sup>1</sup>Fake image of Pentagon explosion on Twitter.

<sup>2</sup>Artificial Intelligence Act: Regulation (EU) 2024/1689 of the European Parliament and of the Council.

and degrading its quality. To mitigate this concern, recent research endeavors propose fine-tuning-based methods [21]–[25], which amalgamate the watermark embedding process with the image generation process. Intuitively, these methods need to modify model parameters, introducing supplementary computational overhead.

To address two main concerns of visible watermark residuals in generated images and excessive computational overhead inherent in prior methods, latent-representation-based methods [26]–[28] has emerged as a promising solution. These methods ensure that watermark information remains invisible in the image and offer plug-and-play functionality without requiring training, making it an increasingly important focus of research. Wen et al. [26] introduced the Tree-Ring Watermark (TRW), the first of its kind, which embeds information by modifying latent representations to align with specific patterns. However, it restricts the randomness of sampling, which impacts the generative performance. Our earlier work, Gaussian Shading [27], resolves this limitation by incorporating pseudorandom keys and distribution-preserving sampling during the mapping of watermark information to latent representations. This ensures that the distribution of watermarked images matches that of non-watermarked images, achieving the first provable performance-lossless watermarking scheme. However, Gaussian Shading [27] requires assigning a unique pseudorandom key to each image, which introduces significant challenges in key management in practical implementation. To tackle the key management issue, Gunn et al. [28] proposed an undetectable watermarking method called the pseudorandom error-correcting codes watermark (PRCW). The core of PRCW [28] lies in pseudorandom error-correcting codes (PRC) [29], where the generator matrix and parity check matrix serve as the watermark key. Even when the watermark key is fixed, PRCW [28] can generate pseudorandom bitstreams, which are then mapped to latent representations that follow a standard normal distribution. This effectively resolves the key management problem.

However, our experiments reveal that the robustness of PRCW [28] is highly sensitive to the *guidance\_scale* parameter. As shown in Tab. VI, PRCW [28] only demonstrates strong robustness when the *guidance\_scale* values used during generation and inversion are identical. Performance degrades significantly when there is a mismatch between these values. This poses a challenge for real-world deployment, as platforms typically allow users to customize generation parameters, which are unknown during watermark verification.

Based on the above analysis, latent-representation-based watermarking methods currently face three main challenges in practical deployment: First, how to achieve performance-lossless watermarking, where the distribution of watermarked images aligns with that of non-watermarked images, with a fixed watermark key to simplify key management. Second, watermarking schemes should accommodate user-customized generation parameters and maintain robust performance even when mismatches occur between the generation and verification phases. Lastly, many existing watermarking schemes depend on operators for both watermark embedding and verification, introducing potential security and trustworthiness risks

in the authentication process [30]. Consequently, watermark verification needs to be publicly accessible to any third party.

To address the aforementioned three challenges, we propose Gaussian Shading++, a more practical performance-lossless watermarking scheme designed for real-world deployment scenarios. Specifically, during watermark embedding, we propose a double-channel design, in which the latent space is evenly divided along the channel dimension into two parts: the GS Channel and the PRC Channel. The GS Channel continues to use Gaussian Shading [27], where the robustness is enhanced by replicating the watermark content. The PRC Channel serves as the watermark header, encoding the seed for the pseudorandom number generator (PRNG) used in Gaussian Shading [27] via PRC [29]. The actual watermark key is the generator matrix and parity check matrix of PRC, and a private key that is used together with the seed to drive the PRNG. While the watermark key remains fixed, the seed can vary freely. The fixed watermark key does not compromise the pseudorandomness of the PRC Channel, and the randomly sampled seed ensures the pseudorandomness of the GS Channel. Thus, the entire watermark remains pseudorandom even with a fixed watermark key. Under the effect of distribution-preserving sampling, both the latent representations mapped by the watermark and those from random sampling follow a standard Gaussian distribution. Since the subsequent generation process remains unchanged, the distribution of watermarked images aligns with that of non-watermarked images, achieving performance-lossless watermarking while addressing the key management issue.

During watermark extraction, inspired by PRCW [28], we employ a more accurate Exact Inversion method [31] to recover the latent representations from images. The seed is then extracted from the PRC Channel. Next, we propose a novel soft decision decoding strategy for the GS Channel, which significantly improves the hard decision method in Gaussian Shading [27] that employs direct binarization of latent representations. By modeling the entire generation and inversion process as an additive white Gaussian noise channel (AWGN Channel), we reformulate watermark recovery as a maximum likelihood decoding problem of repetition codes (REP codes) [32]. Specifically, each watermark bit undergoes multiple embeddings in the latent space, and we compute the posterior expectation of each repeated instance based on the estimated noise. These expectations are then aggregated to form a log-likelihood ratio (LLR) for each bit, enabling near-optimal soft decision decoding that equivalently performs maximum a posteriori (MAP) estimation under the assumption of independent Gaussian noise. Thanks to the accurate inversion, efficient modeling, and soft decision decoding, the robustness of Gaussian Shading++ is further improved, ensuring that the watermark can still be correctly extracted even when the parameter *guidance\_scale* varies.

To further extend the scheme to third-party verifiability, it is necessary to make the model inversion capability public, which implies that third parties can access the latent representations, thereby introducing the risk of forgery using surrogate models [33]. To address this, we incorporate the public-key signature ECDSA [34] into the framework, using

a private signing key to sign the user information in the GS Channel. During extraction, the extracted watermark from the GS Channel is further verified using a public verification key to authenticate the signature. After forgery, the watermark accuracy of the image decreases, making it difficult to pass the signature verification. This ensures that Gaussian Shading++ can resist a certain degree of forgery attacks [33].

To demonstrate the effectiveness of our method, we evaluate Gaussian Shading++ against traditional distortions and neural network-based removal attacks [6], [35]–[37], comparing it with several state-of-the-art methods to validate its superior robustness. Additionally, we highlight the performance-lossless nature of Gaussian Shading++ by comparing both the visual quality of images and the distribution of latent representations. Furthermore, Gaussian Shading++ exhibits strong robustness even in scenarios where the *guidance\_scale* parameter varied during generation. Finally, we assess the robustness of Gaussian Shading++ in third-party verifiable scenarios, demonstrating a certain level of resilience against existing forgery attacks [33]. Overall, Gaussian Shading++ demonstrates strong practicality in real-world applications.

To summarize, our contributions are as follows:

- We propose a double-channel design, utilizing the PRC Channel to encode the random seed required for the pseudorandomization of the GS Channel. This approach overcomes the limitations of complex key management in Gaussian Shading and achieves performance-lossless even with a fixed watermark key.
- By building on the prior modeling of the generation and inversion process as the AWGN Channel, we propose a novel soft decision decoding strategy for the maximum likelihood decoding in REP codes. This near-optimal approach effectively enhances the robustness of the GS Channel, enabling Gaussian Shading++ to achieve excellent performance across varying generation parameters.
- By introducing the public-key signature ECDSA, we extend the application scenario to enable verification by any third party, while providing a certain level of resistance against existing forgery attacks.
- The experimental results demonstrate that our method outperforms state-of-the-art methods in terms of both robustness and performance losslessness, further advancing the practical applicability of watermarking in diffusion models.

## II. RELATED WORK

### A. Diffusion Models

Inspired by nonequilibrium thermodynamics [1], Ho et al. [4] introduced the Denoising Diffusion Probabilistic Model (DDPM). DDPM consists of two Markov chains used for adding and removing noise, and subsequent works [5], [6], [38]–[43] have adopted this bidirectional chain framework. To reduce computational complexity and improve efficiency, the Latent Diffusion Model (LDM) [6] was designed, in which the diffusion process occurs in a latent space  $\mathcal{Z}$ . To map an image  $x \in \mathbb{R}^{H \times W \times 3}$  to the latent space, the LDM employs an encoder  $\mathcal{E}$ , such that  $z_0 = \mathcal{E}(x) \in \mathbb{R}^{h \times w \times ch}$ . Similarly,

to reconstruct an image from the latent space, a decoder  $\mathcal{D}$  is used, such that  $x = \mathcal{D}(z_0)$ . A pretrained LDM can generate images without the encoder  $\mathcal{E}$ . Specifically, a latent representation  $z_T$  is first sampled from a standard Gaussian distribution  $\mathcal{N}(0, I)$ . Subsequently, through iterative denoising using methods like DDIM [5],  $z_0$  is obtained, and an image can be generated using the decoder:  $x = \mathcal{D}(z_0)$ .

### B. Diffusion Inversion

Diffusion inversion [5], [31], [44], [45] can be regarded as the inverse process of generation, recovering latent representations from images for various downstream tasks, such as image editing [46]–[51] and watermark detection [26]–[28]. The most native approach is DDIM Inversion [5], which simply reverses the time axis and employs the same sampling method as DDIM generation. However, DDIM Inversion struggles with stable reconstruction of real images, potentially leading to incorrect image reconstruction in downstream tasks. Wallace et al. [44] proposed maintaining two coupled latent representations and achieving precise inversion of real and generated images through an alternating approach. Zhang et al. [45] introduced a bi-directional integration approximation method to perform exact diffusion inversion. However, while improving inversion accuracy, these methods modify the sampling process and are not universally applicable to images generated by common sampling methods [5], [40]. Hong et al. [31] achieved a more general diffusion inversion method by utilizing gradient descent or forward step methods, further enhancing reconstruction accuracy.

### C. Image Watermark

Digital watermark [52] is an effective means to address copyright protection and content authentication by embedding copyright or traceable identification information within carrier data. Typically, the functionality of a watermark depends on its capacity. For example, a single-bit watermark can determine whether an image was generated by a particular diffusion model, i.e., copyright protection; a multi-bit watermark can further determine which user of the diffusion model generated the image, i.e., traceability.

Image watermark is a method that employs images as carriers for the watermark. Initially, watermark embedding methods primarily focused on the spatial domain [52], but later, to enhance robustness, transform domain watermarking techniques [53]–[59] were developed. In recent years, with the advancement of deep learning, researchers have turned their attention to neural networks [60], [61], harnessing their powerful learning capabilities to develop watermarking techniques [10]–[20].

### D. Image Watermark for Diffusion Models

Existing image watermarking methods for the diffusion model [8]–[28] can be divided into three categories, as shown in Fig. 1. The image watermarking methods described in the previous section can be applied directly to the images generated by the diffusion model, which is called post-processing-based watermarks [8]–[20]. These methods directly modify

the image, thus degrading image quality. Recent research endeavors have amalgamated the watermark embedding process with the image generation process to mitigate this issue. Stable Signature [21] fine-tunes the LDM decoder using a pre-trained watermark extractor, facilitating watermark extraction from images produced by the fine-tuned model. Zhao et al. [22] and Liu et al. [23] suggest fine-tuning the diffusion model to implant a backdoor as a watermark, enabling watermark extraction by triggering. These fine-tuning-based approaches enhance the quality of watermarked images but introduce supplementary computational overhead and modify model parameters.

To address the limitations of the aforementioned two types of methods, Wen et al. [26] proposed the first latent-representation-based method named the Tree-Ring Watermark (TRW), which conveys copyright information by adapting the frequency domain of latent representations to match specific patterns. This method achieves an imperceptible watermark. However, it directly disrupts the Gaussian distribution of noise, limiting the randomness of sampling and resulting in affecting model performance. In our previous work, Gaussian Shading [27], we introduced a stream cipher and distribution-preserving sampling to ensure that the distribution of watermarked images matches that of non-watermarked images, achieving a performance-lossless watermarking scheme. However, Gaussian Shading requires assigning a unique stream key to each image, leading to key management challenges. Gunn et al. [28] proposed the pseudorandom error-correcting codes watermark (PRCW), which uses pseudorandom error-correcting codes [29] to encode watermark information. This approach allows the generation of latent representations that conform to a standard Gaussian distribution even when the watermark key is fixed, effectively addressing the key management issue in the performance-lossless watermark. However, the robustness of PRCW degrades when faced with variations in generation parameters.

In this paper, we aim to design a watermarking method that achieves performance-lossless even with a fixed watermark key, while maintaining strong robustness under scenarios with varying generation parameters. Additionally, considering practical applicability, the method should ideally be extendable to support verification by any third party.

### E. Pseudorandom Error-correcting Codes Based on LDPC Codes

To address the key reuse issue in Gaussian Shading [27] and enable consistent performance-lossless capabilities across multiple image generations, we introduce the pseudorandom error-correcting codes (PRC) based on LDPC codes [29] to embed a header composed of random seeds.

The key generation, encoding, and decoding procedures of the PRC based on LDPC codes are as follows:

- **KeyGen:**  $(n, g, t, r) \mapsto (P, G)$ 
  - Sample a random matrix  $P \in \mathbb{F}_2^{r \times n}$  (parity-check matrix) subject to every row of  $P$  being  $t$ -sparse.
  - Sample a random matrix  $G \in \mathbb{F}_2^{n \times g}$  (generator matrix) subject to  $PG = 0$ .

- **Encode( $m$ ):** Given message  $m \in \mathbb{F}_2^g$ , sample noise  $e \leftarrow \text{Ber}(n, \eta)$ , and output ciphertext  $c = Gm \oplus e$ .
- **Decode( $L$ ):** Given a vector of posterior soft information  $L = (\ell_1, \ell_2, \dots, \ell_n)$  (e.g.,  $\ell_i = \mathbb{E}[m_i | c']$ ), apply the BP-OSD decoder to recover the original message  $\hat{m}$ .

The above definition corresponds to the standard regular LDPC codes, where the sparsity parameter  $t$  is fixed. To satisfy the requirements of PRC, the following constraint must be imposed:

- The sparsity is set as  $t = \Theta(\log n)$ , and each execution of **Encode** samples a fresh message  $m$  uniformly at random from  $\mathbb{F}_2^g$ .

When the above constraints are satisfied, the output of **Encode** is pseudorandom under either **the subexponential Learning Parity with Noise (LPN) assumption**, or under **the standard LPN assumption** combined with **the planted XOR assumption** (see in [29]); that is: For any polynomial-time adversary  $\mathcal{A}$ ,

$$\left| \Pr_{P, G} [\mathcal{A}^{\text{Encode}(\cdot)}(1^\lambda) = 1] - \Pr_{\mathcal{U}} [\mathcal{A}^{\mathcal{U}}(1^\lambda) = 1] \right| \leq \text{negl}(\lambda). \quad (1)$$

In later sections, we will show that by choosing an appropriate sparsity parameter  $t$  and using a randomly sampled seed from  $\mathbb{F}_2^g$  as the input to the PRC encode process, each image generation involves fresh randomness. As a result, our construction inherits the pseudorandomness guarantees established in prior theoretical work.

## III. THREAT MODEL

In this section, we introduce the threat model considered for the proposed method. As illustrated in Fig. 2, it is divided into two scenarios: Operator Verification and Third-party Verification. We also present the Watermark Statistical Test for detection and traceability tasks.

### A. Operator Verification

As shown in Fig. 2, the scenario involves the operator Alice, the thief Carol, and two types of users Bob and Trudy.

1) **Operator Alice:** Alice is responsible for training the model, deploying it on the platform, and providing the corresponding API for users, but she does not open-source the code or model weights. On one hand, to fulfill the detection (copyright protection) requirement, Alice embeds a single-bit watermark into each generated image. The successful extraction of the watermark from an image serves as evidence of Alice's rightful ownership of the copyright, while also indicating that the image is artificially generated (as opposed to natural images). On the other hand, to meet the traceability requirement, Alice assigns a unique watermark to each user. By extracting the watermark from illicit content, it becomes possible to trace malicious user Trudy through comparison with the watermark database. Traceability represents a higher-level objective than detection and can also achieve copyright protection for different users.

2) **Community user Bob:** Bob faithfully adheres to the community guidelines, utilizing the API provided by Alice to generate and disseminate images.

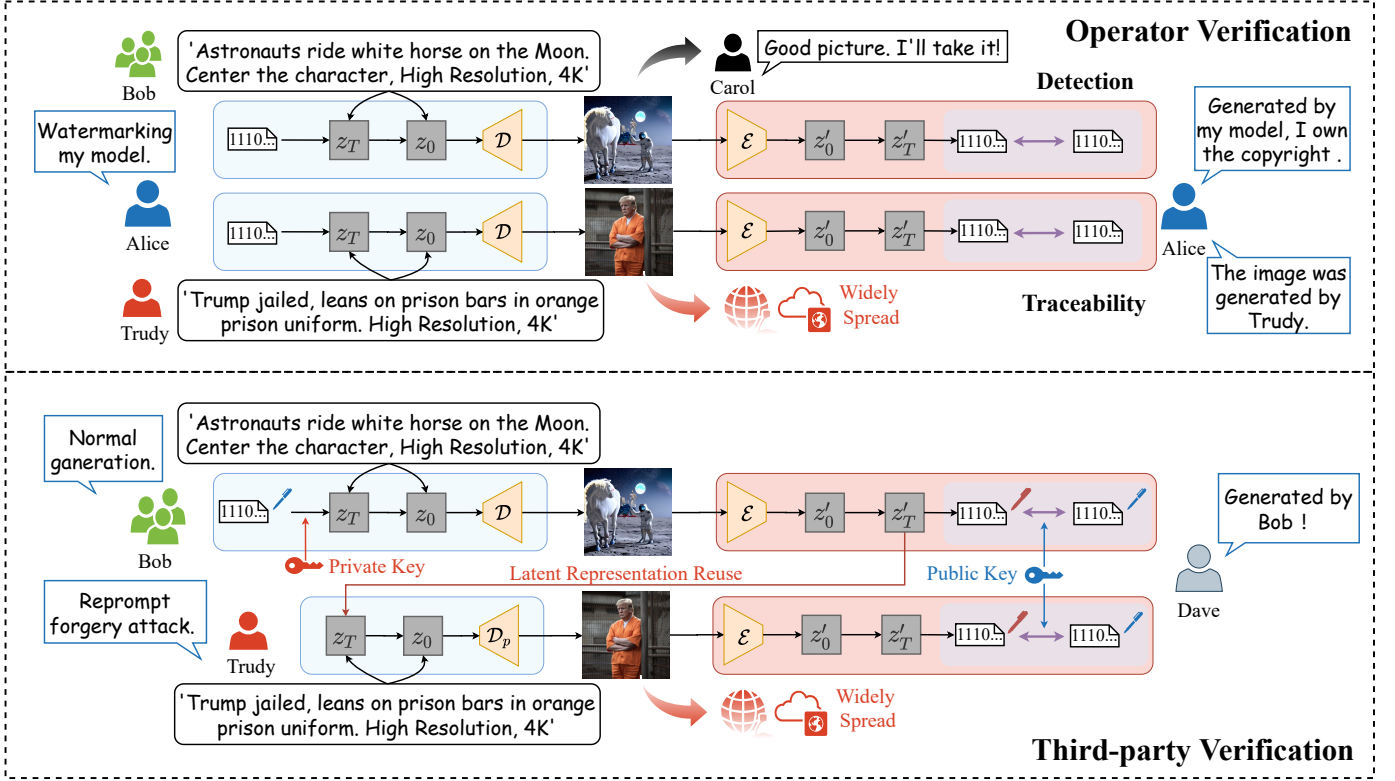


Fig. 2. The two application scenarios of Gaussian Shading++ are Operator Verification and Third-party Verification. In the Operator Verification scenario, Gaussian Shading++ considers satisfying the requirements of generated image detection (copyright protection) and malicious user traceability. In the Third-party Verification scenario, Gaussian Shading++ considers the need for any third party to verify the watermark, and it aims to defend against the reprompt forgery attack that may emerge once model inversion capabilities become publicly available.

3) *Thief Carol*: Carol does not use Alice's services but steals images generated by her model, claiming ownership of the copyrights.

4) *Malicious user Trudy*: Trudy uses the API provided by Alice to generate deepfakes and infringe content. To evade detection and traceability, Trudy can employ various data augmentation techniques to modify illicit images.

### B. Third-party Verification

As shown in Fig. 2, the scenario involves the operator Alice, any third party Dave who wishes to verify the watermark, and two types of users Bob and Trudy.

1) *Operator Alice*: Alice provides users with an API, but in the Third-party Verification scenario, she must expose the model's inversion capability. To mitigate the risk of malicious user Trudy exploiting the watermark for forgery attacks, Alice can maintain a public-private key pair. During the generation process, Alice uses the private signing key to sign the user information, and this signature, combined with the user information, forms the watermark information. When a third party Dave, needs to verify the watermark, Alice can provide the public verification key directly or store it in a digital certificate, making it readily available for Dave's use. Additionally, to enable Dave to trace back to the target user based on the user information after signature verification, Alice also needs to maintain a public watermark database to facilitate Dave's query operations.

2) *Any third party Dave*: Dave can be any third party seeking to verify the watermark, and he requires both Alice's public verification key and access to the publicly available model inversion capability. Once the signature verification is successful, Dave can trace the user information in the watermark database maintained by Alice.

3) *Community user Bob*: Bob faithfully adheres to the community guidelines, utilizing the API provided by Alice to generate and disseminate images. Bob's user information is signed using Alice's private signing key, and his user information is made publicly available in the watermark database.

4) *Malicious user Trudy*: Trudy's goal is to disguise illicit content as being generated by other users. Although Alice has made the watermark database publicly accessible, Trudy cannot directly forge content without access to Alice's private key. Therefore, Trudy can only exploit the publicly available model inversion capability to obtain the target user's latent representations. By leveraging a proxy model  $\mathcal{D}_p$ , Trudy can reuse these latent representations to regenerate content under illicit prompts, thereby executing a reprompt forgery attack [33].

### C. Watermark Statistical Test

1) *Detection*: Alice embeds a single-bit watermark, represented by  $q$ -bit binary watermark  $s \in \{0, 1\}^q$ , into each generated image using Gaussian Shading++. This watermark serves as an identifier for her model. Assuming that the watermark  $s'$  is extracted from the image  $X$ , the detection test for

the watermark can be represented by the number of matching bits between two watermark sequences,  $Acc(s, s')$ . When the threshold  $\tau \in \{0, \dots, q\}$  is determined, if  $Acc(s, s') \geq \tau$ , it is deemed that  $X$  contains the watermark.

In previous works [62], it is commonly assumed that the extracted watermark bits  $s'_1, \dots, s'_q$  from the vanilla images are independently and identically distributed, with  $s'_i$  following a Bernoulli distribution with parameter 0.5. Thus,  $Acc(s, s')$  follows a binomial distribution  $Ber(q, 0.5)$ .

Once the distribution of  $Acc(s, s')$  is determined, the false positive rate (FPR) is defined as the probability that  $Acc(s, s')$  of a vanilla image exceeds the threshold  $\tau$ . This probability can be further expressed using the regularized incomplete beta function  $B_x(a; b)$  [21],

$$\begin{aligned} \text{FPR}(\tau) &= \Pr(Acc(s, s') > \tau) = \frac{1}{2^q} \sum_{i=\tau+1}^q \binom{q}{i} \\ &= B_{1/2}(\tau+1, q-\tau). \end{aligned} \quad (2)$$

2) *Traceability*: To enable traceability, Alice needs to assign a watermark  $s^i \in \{0, 1\}^q$  to each user, where  $i = 1, \dots, N$  and  $N$  represents the number of users. During the traceability test, the bit matching count  $Acc(s^1, s'), \dots, Acc(s^N, s')$  needs to be computed for all  $N$  watermarks. If none of the  $N$  tests exceed the threshold  $\tau$ , the image is considered not generated by Alice's model. However, if at least one test passes, the image is deemed to be generated by Alice's model, and the index with the maximum matching count is traced back to the corresponding user, i.e.,  $\arg\max_{i=1, \dots, N} Acc(s^i, s')$ . When a threshold  $\tau$  is given, the FPR can be expressed as follows [21],

$$\text{FPR}(\tau, N) = 1 - (1 - \text{FPR}(\tau))^N \approx N \cdot \text{FPR}(\tau). \quad (3)$$

#### IV. PROPOSED METHOD

In this section, we first introduce the workflow of the proposed method, Gaussian Shading++, as illustrated in Fig. 3. We propose a double-channel design by partitioning the latent space into two components: the PRC Channel and the GS Channel. In the watermark key generation phase, a ternary key set needs to be generated for the PRC channel. In the Third-party Verification scenario, we also need to introduce the public-key signature ECDSA [34] key pair to defend against forgery attacks [33]. During embedding, the PRC Channel serves as the watermark header, encoding the random seed required for the GS Channel. The GS Channel enhances robustness by performing diffusion on the actual watermark message. The two Channels are then merged and used to drive distribution-preserving sampling, followed by denoising to generate watermarked images. During extraction, Exact Inversion [31] is employed to recover the latent representation. The distortion throughout the entire generation and inversion process is modeled as an AWGN Channel, enabling posterior estimation of the latent representation symbols. Subsequently, the random seed in the PRC Channel is first recovered, followed by decryption and soft decision decoding to extract the watermark from the GS Channel. At the end of this section, we provide theoretical proof of the performance-lossless characteristic of Gaussian Shading++.

##### A. Watermark Key Generation

The watermarking scheme utilizes a composite key consisting of two components. The first component is an LDPC key used to construct the PRC Channel. The second component is a private key  $sk_c$ , which is involved in the generation of the stream key for the GS Channel.

Suppose the latent representations have dimensions  $ch \times h \times w$  and each dimension can represent  $v$  bits. We designate the first half of the channels for PRC embedding, and refer to these channels as the PRC Channel. For the second half of the channels, we refer to them as the GS Channel.

We follow the framework of the pseudorandom error-correcting codes based on LDPC codes (Sec. II-E). Specifically, the LDPC key is  $(P, G) \leftarrow \text{KeyGen}(\frac{ch \times h \times w}{2}, n_{seed}, t, r)$ . The private key  $sk_c$  is drawn uniformly at random from the binary space  $\{0, 1\}^{n_{sk}}$ .

For the Operator Verification scenario, the watermark key components  $(P, G, sk_c)$  are treated as private. In contrast, for the Third-party Verification scenario,  $(P, G, sk_c)$  are considered public parameters, while the actual watermark generation and extraction rely on an ECDSA [34] key pair  $(SK, VK)$ , where  $SK$  is the private signing key and  $VK$  is the corresponding public verification key.

##### B. Watermark Embedding

In both scenarios, watermark embedding should be strictly performed by the operator to prevent high-precision forgery attacks from malicious third parties. The specific process is as follows:

1) *PRC Channel*: We first sample  $seed \in \mathbb{F}_2^{n_{seed}}$  of PRNG uniformly at random, which is used to initialize the PRNG within the GS Channel for stream cipher generation. Then, we encode  $seed$  into a codeword using the LDPC codes-based PRC: sample noise  $e \leftarrow \text{Ber}(n, \eta)$ ,  $m_c = G \cdot seed \oplus e$ . Finally, a mapping is applied to convert 0 to 1 and 1 to  $-1$  on the  $m_c$ , resulting in  $m_{prc} = (-1)^{m_c}$ .

2) *GS Channel*: Given that the entire latent space is equally partitioned into two components, the watermark capacity becomes  $\frac{v \times ch \times h \times w}{2}$  bits. To enhance the robustness of the watermark, we introduce a spatial replication factor  $\frac{1}{f_{hw}}$  and a channel replication factor  $\frac{1}{f_{ch}}$ . We represent the watermark using  $\frac{1}{f_{hw}}$  of the height and width, and  $\frac{1}{f_{ch}}$  of the channel, and replicate the watermark  $f_{ch} \cdot f_{hw}^2$  times. Thus, the watermark  $s$  with dimensions  $v \times \frac{ch}{2f_{ch}} \times \frac{h}{f_{hw}} \times \frac{w}{f_{hw}}$  is expanded into a diffused watermark  $s^d$  with dimensions  $\frac{v \times ch \times h \times w}{2}$ . The actual watermark capacity is  $q = \frac{v \times ch \times h \times w}{2f_{ch} \cdot f_{hw}^2}$  bits. In the third-party verifiable scenario, to defend against forgery attacks [33], we further introduce the public-key signature ECDSA [34], where the watermark  $s$  consists of the user information and its signature generated using the private signing key  $SK$ .

If we know the distribution of the diffused watermark  $s^d$ , we can directly utilize distribution-preserving sampling to obtain the corresponding latent representations  $z_T^s$ . However, in practical scenarios, its distribution is always unknown. Hence, we introduce a stream key  $K$  to transform  $s^d$  into a distribution-known randomized watermark  $m_e$  through encryption. Specifically, we derive  $K$  by concatenating the seed



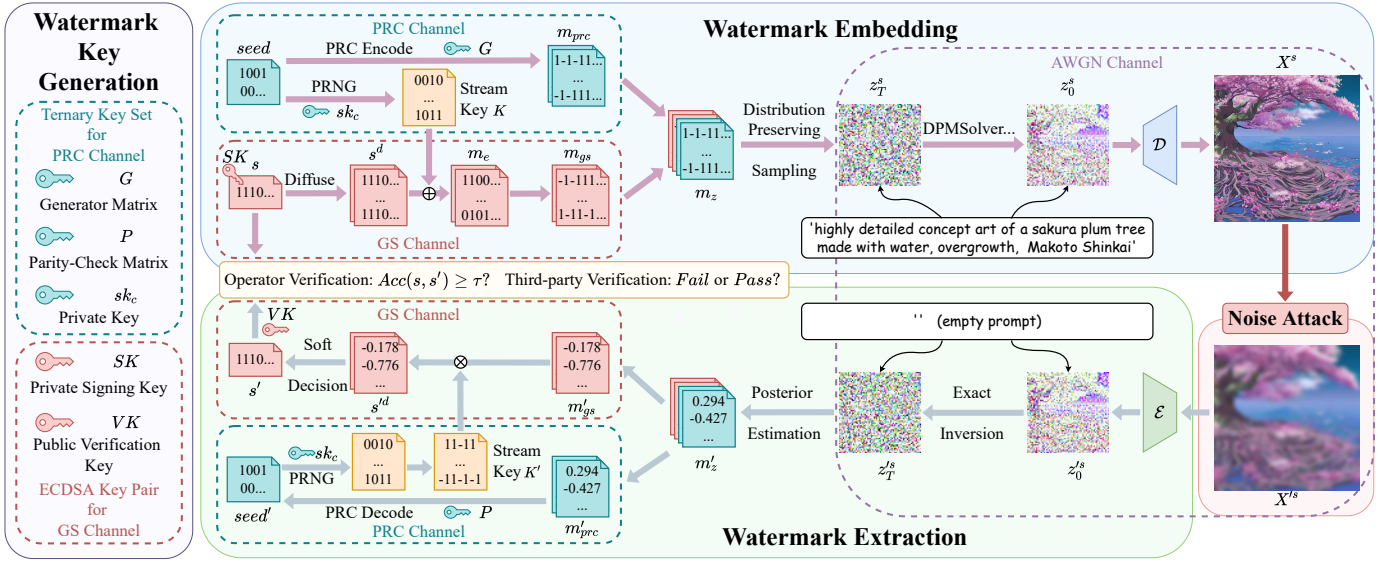


Fig. 3. The framework of Gaussian Shading++. The latent space is divided into the PRC Channel and GS Channel. During the watermark key generation, a ternary key set is generated for the PRC Channel. In the Third-party Verification scenario, a public-key signature ECDSA [34] key pair is introduced. During the watermark embedding, the PRC Channel serves as the header, encoding the random  $seed$  to drive a PRNG, resulting in  $m_{prc}$ . The GS Channel embeds a  $k$ -bit watermark sequence  $s$ , which undergoes diffusion, encryption, and transformation to produce  $m_{gs}$ . The combined sequence of  $m_{prc}$  and  $m_{gs}$  is used to drive distribution-preserving sampling, followed by denoising to generate watermarked images  $X^s$ . For watermark extraction, the process begins with Exact Inversion [31] to recover  $z_T^s$ . The distortion throughout the entire generation and inversion process is modeled as an AWGN channel, enabling posterior estimation of the symbols of  $z_T^s$ . Subsequently, the PRC Channel is first decoded to retrieve the random  $seed$ , which generates  $K'$ .  $K'$  is employed to decrypt the GS Channel, and the final watermark is obtained through soft-decision decoding. In the Third-party Verification scenario,  $s$  includes the user information and its signature, which must be verified after extracting  $s'$ .

embedded in the PRC Channel with the private key  $sk_c$ , i.e.,  $K = \text{PRNG}(\text{H}(\text{seed}||sk_c))$ , where  $\text{H}$  denotes a cryptographic hash function and PRNG is a pseudorandom number generator used to produce  $K$ . Considering the use of PRNG,  $m_e$  follows a uniform distribution, i.e.,  $m_e$  is a random binary bit stream. To be compatible with the PRC Channel, a mapping from 0 to 1 and from 1 to  $-1$  is required. That is,  $m_{gs} = (-1)^{m_e}$ .

3) *Distribution-preserving sampling driven by randomized watermark.*: At this point,  $m_{prc}$  and  $m_{gs}$  are combined into the watermark  $m_z$  according to the channel where  $m_{prc}$  is placed in the first two channels of the latent space, and  $m_{gs}$  is placed in the last two channels.

Since both  $m_{prc}$  and  $m_{gs}$  are pseudorandom,  $m_z$  is also pseudorandom. We will prove this in Sec. IV-D. Once the distribution of  $m_z$  is known, a distribution-preserving sampling can be performed.

When each dimension represents  $v$ -bit randomized watermark  $m_z$ , these  $v$  bits can be regarded as an integer  $y \in [0, 2^v - 1]$ , where we treat  $-1$  as binary 1 and 1 as binary 0. Since  $m_z$  is pseudorandom,  $y$  follows a discrete uniform distribution, i.e.,  $p(y) = \frac{1}{2^v}$  for  $y = 0, 1, 2, \dots, 2^v - 1$ . Let  $f(x)$  denote the probability density function of the Gaussian distribution  $\mathcal{N}(0, I)$ , and  $\text{ppf}(\cdot)$  denotes the quantile function. We divide  $f(x)$  into  $2^v$  equal cumulative probability portions. When  $y = i$ , the watermarked latent representation  $z_T^s$  falls into the  $i$ -th interval, which means  $z_T^s$  should follow the conditional distribution:

$$p(z_T^s|y=i) = \begin{cases} 2^v \cdot f(z_T^s) & \text{ppf}(\frac{i}{2^v}) < z_T^s \leq \text{ppf}(\frac{i+1}{2^v}) \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

The probability distribution of  $z_T^s$  is given by:

$$p(z_T^s) = \sum_{i=0}^{2^v-1} p(z_T^s|y=i)p(y=i) = f(z_T^s). \quad (5)$$

Eq. 5 indicates that  $z_T^s$  follows the same distribution as the randomly sampled latent representation  $z_T \sim \mathcal{N}(0, I)$ . Next, we elaborate on how this sampling is implemented.

Let the cumulative distribution function of  $\mathcal{N}(0, I)$  be denoted as  $\text{cdf}(\cdot)$ . We can obtain the cumulative distribution function of Eq. 4 as follows,

$$F(z_T^s|y=i) = \begin{cases} 0 & z_T^s < \text{ppf}(\frac{i}{2^v}) \\ 2^v \cdot \text{cdf}(z_T^s) - i & \text{ppf}(\frac{i}{2^v}) \leq z_T^s \leq \text{ppf}(\frac{i+1}{2^v}) \\ 1 & z_T^s > \text{ppf}(\frac{i+1}{2^v}) \end{cases} \quad (6)$$

Given  $y = i$ , we aim to perform random sampling of  $z_T^s$  within the interval  $[\text{ppf}(\frac{i}{2^v}), \text{ppf}(\frac{i+1}{2^v})]$ . The commonly used method is rejection sampling [63]–[65], which can be time-consuming as it requires repeated sampling until  $z_T^s$  falls into the correct interval. Instead, we can utilize the cumulative probability density. When randomly sampling  $F(z_T^s|y=i)$ , the corresponding  $z_T^s$  is naturally obtained through random sampling. Since  $F(z_T^s|y=i)$  takes values in  $[0, 1]$ , sampling from it is equivalent to sampling from a standard uniform distribution, denoted as  $u = F(z_T^s|y=i) \sim \mathcal{U}(0, 1)$ . Shift the terms of Eq. 6, and take into account that  $\text{cdf}$  and  $\text{ppf}$  are inverse functions, we have

$$z_T^s = \text{ppf}(\frac{u+i}{2^v}). \quad (7)$$

Eq. 7 represents the process of sampling the watermarked latent representation  $z_T^s$  driven by the randomized watermark  $m_z$ .

4) *Image Generation*: After the sampling process, the watermark is embedded in the latent representation  $z_T^s$ , and the subsequent generation process is no different from the regular generation process of SD. Here, we utilize the commonly adopted DPMSolver [40] for iterative denoising of  $z_T^s$ . After obtaining denoised  $z_0^s$ , the watermarked image  $X^s$  is generated using the decoder  $\mathcal{D}$ :  $X^s = \mathcal{D}(z_0^s)$ .

### C. Watermark Extraction

In the Operator Verification scenario, watermark extraction is performed by the operator, and the verification result is made publicly available. In the Third-party Verification scenario, the operator is required to disclose the model inversion capability, allowing any third party to perform watermark extraction. The specific process is as follows:

1) *Inversion and Posterior Estimation*: Using the encoder  $\mathcal{E}$ , we first restore  $X'^s$  to the latent space  $z_0'^s = \mathcal{E}(X'^s)$ . Imprecise latent representation recovery can significantly reduce the effectiveness of watermark extraction. To address this, we employ the more precise Exact Inversion method [31] to estimate the additive noise. After a sufficient number of inversion steps,  $z_T'^s$  can be considered approximately equal to  $z_T^s$  within an acceptable error margin.

After obtaining the inversion result  $z_T'^s$ , we follow the approach of Gunn et al. [29] and model the entire image generation and inversion process as a noisy channel. Specifically, the concatenated codeword  $m_z = m_{prc} || m_{gs} \in \{-1, 1\}^{ch \times h \times w}$  is viewed as passing through an AWGN Channel. The noise strength is characterized by  $\sigma = \sqrt{3/2}$ , and the posterior estimates  $m'_z = m'_{prc} || m'_{gs}$  are obtained via the error function (erf) corresponding to the AWGN Channel, that is:

$$m'_z \triangleq \mathbb{E}[m | z_T'^s] = \text{erf}\left(\frac{z_T'^s}{\sqrt{2\sigma^2(1+\sigma^2)}}\right) \quad (8)$$

After performing posterior estimation to obtain  $m'_z$ , we evenly split  $m'_z$  along the channel dimension into two components: the PRC Channel watermark header  $m'_{prc}$  and the GS Channel watermark information  $m'_{gs}$ , which are then processed separately.

2) *PRC Channel*: Assuming the LDPC key is given by  $(P, G)$ , we employ the belief propagation with ordered statistics decoding (BP-OSD) [66], [67] to recover the estimated seed  $n'_{seed} : n'_{seed} = \text{BP-OSD}(P, m'_{prc})$ . This recovered seed is then used to initialize the PRNG within the GS Channel for stream cipher regeneration during watermark extraction.

3) *GS Channel*: For the GS Channel watermark information  $m'_{gs}$ , decryption is performed via element-wise multiplication with the stream key  $K' = \text{PRNG}(\text{H}(seed' || sk_c))$ , since the following equation holds:

$$s'^d \triangleq \mathbb{E}\left[(-1)^{s^d} | z_T'^s\right] = (-1)^{K'} \cdot m'_{gs}, \quad (9)$$

where  $s'^d \in [-1, 1]$  is the posterior expectation of the repeated message  $(-1)^{s^d}$ . To obtain the maximum a posteriori (MAP)

decoding of the original message  $s$ , the problem essentially reduces to decoding a repetition code (REP code [32]).

For each bit  $s_j$  of the watermark information  $s$ , the log-likelihood ratio (LLR) of its  $i$ -th repetition  $s_{i,j}^d$  can be computed as follows:

$$LLR_{i,j} = \log \frac{\Pr\left((-1)^{s_{i,j}^d} = 1 | z_T'^s\right)}{\Pr\left((-1)^{s_{i,j}^d} = -1 | z_T'^s\right)}. \quad (10)$$

Since  $s_{i,j}^{td} \triangleq \mathbb{E}\left[(-1)^{s_{i,j}^d} | z_T'^s\right] = \Pr\left((-1)^{s_{i,j}^d} = 1 | z_T'^s\right) - \Pr\left((-1)^{s_{i,j}^d} = -1 | z_T'^s\right)$ , it is easy to derive that:

$$s_{i,j}^{td} = \frac{e^{LLR_{i,j}} - 1}{e^{LLR_{i,j}} + 1} = \tanh\left(\frac{LLR_{i,j}}{2}\right). \quad (11)$$

For each bit  $s_j$  of the watermark information  $s$ , due to the independence of repeated observations, the total log-likelihood ratio  $LLR_{\text{total},j}$  for a repetition count of  $num = \frac{ch \times h \times w}{2f_{ch} \cdot f_{hw}^2}$  can be calculated by:

$$\begin{aligned} LLR_{\text{total},j} &\triangleq \log \frac{\Pr\left((-1)^{s_j} = 1 | z_T'^s\right)}{\Pr\left((-1)^{s_j} = -1 | z_T'^s\right)} \\ &= \sum_{i=1}^{num} LLR_{i,j} = \sum_{i=1}^{num} 2 \cdot \text{arctanh}(s_{i,j}^{td}), \end{aligned} \quad (12)$$

Therefore, the final estimated  $s'_j$  of  $s_j$  can be calculated by:

$$s'_j = \text{sign}(LLR_{\text{total},j}). \quad (13)$$

In practice, we use the first-order approximation of the arctanh function for computational efficiency, i.e.,  $LLR_{i,j} \simeq 2s_{i,j}^{td}$ . The final estimated  $s'_j$  is then given by:

$$s'_j = \text{sign}\left(\sum_{i=1}^{num} 2s_{i,j}^{td}\right) = \text{sign}\left(\sum_{i=1}^{num} s_{i,j}^{td}\right). \quad (14)$$

The estimation method (Eq. 14) shows negligible performance difference compared to the optimal scheme (Eq. 13) that accumulates full LLRs.

In the Third-party Verification scenario, after the watermark is extracted, the user information and signature need to be verified using the public verification key  $VK$ .

### D. Proof of Lossless Performance

To demonstrate that our proposed hybrid watermarking scheme is performance-lossless, a sufficient condition is to prove that the embedded ciphertext  $m_{prc} || m_{gs}$  satisfies the IND\$-CPA security property—i.e., it is computationally indistinguishable from a random bitstring under chosen-plaintext attacks. This suffices because the sampling process we adopt is distribution-preserving: as long as the ciphertext driving the sampling is IND\$-CPA secure, the resulting watermarked image remains computationally indistinguishable from a randomly sampled image under chosen-plaintext attacks [68].

In the following, we present the formal definition of IND\$-CPA security, and subsequently provide a proof that the ciphertext  $m_{prc} || m_{gs}$  used in our scheme satisfies this security notion.



**Definition 1. (Chosen Plaintext Attack)**

Consider a symmetric encryption scheme with a key tuple  $K_{CS} = (P, G, sk_c)$ , which produces ciphertexts of the form:

$$(m_{prc} \parallel m_{gs})_{s^d} \triangleq (G \cdot seed \oplus e) \parallel (\text{PRNG}(\text{H}(seed \parallel sk_c)) \oplus s^d),$$

where  $s^d$  denotes the plaintext to be encrypted, and  $H : \{0, 1\}^* \rightarrow \mathbb{F}_2^k$  is a hash function. The LDPC key pair  $(P, G)$  is generated via the randomized algorithm  $\text{KeyGen}(l(k), k, t, l'(k))$  and  $sk_c$  is randomly sampled with a length of  $k$  bits, all parameters depend on the security parameter  $k$ , except for the constant  $t$ .

We define a chosen-plaintext attack (CPA) game between an adversary  $\mathcal{A}$  and a challenger as follows:

- **Key generation stage.**  $K_{CS} = (P, G, sk_c) \leftarrow (\text{KeyGen}(l(k), k, t, l'(k)), 1^k)$ .
- **Learning stage.**  $\mathcal{A}$  sends plaintext  $s_A^d$  to the oracle and returns  $(m_{prc} \parallel m_{gs})_{s_A^d}$ .  $\mathcal{A}$  can perform this stage multiple times.
- **Challenge stage.**  $\mathcal{A}$  sends plaintext  $s^d \in \mathcal{M} \setminus \{s_A^d\}$  to the oracle, which will flip a coin  $b \in \{0, 1\}$ . If  $b = 0$ ,  $\mathcal{A}$  obtains  $m_z = (m_{prc} \parallel m_{gs})_{s^d}$ ; If  $b = 1$ ,  $\mathcal{A}$  obtains  $u \leftarrow U_{(m_{prc} \parallel m_{gs})_{s^d}}$ .
- **Guessing stage.**  $\mathcal{A}$  output a bit  $b'$  as a “guess” about whether it obtains a plaintext or a random string.

Define the Chosen Plaintext Attack (CPA) advantage of  $\mathcal{A}$  against the scheme by:

$$\text{Adv}_{CS}^{\text{cpa}}(\mathcal{A}, k) \triangleq \left| \Pr_{K_{CS}} [\mathcal{A}(m_z) = 1] - \Pr_{K_{CS}} [\mathcal{A}(u) = 1] \right|. \quad (15)$$

The cryptographic scheme is **indistinguishable from uniformly random bits under chosen plaintext attack (IND\$-CPA)** if  $\text{InSec}_{CS}^{\text{cpa}}(t, l, k) \triangleq \max_{\mathcal{A} \in \mathcal{A}(t, l)} \{\text{Adv}_{CS}^{\text{cpa}}(\mathcal{A}, k)\}$  is negligible in  $k$ .

**Theorem 1.** Let the sparsity parameter be set as  $t = \Theta(\log l(k))$ , and suppose that each encryption execution samples a fresh seed uniformly at random from  $\mathbb{F}_2^k$ . Assume that the pseudorandom generator PRNG satisfies standard pseudorandomness under the security parameter  $k$ , and that the hash function  $H$  is modeled as a **random oracle**. Then the resulting ciphertext  $(m_{prc} \parallel m_{gs})_{s^d}$  is computationally indistinguishable from a uniformly random bitstring under a chosen-plaintext attack; that is, the scheme satisfies IND\$-CPA security.

*Proof.* Define  $H_0 \triangleq (m_{prc} \parallel m_{gs})_{s^d} = (G \cdot seed \oplus e) \parallel (\text{PRNG}(\text{H}(seed \parallel sk_c)) \oplus s^d)$ .

Define  $H_1$  as the variant of  $H_0$  where  $\text{H}(seed \parallel sk_c)$  is replaced by a random element of  $\mathbb{F}_2^k$ , i.e.  $H_1 \triangleq (G \cdot seed \oplus e) \parallel (\text{PRNG}(r_1) \oplus s^d)$ .

Define  $H_2$  as the variant of  $H_1$  where  $(G \cdot seed \oplus e)$  is replaced by a random element of  $\{0, 1\}^{l(k)}$ , i.e.  $H_2 \triangleq r_2 \parallel (\text{PRNG}(r_1) \oplus s^d)$ .

Define  $H_3$  as the variant of  $H_2$  where  $\text{PRNG}(r_1)$  is replaced by a random element of  $\{0, 1\}^{|s^d|}$ , i.e.  $H_3 \triangleq r_2 \parallel (r_3 \oplus s^d)$ .

As shown in Fig. 4, we claim that the advantage of distinguishing between  $H_0$  and  $H_1$ ,  $H_1$  and  $H_2$ ,  $H_2$  and  $H_3$ , as well as  $H_3$  and random bits, are all negligible in  $k$ .



Fig. 4. The hardness of distinguishing between  $H_0$  and  $H_3$ .

(1) The advantage of distinguishing  $H_0$  from  $H_1$  is negligible if the hash function  $H$  is modeled as a random oracle and the assumption that  $sk_c$  remains secret. Specifically, since the adversary does not know  $sk_c$ , and  $seed$  is freshly sampled for each encryption, the value  $seed \parallel sk_c$  is unpredictable to the adversary.

In the random oracle model, unless the adversary queries the oracle at exactly  $seed \parallel sk_c$ , the output  $\text{H}(seed \parallel sk_c)$  remains statistically independent of all previously seen values. Since  $sk_c$  is a  $k$ -bit secret key, each query hits the correct input with probability at most  $2^{-k}$ , making the overall success probability negligible in  $k$ . Therefore, replacing  $\text{H}(seed \parallel sk_c)$  with a truly random string  $r_1 \leftarrow \mathbb{F}_2^k$  results in a distribution that is computationally indistinguishable from the original.

(2) Distinguishing  $H_1$  from  $H_2$  would contradict the pseudorandomness of the PRC construction. Given that the sparsity parameter is set as  $t = \Theta(\log l(k))$ , and each encryption execution samples a fresh seed uniformly at random from  $\mathbb{F}_2^k$ , the term  $G \cdot seed \oplus e$  constitutes an LDPC codes-based PRC codeword. As shown in Sec. II-E, the pseudorandomness of this construction is guaranteed either by the subexponential LPN assumption, or by the standard LPN assumption in combination with the planted XOR assumption.

Moreover, since each PRC codeword is generated using an independently sampled random seed, ciphertexts corresponding to different queries are statistically independent. This ensures that the construction satisfies chosen-plaintext attack (CPA) security under the assumed hardness of the underlying PRC.

(3) Distinguishing  $H_2$  from  $H_3$  would contradict the security of the PRNG. By assumption, the PRNG satisfies standard pseudorandomness under the security parameter  $k$ ; that is, for a uniformly sampled seed  $r_1 \in \{0, 1\}^k$ , the output  $\text{PRNG}(r_1)$  is computationally indistinguishable from a uniformly random bitstring of the same length. Therefore, if an adversary could distinguish  $H_2$  from  $H_3$ , it would imply a distinguisher against the PRNG.

In addition, since each encryption instance uses a freshly sampled seed, the input to the PRNG is independent across queries. Together with the pseudorandomness of PRNG, this ensures that ciphertexts are CPA-secure even under multiple chosen plaintexts.

(4) Distinguishing  $H_3$  from a uniformly random bitstring is information-theoretically impossible, as the distribution of  $H_3$  is identical to the uniform distribution over  $\{0, 1\}^{l(k)+|s^d|}$ .

By a sequence of hybrid arguments, we have shown that the ciphertext generated by the scheme is computationally indistinguishable from a uniformly random bitstring under a chosen-plaintext attack. Therefore, the scheme satisfies IND\$-CPA security under the random oracle model, assuming the pseudorandomness of the PRNG, the pseudorandomness of

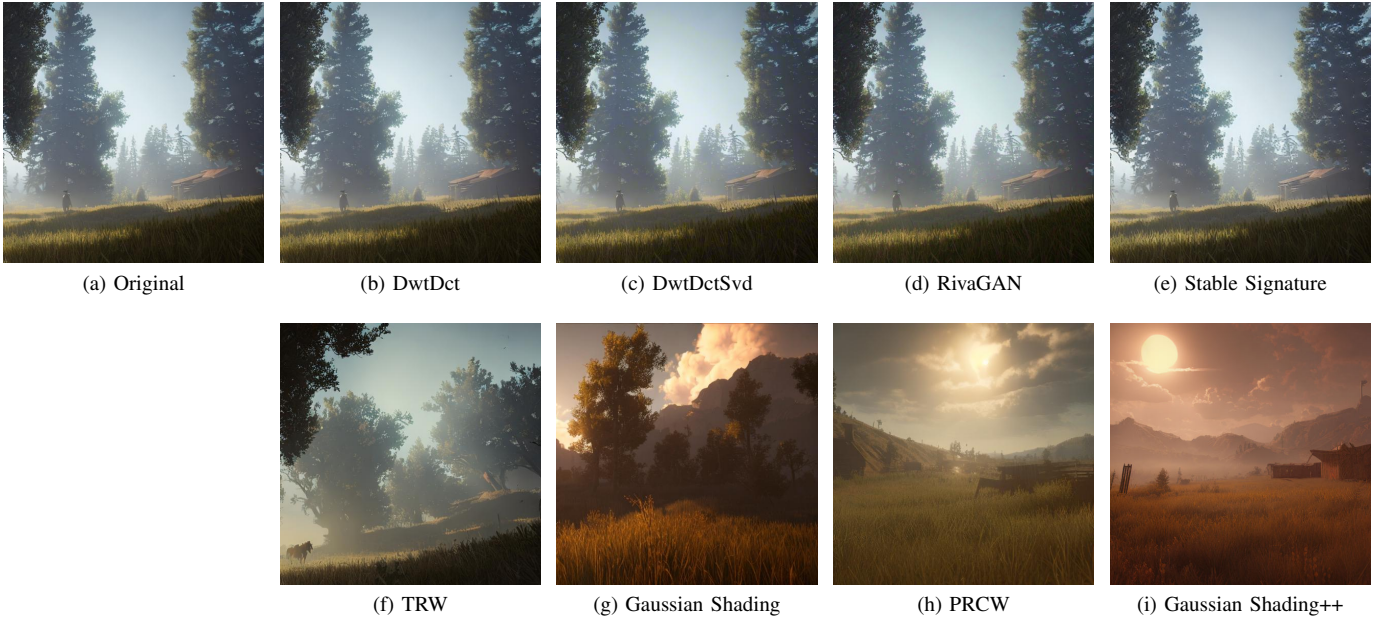


Fig. 5. Watermarked images generated using different watermarking methods with the same prompt: “Red dead redemption 2, cinematic view, epic sky, detailed, concept art, low angle, high detail, warm lighting, volumetric, godrays, vivid, beautiful, trending on artstation, by jordan grimmer, huge scene, grass, art greg rutkowski.”. Among them, post-processing-based methods (b) (c) (d) and the fine-tuning-based method (e) only add image residuals compared to the original image (a).

the PRC, and the secrecy of the private key  $sk_c$ .  $\square$

**Remark 1.** As demonstrated above, our hybrid watermarking scheme inherits the pseudorandomness guarantees established in prior theoretical work. This represents a conceptual advancement over PRCW [28], whose watermark ciphertext takes the form  $G \cdot (\text{testbits} \parallel m \parallel r) \oplus e$ . Since testbits and  $m$  are not randomly sampled during use, PRCW cannot theoretically reduce its performance-lossless property to the hardness assumptions underlying LDPC codes-based PRC. In practical scenarios where  $m$  remains fixed, the pseudorandomness of PRCW relies solely on  $r$ , making it potentially vulnerable under large sample sizes. This limitation in pseudorandomness may be empirically observed, as discussed in Sec. V-C4.

This proof above is conducted in the context of **Operator Verification**, where our watermarking scheme is shown to be provably performance-lossless. When extended to the **Third-party Verification** setting, we no longer consider the undetectability of the watermark compared to the unwatermarked image, since public verifiability inherently conflicts with undetectability. This is because undetectability requires that no polynomial-time algorithm can distinguish between the two, whereas in the **Third-party Verification** setting, the watermark extraction algorithm is publicly available and can naturally be used for detection.

## V. EXPERIMENTS

This section presents the experimental analysis. We begin by evaluating the performance of Gaussian Shading++ in both the Operator Verification and Third-party Verification scenarios. Furthermore, we conduct comprehensive comparisons with

several state-of-the-art methods. Lastly, to validate the effectiveness of each component within Gaussian Shading++, we perform thorough ablation studies.

### A. Implementation Details

1) *Diffusion Models*: In this paper, we focus on text-to-image latent diffusion models, hence we select the Stable Diffusion (SD) [6] provided by huggingface. We evaluate Gaussian Shading++ as well as baseline methods, using SD V2.1. The size of the generated images is  $512 \times 512$ , and the latent space dimension is  $4 \times 64 \times 64$ . During inference, we adopt prompts from the Stable-Diffusion-Prompt (SDP) dataset<sup>3</sup>, using a guidance scale of 7.5 (the default setting in SD). Image generation is performed with 50 sampling steps using DPMSolver [40]. Considering that users typically share generated images without preserving the original prompts, we perform 50 steps of inversion using Exact Inversion [31] with an empty prompt and a guidance scale of 3 (the default setting in Exact Inversion [31]).

2) *Watermark Methods*: In the main experiments, The PRC Channel of Gaussian Shading++ encodes a 32-bit seed using the PRC. For the GS Channel, we set the parameters as  $f_{ch} = 2$ ,  $f_{hw} = 4$ ,  $v = 1$ , resulting in an actual capacity of 256 bits. In the Third-party Verification scenario, we employ the public-key signature scheme ECDSA [34], the 256-bit watermark is further split into 32 bits of user information and 224 bits of signature.

For comparison, we select three representative categories of watermark methods as baselines:

<sup>3</sup>Stable-Diffusion-Prompts

- Post-processing-based: We adopt three officially used by SD, which embed specific patterns in either the frequency or spatial domain: DwtDct [8], DwtDctSvd [8], and RivaGAN [9]. The capacities of DwtDct [8] and DwtDctSvd [8] are set to 256 bits. Since RivaGAN [9] supports a maximum capacity of 32 bits, we retain this setting.
- Fine-tuning-based: We adopt Stable Signature [21], which injects watermark information by fine-tuning the VAE decoder of Stable Diffusion, ensuring the watermark is embedded in the generated images. Due to convergence issues caused by large watermark capacities, we instead use the official open-source model of Stable Signature with a capacity of 48 bits<sup>4</sup>. During fine-tuning, we use 400 images from the ImageNet2014 [69] validation set, with a batch size of 4 and 100 training steps.
- Latent-representation-based: We adopt TRW [26], Gaussian Shading [27], and PRCW [28]. TRW [26] embeds the watermark by modifying latent representations to align with specific patterns. As it is a single-bit watermark, we evaluate it only in the detection task. Since its Rand mode better aligns with the notion of performance-lossless watermark, we adopt this setting. Gaussian Shading [27] uses a stream cipher to pseudorandomize the watermark. However, considering the challenges of key management in practical deployment, the stream cipher is fixed during our experiments. PRCW [28] encodes the watermark using pseudorandom error-correcting codes. For a fair comparison, the watermark capacities of both Gaussian Shading [27] and PRCW [28] are set to 256 bits.

Examples of images from all of the above watermarking methods are shown in Fig. 5.

3) *Robustness Evaluation*: We evaluate the robustness of the methods from two perspectives: traditional noise distortions and removal attack [70]. For traditional distortions, we select six representative types of noise: (a) JPEG Compression,  $QF = 70$  (JPEG). (b) Brightness,  $factor = 1$ . (c) Gaussian Blur,  $radius = 3$  (GauBlur). (d) Gaussian Noise,  $\mu = 0$ ,  $\sigma = 0.01$  (GauNoise). (e) Median Filtering,  $kernel\_size = 7$  (MedFilter). (f) 50% Resize and restore (Resize). For removal attack, we adopt Variational AutoEncoder (VAE) [6], [35]–[37], and Stable Diffusion (SD) [6] as erasure networks.

4) *Evaluation Metrics*: In the Operator Verification scenario, we focus on two primary tasks: detection and traceability. For detection, we compute the bit accuracy threshold corresponding to a fixed false positive rate (FPR) based on Eq. 2, and then report the true positive rate (TPR) on watermarked images. For traceability, we directly evaluate the bit accuracy of watermark extraction. In the Third-party Verification scenario, we evaluate the accuracy of traceability, measuring the success rate of accurately tracing the watermark back to the target user.

Apart from the above metrics, we assess the impact of each method on model performance from two perspectives: visual quality and distribution of latent representations. For visual quality, we employ Fréchet Inception Distance (FID) [71] and

CLIP-Score [72] to measure the realism and text-image alignment of watermarked images, respectively. Specifically, we compute FID and CLIP-Score over 10 batches of watermarked images and conduct a  $t$ -test comparing the mean values against those of non-watermarked images. A smaller  $t$ -value indicates less performance degradation caused by watermarking. For the distribution of latent representations, which mainly targets latent-representation-based methods, we generate 80,000 latent representations and model them as samples from a standard Gaussian distribution. We then perform hypothesis testing, including the K-S test, Shapiro-Wilk test, and other statistical methods, to compute the  $p$ -value. A higher  $p$ -value indicates a closer match to the standard Gaussian distribution.

All experiments are conducted using the PyTorch 2.1.0 framework, running on a single RTX A6000 GPU.

## B. Performance of Gaussian Shading++

1) *Operator Verification*: To enable detection, we consider Gaussian Shading++ as a single-bit watermark, with a fixed watermark  $s$ . We approximate the FPR to be controlled at  $10^0, 10^{-1}, \dots, 10^{-13}$ , calculate the corresponding threshold  $\tau$ , and test the TPR on 500 watermarked images. As shown in Fig. 6a, when the FPR is controlled at  $10^{-13}$ , the TPR remains at least 0.97 for five out of the seven cases. Although the TPR for Gaussian Noise is only 0.918, it is still a promising result.

For traceability, Gaussian Shading++ serves as a multi-bit watermark. Assuming Alice provides services to  $N$  users, Alice needs to allocate one watermark for each user. In our experiments, we assume that  $N' = 1,000$  users generate images, with each user generating 5 images, resulting in a dataset of 5,000 watermarked images.

During testing, we calculate the threshold  $\tau$  to control the FPR at  $10^{-6}$ . Note that when computing traceability accuracy, we need to consider two types of errors: false positives, where watermarked images are not detected, and traceability errors, where watermarked images are detected but attributed to the wrong user. Therefore, we first determine whether the image contains a watermark. If it does, we calculate the number of matching bits  $Acc$  with all  $N$  users on the platform. The user with the highest  $Acc$  is considered the one who generated the image. Finally, we verify whether the correct user has been traced. When  $N > N'$ , it can be assumed that some users have been assigned a watermark but have not generated any images.

As shown in Fig. 6b, when  $N = 10^6$ , Gaussian Shading++ achieves a traceability accuracy of over 96% in six cases. Although the traceability accuracy for Gaussian Noise is only 92.26%, if a user generates two images, the probability of successfully tracing him is still no less than 99%.

2) *Third-party Verification*: In this scenario, any third party can perform traceability on the target user using the watermark database publicly provided by the operator. Since the public-key signature is unforgeable, the probability that a randomly generated signature passes verification is negligible. Therefore, the traceability accuracy is effectively equivalent to the signature verification success rate. We generate 500 images and evaluate the traceability accuracy of Gaussian Shading++

<sup>4</sup>The GitHub Repository for Stable Signature

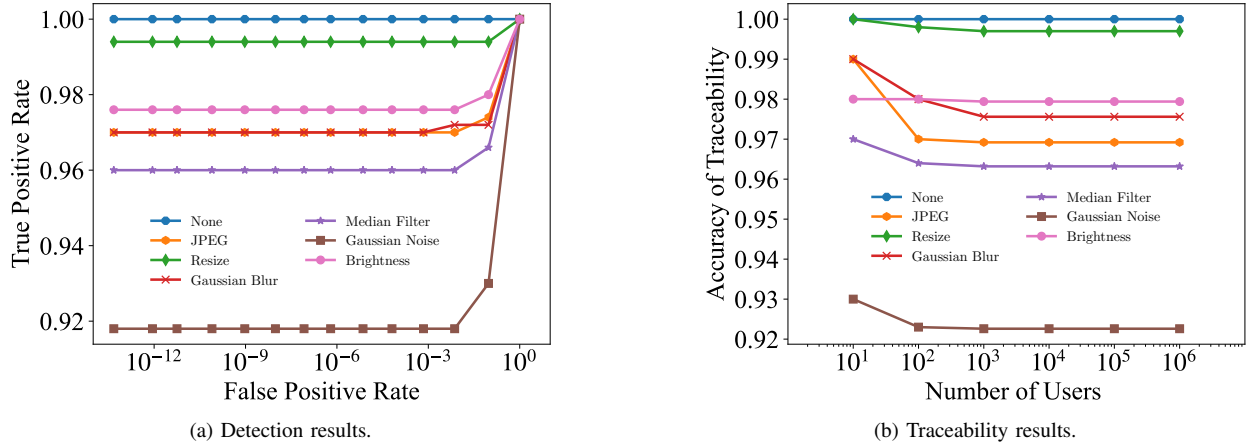


Fig. 6. Performance of Gaussian Shading++ in Operator Verification scenario. The results are presented separately for detection and traceability tasks.

under various traditional noise distortions, both before and after incorporating ECDSA. As shown in Tab. I, after introducing the public-key signature, Gaussian Shading++ experiences a significant performance drop when facing filtering-based distortions such as Gaussian Blur and Median Filter. However, it still maintains robust performance under other conditions, with traceability accuracy remaining above 70%.

TABLE I  
TRACEABILITY ACCURACY OF GAUSSIAN SHADING++. “OURS” DENOTES THE NAIVE VERSION, WHILE “OURS + ECDSA” REPRESENTS THE VERSION WITH THE PUBLIC-KEY SIGNATURE.

Noise	Methods	
	Ours	Ours + ECDSA
None	1.000	0.994
JPEG	0.972	0.722
Brightness	0.980	0.896
GauBlur	0.976	0.158
GauNoise	0.922	0.714
MedFilter	0.964	0.358
Resize	0.996	0.870

After the operator publicly releases the model inversion capability, malicious users can reuse latent representations and generate forged images with illicit prompts on a proxy model to falsely frame target users, thereby launching a reprompt attack (“Attk”) [33]. Moreover, since the sign of the latent representations in Gaussian Shading++ encodes the watermark, attackers can enhance the attack (“Attk+”) by resampling within specific intervals corresponding to the watermark bits.

In our experiments, we consider five potential proxy models that attackers might use: SD V1.4, SD V1.5, SD V2.0, SD V2.1 [6], and SD-XL [43]. The first 500 prompts from the I2P [73] dataset are used as illicit prompts. For the “Attk” scenario, we use SD V2.1 to generate 500 forged images. For the enhanced “Attk+” scenario, we perform three resampling attempts for each illicit prompt, resulting in a total of 1,500 forged images. We compared the attack success rate based on the use of ECDSA, and the results are presented in Tab. II. It is evident that introducing the public-key signature signifi-

cantly reduces the risk of Gaussian Shading++ being forged, particularly when the proxy models are SDV1.4 and SDV1.5. Since SD-XL has a model architecture and parameters that are substantially different from SD V2.1, forgery is difficult to achieve. However, due to the similarity in parameters between SD V2.0 and SD V2.1, these two proxy models can easily perform forgery, which further emphasizes the importance of the operator safeguarding the model parameters.

Overall, although introducing ECDSA may compromise some robustness, it significantly reduces the risk of successful forgery when attackers cannot access precise model parameters. This trade-off is worthwhile when extending the functionality to the Third-party Verification scenario.

TABLE II  
ATTACK SUCCESS RATE OF REPROMPT ATTACKS (“ATTK” / “ATTK+”) ON GAUSSIAN SHADING++. “OURS” DENOTES THE NAIVE VERSION, WHILE “OURS + ECDSA” REPRESENTS THE VERSION WITH THE PUBLIC-KEY SIGNATURE.

Proxy Model	Methods	
	Ours	Ours + ECDSA
SD V1.4	0.894 / 0.937	<b>0.483 / 0.523</b>
SD V1.5	0.886 / 0.937	<b>0.362 / 0.433</b>
SD V2.0	0.941 / 0.985	<b>0.832 / 0.889</b>
SD V2.1	0.944 / 0.985	<b>0.839 / 0.889</b>
SD-XL	<b>0.000 / 0.000</b>	<b>0.000 / 0.000</b>

### C. Comparison to Baselines

In this section, we compare the performance of Gaussian Shading++ with baselines on SD V2.1. Robustness is evaluated from two aspects: traditional noise distortions and removal attacks. Additionally, performance-lossless characteristic is demonstrated in terms of visual quality and the distribution of latent representations. Finally, we compare PRCW [28] by evaluating the performance under varying *guidance\_scale*.

1) *Robustness to Noise*: We conduct tests on 500 generated images for each method respectively. As shown in Tab. III, Gaussian Shading++ achieves near-optimal performance under various noise conditions, second only to Gaussian Shading [27]. It outperforms the state-of-the-art method



TABLE III

COMPARISON OF ROBUSTNESS UNDER TRADITIONAL NOISE DISTORTIONS. WE CONTROL THE FPR AT  $10^{-6}$ , AND EVALUATE THE TPR / BIT ACCURACY FOR SD V2.1. **BOLD** REPRESENTS THE BEST, UNDERLINE REPRESENTS THE SECOND BEST.

Methods	Noise						
	None	JPEG	Brightness	GauBlur	GauNoise	MedFilter	Resize
DwtDct [8]	0.832 / 0.790	0.000 / 0.509	0.292 / 0.579	0.002 / 0.509	0.814 / 0.770	0.000 / 0.521	0.218 / 0.594
DwtDctSvd [8]	<b>1.000 / 0.999</b>	0.998 / 0.870	0.186 / 0.503	0.904 / 0.771	<u>0.998 / 0.979</u>	0.998 / 0.937	<u>0.996 / 0.979</u>
RivaGAN [9]	<u>0.970 / 0.993</u>	0.844 / 0.964	0.764 / 0.936	0.718 / 0.942	0.738 / 0.933	0.914 / 0.970	0.940 / 0.986
Stable Signature [21]	<b>1.000 / 0.998</b>	0.856 / 0.889	0.886 / 0.937	0.000 / 0.413	0.948 / 0.973	0.000 / 0.647	0.332 / 0.806
TRW [26]	<b>1.000 / -</b>	<b>1.000 / -</b>	0.906 / -	<b>1.000 / -</b>	0.730 / -	<b>1.000 / -</b>	<b>1.000 / -</b>
Gaussian Shading [27]	<b>1.000 / 1.000</b>	<b>1.000 / 0.999</b>	<b>0.998 / 0.999</b>	<b>1.000 / 0.998</b>	<b>1.000 / 0.998</b>	<b>1.000 / 0.998</b>	<b>1.000 / 1.000</b>
PRCW [28]	<b>1.000 / 1.000</b>	0.950 / 0.972	0.964 / 0.984	0.846 / 0.933	0.878 / 0.943	0.886 / 0.943	0.988 / 0.992
<b>Gaussian Shading++</b>	<b>1.000 / 1.000</b>	0.974 / <u>0.984</u>	<u>0.974 / 0.986</u>	<u>0.974 / 0.972</u>	0.918 / 0.956	0.964 / <u>0.974</u>	<u>0.996 / 0.996</u>

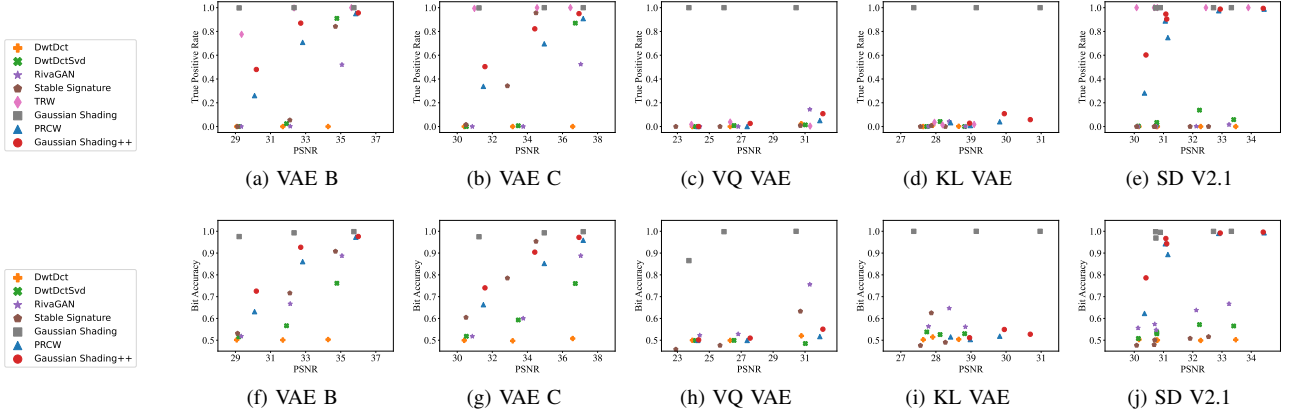


Fig. 7. Comparison of robustness under removal attacks. The first row presents the TPR of watermark methods, while the second row shows their bit accuracy. For VAE B [35] and VAE C [36], we select three strength levels  $quality = 2, 4, 6$ . For VQ VAE [37] and KL VAE [6], we choose three strength levels  $f = 4, 8, 16$ . For SD V2.1 [6], we set five removal steps  $t_{step} = 10, 25, 50, 100, 200$ .

PRCW [28], and even improves bit accuracy by up to 4% under Gaussian Blur. This is because, on the one hand, the insufficient robustness of the PRC Channel leads to decoding failure of the *seed*, and the fewer watermark copies in the GS Channel compared to Gaussian Shading [27] result in reduced robustness. On the other hand, the GS Channel in Gaussian Shading++ exhibits stronger robustness than PRCW [28] when the PRC Channel successfully decodes. As a result, the overall performance of Gaussian Shading++ is between that of Gaussian Shading [27] and PRCW [28]. Considering the challenges of key management in Gaussian Shading [27], Gaussian Shading++ is the optimal solution for practical deployment.

2) *Removal Attack*: Besides traditional noise distortions, recent studies [70], [74] emphasize the necessity for watermarking methods to resist removal attacks. The main idea is to reconstruct watermarked images through networks to erase watermark signals. We consider two types of removal networks and evaluate five methods: VAE (VAE B [35], VAE C [36], VQ VAE [37], KL VAE [6]), and SD V2.1 [6]. For each watermarking method, we evaluate 500 images under different levels of each removal attack. The experimental results are shown in Fig. 7. Post-processing-based [8], [9] and fine-tuning-based [21] methods struggle to resist removal attacks since their watermark signals are encoded in the residuals. For latent-representation-based methods, Gaussian Shading [27]

demonstrates strong robustness against all removal attacks, whereas TRW [26], PRCW [28], and Gaussian Shading++ struggle to resist VAE-based removal attacks. This is because, although the GS Channel exhibits strong robustness, the PRC Channel consistently decodes an incorrect *seed*, leading to extraction errors in the GS Channel.

TABLE IV  
THE  $t$ -TEST RESULTS FOR VISUAL QUALITY ON SD V2.1. **BOLD** REPRESENTS THE BEST, UNDERLINE REPRESENTS THE SECOND BEST.

Methods	Metrics	
	FID ( $t$ -value $\downarrow$ )	CLIP-Score ( $t$ -value $\downarrow$ )
Stable Diffusion	25.23 $\pm$ .18	0.3629 $\pm$ .0006
DwtDct [8]	24.97 $\pm$ .19 (3.026)	0.3617 $\pm$ .0007 (3.045)
DwtDctSvd [8]	24.45 $\pm$ .22 (8.253)	0.3609 $\pm$ .0009 (4.452)
RivaGAN [9]	24.24 $\pm$ .16 (12.29)	0.3611 $\pm$ .0009 (4.259)
Stable Signature [21]	25.45 $\pm$ .18 (2.477)	0.3622 $\pm$ .0027 ( <u>0.7066</u> )
TRW [26]	25.43 $\pm$ .13 (2.581)	0.3632 $\pm$ .0006 (0.8278)
Gaussian Shading [27]	25.15 $\pm$ .16 (1.005)	0.3624 $\pm$ .0006 (1.469)
PRCW [28]	25.22 $\pm$ .15 ( <u>0.1636</u> )	0.3624 $\pm$ .0007 (0.8220)
<b>Gaussian Shading++</b>	25.22 $\pm$ .10 ( <b>0.1597</b> )	0.3626 $\pm$ .0011 ( <b>0.6228</b> )

3) *Performance Bias in Visual Quality*: To assess the performance bias from watermark embedding, we conduct a  $t$ -test. We generate 50,000/10,000 images with SD V2.1 for each watermarking method, split into 10 groups of 5,000/1,000 images. The FID [71]/CLIP-Score [72] is com-

TABLE V

COMPARISON OF LATENT DISTRIBUTIONS UNDER DIFFERENT STATISTICS AND STATISTICAL TESTS. WE HIGHLIGHT CLEARLY ABNORMAL STATISTICS AND FAILED NORMALITY TESTS IN **BOLD**, WHILE UNDERLINE INDICATES TESTS WITH LOW CONFIDENCE (I.E., SMALL P-VALUES CLOSE TO ZERO).

Methods	Statistics and Statistical Tests (Statistic / p-value $\uparrow$ )						
	Mean	Frobenius norm	K-S	Shapiro–Wilk	Cramér–von Mises	Jarque–Bera	D’Agostino’s K-squared
Standard Sampling	-4.4599e-6 / 0.9990	57.9245 / -	1.5989e-5 / 0.8910	0.9999 / 1.0000	0.0212 / 0.9957	0.0435 / 0.9785	0.0435 / 0.9785
Gaussian Shading [27]	<b>0.3989 / 0.0000</b>	<b>183.4432 / -</b>	<b>0.5000 / 0.0000</b>	<b>0.7335 / 0.0000</b>	<b>5.4613e+7 / 0.0000</b>	<b>9.0466e+8 / 0.0000</b>	<b>3.7927e+8 / 0.0000</b>
TRW [26]	-8.6425e-6 / 0.9980	57.9628 / -	<b>0.0003 / 0.0000</b>	0.9999 / 1.0000	<b>60.4934 / 0.0000</b>	0.0055 / 0.9972	0.0055 / 0.9972
PRCW [28]	1.966e-5 / 0.9956	57.9865 / -	2.3074e-5 / 0.4877	0.9999 / 1.0000	0.1636 / 0.3507	<u>5.2902 / 0.0710</u>	<u>5.2904 / 0.0710</u>
<b>Gaussian Shading++</b>	-2.6283e-5 / 0.9941	57.9882 / -	2.2996 / 0.4922	0.9999 / 1.0000	0.1093 / 0.5413	0.8647 / 0.6489	0.8647 / 0.6489

puted for each group, and the mean  $\mu_s$  is obtained. Similarly, we generate 50,000/10,000 non-watermarked images, evaluate FID/CLIP-Score across 10 groups, and calculate  $\mu_0$ . For FID, 5,000 images are sampled from MS-COCO-2017 [75] validation set for score computation. For CLIP-Score, OpenCLIP-ViT-G [76] is used to assess image-text relevance.

If the model performance is maintained, then  $\mu_s$  and  $\mu_0$  should be statistically close to each other. Therefore, the hypotheses are  $H_0 : \mu_s = \mu_0, H_1 : \mu_s \neq \mu_0$ . A lower  $t$ -value suggests a higher likelihood of  $H_0$  holding. If it exceeds a threshold,  $H_0$  is rejected, indicating model performance degradation. As shown in Tab. IV, most existing method [8], [9], [21], [26], [27] significantly impact model generation performance from a statistical perspective. In contrast, Gaussian Shading++ achieves the smallest  $t$ -value for both FID and CLIP-Score, indirectly confirming its performance-lossless characteristic.

4) *Latent Distribution*: To evaluate whether the latent space distribution of our watermarking scheme conforms to the standard Gaussian distribution  $\mathcal{N}(0, I)$ , we generated a set of latent vectors containing a fixed watermark message. We computed the sample mean and the Frobenius norm of the covariance matrix, and conducted five hypothesis tests to assess normality: the Kolmogorov–Smirnov (K-S) test [77], the Shapiro–Wilk test [78], the Cramér–von Mises test [79], the Jarque–Bera test [80], and D’Agostino’s K-squared test [81]. Specifically, the K-S test measures the maximum deviation between the empirical and theoretical cumulative distribution functions; the Shapiro–Wilk test evaluates the correlation between the data and expected normal scores; the Cramér–von Mises test quantifies the integrated squared difference between distributions; and the Jarque–Bera and D’Agostino’s tests assess skewness and kurtosis.

Our baseline methods include: standard Gaussian sampling via ‘torch.randn’, Gaussian shading [27] with a fixed key, TRW [26], and PRCW [28]. For all comparison methods, we fix the same watermark message and generate 80,000 latent vectors of shape [4, 64, 64] for evaluation.

The experimental results in Tab. V show that Gaussian Shading [27], under a fixed key and fixed watermark message, fails all normality tests. This is expected, as the output of its watermark ciphertext is deterministic under these fixed conditions, causing the latent space samples to concentrate in a limited subspace. TRW [26] fails two of the tests—the Kolmogorov–Smirnov and the Cramér–von Mises tests—which is also reasonable, as TRW does not explicitly enforce distribu-

tional preservation in the latent space.

PRCW [28] passes all the tests; however, its  $p$ -values in the Jarque–Bera and D’Agostino’s K-squared tests are relatively low, representing a weak pass. We hypothesize that this is because the watermark ciphertext in PRCW takes the form  $G \cdot (\text{testbits} \parallel m \parallel r) \oplus e$ . When both  $\text{testbits}$  and  $m$  are fixed, the only source of randomness comes from  $r$ , which may be insufficient under large sample sizes and could lead to slight deviations from normality.

In contrast, our proposed method passes all normality tests perfectly. Since the watermark header is computed as  $G \cdot \text{seed} \oplus e$ , and  $\text{seed}$  is truly sampled at random, the resulting pseudorandomness is theoretically guaranteed by the pseudorandomness of the underlying PRC construction.

5) *Performance across Guidance\_scale*: Our experiments have confirmed that Gaussian Shading++ achieves performance-lossless watermarking with a fixed key and enables third-party verifiability. To further validate its practicality, evaluating its performance under varying  $\text{guidance\_scale}$  is necessary. We primarily compare Gaussian Shading++ with PRCW [28]. For each value of  $\text{guidance\_scale}$  set to 3, 6, 9, 12, 15, we generate 500 watermarked images for evaluation. Since the generation parameters are unknown during the inversion phase, the  $\text{guidance\_scale}$  is fixed at 3 for all inverse processes. The results in Tab. VI demonstrate that Gaussian Shading++ consistently outperforms PRCW [28] across all settings. This performance gain is attributed to our modeling of the AWGN Channel across the entire generation and inversion process, and the use of soft decision decoding. These results highlight the suitability of Gaussian Shading++ for real-world deployments where generation parameters may vary.

#### D. Ablation Studies

In this section, we conduct ablation studies on key modules of Gaussian Shading++, including the pseudorandom error-correcting codes (PRC), soft decision decoding (SDD), and the location of the PRC Channel in the latent space (PRC Loc). The last row of Tab. VII represents the default configuration of Gaussian Shading++. Finally, we evaluate the performance of Gaussian Shading++ under various noise strengths.

1) *Effect of PRC*: To evaluate the effectiveness of encoding the  $\text{seed}$  in the PRC Channel using PRC codes, we replace PRC with Bose–Chaudhuri–Hocquenghem codes (BCH) [82]. As shown in the first row of Tab. VII, this substitution leads to a notable decline in robustness. Furthermore, BCH codes lack pseudorandomness, making it challenging to achieve provable performance-lossless.



TABLE VI

ROBUSTNESS COMPARISON UNDER DIFFERENT GUIDANCE\_SCALE DURING GENERATION. WE CONTROL THE FPR AT  $10^{-6}$ , AND EVALUATE THE TPR / BIT ACCURACY FOR SD V2.1. THE GUIDANCE\_SCALE IS FIXED TO 3 DURING INVERSION. WITH NO BACKGROUND REPRESENTING PRCW [28] AND A GRAY BACKGROUND REPRESENTING GAUSSIAN SHADING++.

Guidance Scale	Noise						
	None	JPEG	Brightness	GauBlur	GauNoise	MedFilter	Resize
3	<b>1.000 / 1.000</b>	0.992 / 0.996	0.982 / 0.993	0.976 / <b>0.999</b>	0.996 / 0.997	0.98 / 0.992	0.998 / 0.999
	<b>1.000 / 1.000</b>	<b>0.994 / 0.997</b>	<b>0.990 / 0.995</b>	<b>0.998 / 0.993</b>	<b>0.998 / 0.999</b>	<b>0.994 / 0.994</b>	<b>1.000 / 1.000</b>
6	<b>1.000 / 1.000</b>	0.964 / 0.984	0.964 / 0.984	0.922 / 0.964	0.946 / 0.972	0.910 / 0.964	0.994 / 0.996
	<b>1.000 / 1.000</b>	<b>0.982 / 0.990</b>	<b>0.982 / 0.990</b>	<b>0.984 / 0.980</b>	<b>0.970 / 0.984</b>	<b>0.974 / 0.980</b>	<b>0.996 / 0.998</b>
9	<b>1.000 / 1.000</b>	0.898 / 0.953	0.952 / 0.977	0.784 / 0.907	0.790 / 0.892	0.832 / 0.919	0.980 / 0.992
	<b>1.000 / 1.000</b>	<b>0.966 / 0.979</b>	<b>0.964 / 0.981</b>	<b>0.938 / 0.952</b>	<b>0.848 / 0.919</b>	<b>0.938 / 0.960</b>	<b>0.996 / 0.997</b>
12	0.996 / 0.998	0.824 / 0.917	0.924 / 0.963	0.622 / 0.813	0.594 / 0.798	0.736 / 0.875	0.948 / 0.976
	<b>1.000 / 1.000</b>	<b>0.914 / 0.952</b>	<b>0.960 / 0.978</b>	<b>0.822 / 0.919</b>	<b>0.724 / 0.885</b>	<b>0.890 / 0.932</b>	<b>0.972 / 0.984</b>
15	0.994 / <b>0.998</b>	0.746 / 0.877	0.874 / 0.945	0.434 / 0.717	0.434 / 0.709	0.582 / 0.806	0.900 / 0.960
	<b>0.996 / 0.998</b>	<b>0.838 / 0.912</b>	<b>0.922 / 0.958</b>	<b>0.782 / 0.867</b>	<b>0.536 / 0.761</b>	<b>0.802 / 0.886</b>	<b>0.974 / 0.982</b>

TABLE VII

ABLATION STUDY ON KEY MODULES OF GAUSSIAN SHADING++. WE CONTROL THE FPR AT  $10^{-6}$ , AND EVALUATE THE TPR/BIT ACCURACY UNDER VARIOUS NOISE DISTORTIONS FOR SD V2.1. **BOLD** REPRESENTS THE BEST, UNDERLINE REPRESENTS THE SECOND BEST.

Module			Noise						
Encode	Decode	PRC Loc	None	JPEG	Brightness	GauBlur	GauNoise	MedFilter	Resize
BCH	SDD	front	0.994 / 0.997	0.588 / 0.794	0.820 / 0.910	0.008 / 0.503	0.644 / 0.820	0.062 / 0.531	0.732 / 0.866
PRC	HDD	front	<b>1.000 / 1.000</b>	0.966 / <u>0.977</u>	0.980 / 0.987	0.970 / 0.948	<u>0.914 / 0.952</u>	<b>0.964 / 0.959</b>	<b>0.996 / 0.995</b>
PRC	SDD-LLR	front	<b>1.000 / 1.000</b>	<u>0.970 / 0.984</u>	<b>0.990 / 0.994</b>	<b>0.974 / 0.972</b>	0.910 / 0.953	<b>0.964 / 0.975</b>	0.990 / 0.994
PRC	SDD	top	<b>1.000 / 1.000</b>	0.940 / 0.967	0.960 / 0.979	0.906 / 0.946	0.872 / 0.932	<u>0.936 / 0.962</u>	0.992 / <u>0.995</u>
PRC	SDD	bottom	<b>1.000 / 1.000</b>	0.958 / 0.974	0.966 / 0.981	0.934 / <u>0.959</u>	0.914 / 0.952	<u>0.932 / 0.959</u>	0.994 / <b>0.996</b>
PRC	SDD	back	<b>1.000 / 1.000</b>	0.948 / 0.973	0.966 / 0.982	0.846 / 0.920	0.888 / 0.942	0.906 / 0.950	0.988 / 0.993
PRC	SDD	front	<b>1.000 / 1.000</b>	<b>0.974 / 0.984</b>	<u>0.974 / 0.986</u>	<b>0.974 / 0.972</b>	<b>0.918 / 0.956</b>	<b>0.964 / 0.974</b>	<b>0.996 / 0.996</b>

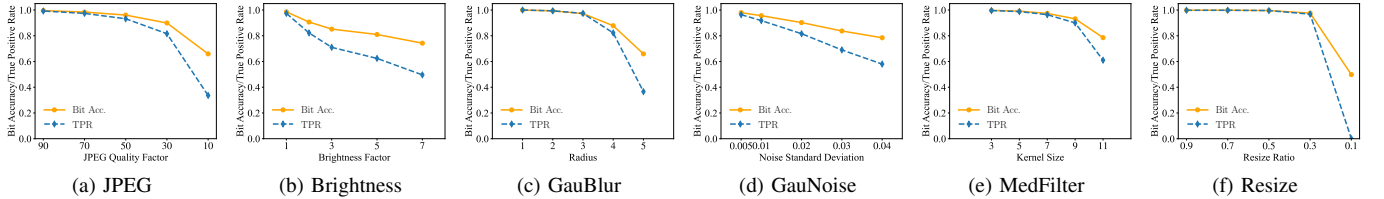


Fig. 8. Ablation study on different noise strengths.

2) *Effect of SDD*: To assess the effectiveness of the soft decision decoding, we remove the posterior estimation step in the GS Channel and instead apply hard decision decoding (HDD), as used in Gaussian Shading [27]. Specifically, watermark bits are directly inferred from the sign of elements in  $z_T^s$ , and final watermark recovery is performed via majority voting over bits at the same positions. As shown in the second row of Tab. VII, using HDD leads to a noticeable decline in overall robustness, particularly under filtering-based distortions such as Gaussian Blur and Median Filtering. These results highlight the advantage of our soft decision decoding strategy.

Additionally, we evaluate the soft decision decoding method using full LLR (SDD-LLR), as defined in Eq. 13. As shown in the third row of Tab. VII, SDD-LLR, which utilizes the arctanh function, performs similarly to the soft decision decoding, which applies a first-order approximation of arctanh. Nevertheless, soft decision decoding offers reduced computational complexity.

3) *PRC Channel Location*: Since the entire latent space is divided into the PRC Channel and the GS Channel, how these two are combined is crucial. We mainly discuss the position of the PRC Channel, with the remaining part being the GS Channel. Considering the latent space dimensions of  $[4, 64, 64]$ , we define four positions for the PRC Channel: top, occupying the upper half, i.e.,  $[0 : 4, 0 : 32, 0 : 64]$ ; bottom, occupying the lower half, i.e.,  $[0 : 4, 32 : 64, 0 : 64]$ ; front, occupying the first two channels, i.e.,  $[0 : 2, 0 : 64, 0 : 64]$ ; back, occupying the last two channels, i.e.,  $[2 : 4, 0 : 64, 0 : 64]$ . The experimental results, shown in the last four rows of Tab. VII, indicate that when the PRC Channel occupies the first two channels (front), Gaussian Shading++ achieves the best performance, which is also our default setting.

4) *Noise Strengths*: To further test the robustness, we conduct experiments using different intensities of noises. As show in Fig. 8, performance declines with higher intensities. However, for Brightness, Gaussian Noise, and Median Filter, even at high intensities, the bit accuracy remains above 75%.

## VI. CONCLUSION, LIMITATIONS, AND FUTURE WORK

We propose Gaussian Shading++, a performance-lossless watermarking method for diffusion models designed for practical deployment. First, we propose a double-channel design, utilizing the PRC Channel to encode the random seed required for the pseudorandomization of the GS Channel. This enables performance-lossless watermarking when the watermark key is fixed, addressing key management challenges in real-world applications. Second, we model the entire generation and inversion process as an additive white Gaussian noise channel and propose a novel soft decision decoding strategy for the maximum likelihood decoding in REP codes, ensuring strong robustness even when generation parameters vary. Third, we introduce the public-key signature ECDSA, extending watermark verification to any third party while providing resistance against forgery attacks. Extensive experiments validate that our method outperforms state-of-the-art approaches in both robustness and performance losslessness, making it a more practical watermarking solution for real-world deployment.

Although Gaussian Shading++ demonstrates excellent performance in practical deployment scenarios, there are still some limitations. First, while the GS Channel is sufficiently robust, the performance bottleneck lies in the PRC Channel. In scenarios where PRC [29] performs poorly, such as erasure attacks from VQ VAE [37], the robustness of Gaussian Shading++ is also affected. Second, the anti-forgery capability of Gaussian Shading++ in publicly verifiable scenarios still has room for improvement, as it remains vulnerable to forgery when attackers use proxy models with similar parameters. Lastly, although Exact Inversion [31] improves extraction accuracy, its optimization process inevitably increases the computational cost of watermark extraction.

Therefore, future work will focus on three key aspects. First, to fully leverage the performance of the GS Channel, more advanced pseudorandom coding schemes [83] should be introduced to construct a more robust watermark header. Second, exploring denoising sampling or latent representation decoding could help develop more secure and controllable watermarking methods to enhance anti-forgery capabilities in third-party verifiable scenarios. Third, faster inversion techniques [84] should be incorporated to reduce the computational cost of watermark extraction without sacrificing accuracy.

## REFERENCES

- [1] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *International conference on machine learning*. PMLR, 2015, pp. 2256–2265.
- [2] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," *Advances in neural information processing systems*, vol. 32, 2019.
- [3] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *International Conference on Learning Representations*, 2020.
- [4] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.
- [5] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *International Conference on Learning Representations*, 2020.
- [6] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.
- [7] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen, "Glaze: Towards photorealistic image generation and editing with text-guided diffusion models," *arXiv preprint arXiv:2112.10741*, 2021.
- [8] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker, *Digital watermarking and steganography*. Morgan kaufmann, 2007.
- [9] K. A. Zhang, L. Xu, A. Cuesta-Infante, and K. Veeramachaneni, "Robust invisible video watermarking with attention," *arXiv preprint arXiv:1909.01285*, 2019.
- [10] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, "Hidden: Hiding data with deep networks," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 657–672.
- [11] X. Luo, R. Zhan, H. Chang, F. Yang, and P. Milanfar, "Distortion agnostic deep watermarking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 13 548–13 557.
- [12] C. Zhang, P. Benz, A. Karjauv, G. Sun, and I. S. Kweon, "Udh: Universal deep hiding for steganography, watermarking, and light field messaging," *Advances in Neural Information Processing Systems*, vol. 33, pp. 10 223–10 234, 2020.
- [13] V. Kishore, X. Chen, Y. Wang, B. Li, and K. Q. Weinberger, "Fixed neural network steganography: Train the images, not the network," in *International Conference on Learning Representations*, 2021.
- [14] M. Tancik, B. Mildenhall, and R. Ng, "Stegastamp: Invisible hyperlinks in physical photographs," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2117–2126.
- [15] X. Zhong, P.-C. Huang, S. Mastorakis, and F. Y. Shih, "An automated and robust image watermarking scheme based on deep neural networks," *IEEE Transactions on Multimedia*, vol. 23, pp. 1951–1961, 2020.
- [16] Z. Jia, H. Fang, and W. Zhang, "Mbrs: Enhancing robustness of dnn-based watermarking by mini-batch of real and simulated jpeg compression," in *Proceedings of the 29th ACM international conference on multimedia*, 2021, pp. 41–49.
- [17] R. Ma, M. Guo, Y. Hou, F. Yang, Y. Li, H. Jia, and X. Xie, "Towards blind watermarking: Combining invertible and non-invertible mechanisms," in *Proceedings of the 30th ACM International Conference on Multimedia*, 2022, pp. 1532–1542.
- [18] P. Fernandez, A. Sablayrolles, T. Furon, H. Jégou, and M. Douze, "Watermarking images in self-supervised latent spaces," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 3054–3058.
- [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 *Proceedings of the 30th ACM international conference on multimedia*, 2022, pp. 2267–2275.
- [20] H. Fang, Y. Qiu, K. Chen, J. Zhang, W. Zhang, and E.-C. Chang, "Flow-based robust watermarking with invertible noise layer for black-box distortions," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, no. 4, 2023, pp. 5054–5061.
- [21] P. Fernandez, G. Couairon, H. Jégou, M. Douze, and T. Furon, "The stable signature: Rooting watermarks in latent diffusion models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 22 466–22 477.
- [22] Y. Zhao, T. Pang, C. Du, X. Yang, N.-M. Cheung, and M. Lin, "A recipe for watermarking diffusion models," *arXiv preprint arXiv:2303.10137*, 2023.
- [23] Y. Liu, Z. Li, M. Backes, Y. Shen, and Y. Zhang, "Watermarking diffusion model," *arXiv preprint arXiv:2305.12502*, 2023.
- [24] Y. Cui, J. Ren, H. Xu, P. He, H. Liu, L. Sun, and J. Tang, "Diffusion-shield: A watermark for copyright protection against generative diffusion models," *arXiv preprint arXiv:2306.04642*, 2023.
- [25] C. Xiong, C. Qin, G. Feng, and X. Zhang, "Flexible and secure watermarking for latent diffusion model," in *Proceedings of the 31st ACM International Conference on Multimedia*, 2023, pp. 1668–1676.
- [26] Y. Wen, J. Kirchenbauer, J. Geiping, and T. Goldstein, "Tree-ring watermarks: Fingerprints for diffusion images that are invisible and robust," *arXiv preprint arXiv:2305.20030*, 2023.
- [27] Z. Yang, K. Zeng, K. Chen, H. Fang, W. Zhang, and N. Yu, "Gaussian shading: Provable performance-lossless image watermarking for diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 12 162–12 171.
- [28] S. Gunn, X. Zhao, and D. Song, "An undetectable watermark for generative image models," in *The Thirteenth International Conference on Learning Representations*.

- [29] M. Christ and S. Gunn, "Pseudorandom error-correcting codes," in *Annual International Cryptology Conference*. Springer, 2024, pp. 325–347.
- [30] J. Fairuze, G. Ortiz-Jiménez, M. Vecerik, S. Jha, and S. Goyal, "On the difficulty of constructing a robust and publicly-detectable watermark," 2025. [Online]. Available: <https://arxiv.org/abs/2502.04901>
- [31] S. Hong, K. Lee, S. Y. Jeon, H. Bae, and S. Y. Chun, "On exact inversion of dpm-solvers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024, pp. 7069–7078.
- [32] W. Ryan and S. Lin, *Channel codes: classical and modern*. Cambridge university press, 2009.
- [33] A. Müller, D. Lukovnikov, J. Thietke, A. Fischer, and E. Quiring, "Black-box forgery attacks on semantic watermarks for diffusion models," 2024. [Online]. Available: <https://arxiv.org/abs/2412.03283>
- [34] X. ANSI, "62: public key cryptography for the financial services industry: the elliptic curve digital signature algorithm (ecdsa)," *Am. Nat'l Standards Inst.*, 1999.
- [35] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," *arXiv preprint arXiv:1802.01436*, 2018.
- [36] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Learned image compression with discretized gaussian mixture likelihoods and attention modules," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 7939–7948.
- [37] P. Esser, R. Rombach, and B. Ommer, "Taming transformers for high-resolution image synthesis," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 12 873–12 883.
- [38] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," *Advances in neural information processing systems*, vol. 34, pp. 8780–8794, 2021.
- [39] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8162–8171.
- [40] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu, "Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps," *Advances in Neural Information Processing Systems*, vol. 35, pp. 5775–5787, 2022.
- [41] S. Gu, D. Chen, J. Bao, F. Wen, B. Zhang, D. Chen, L. Yuan, and B. Guo, "Vector quantized diffusion model for text-to-image synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 696–10 706.
- [42] J. Ho and T. Salimans, "Classifier-free diffusion guidance," *arXiv preprint arXiv:2207.12598*, 2022.
- [43] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach, "Sdxl: Improving latent diffusion models for high-resolution image synthesis," 2023. [Online]. Available: <https://arxiv.org/abs/2307.01952>
- [44] B. Wallace, A. Gokul, and N. Naik, "Edict: Exact diffusion inversion via coupled transformations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 22 532–22 541.
- [45] G. Zhang, J. P. Lewis, and W. B. Kleijn, "Exact diffusion inversion via bidirectional integration approximation," in *European Conference on Computer Vision*. Springer, 2024, pp. 19–36.
- [46] A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, and D. Cohen-Or, "Prompt-to-prompt image editing with cross attention control," *arXiv preprint arXiv:2208.01626*, 2022.
- [47] R. Mokady, A. Hertz, K. Aberman, Y. Pritch, and D. Cohen-Or, "Null-text inversion for editing real images using guided diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 6038–6047.
- [48] Y. Zhang, N. Huang, F. Tang, H. Huang, C. Ma, W. Dong, and C. Xu, "Inversion-based style transfer with diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 10 146–10 156.
- [49] S. Mo, F. Mu, K. H. Lin, Y. Liu, B. Guan, Y. Li, and B. Zhou, "Freecontrol: Training-free spatial control of any text-to-image diffusion model with any condition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 7465–7475.
- [50] H. Li, C. Shen, P. Torr, V. Tresp, and J. Gu, "Self-discovering interpretable diffusion latent directions for responsible text-to-image generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 12 006–12 016.
- [51] J. Chung, S. Hyun, and J.-P. Heo, "Style injection in diffusion: A training-free approach for adapting large-scale diffusion models for style transfer," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024, pp. 8795–8805.
- [52] R. G. Van Schyndel, A. Z. Tirkel, and C. F. Osborne, "A digital watermark," in *Proceedings of 1st international conference on image processing*, vol. 2. IEEE, 1994, pp. 86–90.
- [53] D. Kundur and D. Hatzinakos, "A robust digital image watermarking method using wavelet-based fusion," in *Proceedings of International Conference on Image Processing*, vol. 1. IEEE, 1997, pp. 544–547.
- [54] M.-J. Tsai, K.-Y. Yu, and Y.-Z. Chen, "Joint wavelet and spatial transformation for digital watermarking," *IEEE Transactions on Consumer Electronics*, vol. 46, no. 1, p. 237, 2000.
- [55] H. Guo and N. D. Georganas, "Digital image watermarking for joint ownership," in *Proceedings of the tenth ACM international conference on Multimedia*, 2002, pp. 362–371.
- [56] S. Lee, C. D. Yoo, and T. Kalker, "Reversible image watermarking based on integer-to-integer wavelet transform," *IEEE Transactions on information forensics and security*, vol. 2, no. 3, pp. 321–330, 2007.
- [57] A. Al-Haj, "Combined dwt-dct digital image watermarking," *Journal of computer science*, vol. 3, no. 9, pp. 740–746, 2007.
- [58] S. Stankovic, I. Orovic, and N. Zaric, "An application of multidimensional time-frequency analysis as a base for the unified watermarking approach," *IEEE Transactions on Image Processing*, vol. 19, no. 3, pp. 736–745, 2009.
- [59] M. Hamidi, M. E. Haziti, H. Cherifi, and M. E. Hassouni, "Hybrid blind robust image watermarking technique based on dft-dct and arnold transform," *Multimedia Tools and Applications*, vol. 77, pp. 27 181–27 214, 2018.
- [60] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [61] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
- [62] N. Yu, V. Skripniuk, S. Abdelnabi, and M. Fritz, "Artificial fingerprinting for generative models: Rooting deepfake attribution in training data," in *Proceedings of the IEEE/CVF International conference on computer vision*, 2021, pp. 14 448–14 457.
- [63] N. J. Hopper, J. Langford, and L. Von Ahn, "Provably secure steganography," in *Advances in Cryptology—CRYPTO 2002: 22nd Annual International Cryptology Conference Santa Barbara, California, USA, August 18–22, 2002 Proceedings*. Springer, 2002, pp. 77–92.
- [64] K. Chen, H. Zhou, H. Zhao, D. Chen, W. Zhang, and N. Yu, "Distribution-preserving steganography based on text-to-speech generative models," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 5, pp. 3343–3356, 2021.
- [65] W. Zhang, K. Chen, and N. Yu, "Provable secure steganography: Theory, application and prospects," *Journal of Cybersecurity*, vol. 1, pp. 38–46, 2023.
- [66] W. Ryan and S. Lin, *Channel codes: classical and modern*. Cambridge university press, 2009.
- [67] M. P. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Transactions on information Theory*, vol. 41, no. 5, pp. 1379–1396, 2002.
- [68] J. Thietke, A. Müller, D. Lukovnikov, A. Fischer, and E. Quiring, "Towards a correct usage of cryptography in semantic watermarks for diffusion models," *arXiv preprint arXiv:2503.11404*, 2025.
- [69] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.
- [70] X. Zhao, K. Zhang, Z. Su, S. Vasan, I. Grishchenko, C. Kruegel, G. Vigna, Y.-X. Wang, and L. Li, "Invisible image watermarks are provably removable using generative ai," *Advances in Neural Information Processing Systems*, vol. 37, pp. 8643–8672, 2024.
- [71] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems*, vol. 30, 2017.
- [72] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.
- [73] P. Schramowski, M. Brack, B. Deiseroth, and K. Kersting, "Safe latent diffusion: Mitigating inappropriate degeneration in diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 22 522–22 531.
- [74] B. An, M. Ding, T. Rabbani, A. Agrawal, Y. Xu, C. Deng, S. Zhu, A. Mohamed, Y. Wen, T. Goldstein *et al.*, "Waves: Benchmarking the robustness of image watermarks," *arXiv preprint arXiv:2401.08573*, 2024.

- [75] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13*. Springer, 2014, pp. 740–755.
- [76] M. Cherti, R. Beaumont, R. Wightman, M. Wortsman, G. Ilharco, C. Gordon, C. Schuhmann, L. Schmidt, and J. Jitsev, "Reproducible scaling laws for contrastive language-image learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2818–2829.
- [77] N. V. Smirnov, "On the estimation of the discrepancy between empirical curves of distribution for two independent samples," *Bull. Math. Univ. Moscou*, vol. 2, no. 2, pp. 3–14, 1939.
- [78] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality (complete samples)," *Biometrika*, vol. 52, no. 3-4, pp. 591–611, 1965.
- [79] T. W. Anderson, "On the distribution of the two-sample cramer-von mises criterion," *The Annals of Mathematical Statistics*, pp. 1148–1159, 1962.
- [80] C. M. Jarque and A. K. Bera, "A test for normality of observations and regression residuals," *International Statistical Review/Revue Internationale de Statistique*, pp. 163–172, 1987.
- [81] R. B. D'agostino, A. Belanger, and R. B. D'Agostino Jr, "A suggestion for using powerful and informative tests of normality," *The American Statistician*, vol. 44, no. 4, pp. 316–321, 1990.
- [82] R. C. Bose and D. K. Ray-Chaudhuri, "On a class of error correcting binary group codes," *Information and control*, vol. 3, no. 1, pp. 68–79, 1960.
- [83] O. Alrabiah, P. Ananth, M. Christ, Y. Dodis, and S. Gunn, "Ideal pseudorandom codes," *arXiv preprint arXiv:2411.05947*, 2024.
- [84] S. Hong, S. Y. Jeon, K. Lee, E. Ryu, and S. Y. Chun, "Gradient-free decoder inversion in latent diffusion models," *Advances in Neural Information Processing Systems*, vol. 37, pp. 82 982–83 007, 2024.