frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 20 January 2026
DOI 10.3389/fphy.2025.1750515

OPEN ACCESS

EDITED BY
Muhammad Shahbaz Khan,
Edinburgh Napier University, United Kingdom

REVIEWED BY
Guang Hua,
Singapore Institute of Technology, Singapore
Yuan Rao,
Guangzhou University, China
Pallavi Kulkarni,
Dayananda Sagar College of Engineering, India

*CORRESPONDENCE
Ahmad Saeed Khan,
ahmad-saeed.khan@oru.se

RECEIVED 20 November 2025
REVISED 16 December 2025
ACCEPTED 29 December 2025
PUBLISHED 20 January 2026

CITATION
Hu X, Zhang B, Al-Dossari M, El-Gawaad NSA, Rakhimzhanova M and Khan AS (2026) Robust watermarking for diffusion models using error-correcting codes and post-quantum key encapsulation.
Front. Phys. 13:1750515.
doi: 10.3389/fphy.2025.1750515

Robust watermarking for diffusion models using error-correcting codes and post-quantum key encapsulation

Xianglei Hu[1], Beining Zhang[1], Mawaheb Al-Dossari[2], N. S. Abd El-Gawaad[3], Mira Rakhimzhanova[4] and Ahmad Saeed Khan[5]*

[1]School of Computer Science, Guangdong Polytechnic Normal University, Guangzhou, China, [2]Department of Physics, Faculty of Science, King Khalid University, Abha, Saudi Arabia, [3]Health Specialties, Basic Sciences and Applications Unit, Applied College, King Khalid University, Muhayil Asir, Abha, Saudi Arabia, [4]School of Artificial Intelligence and Data Science, Astana IT University, Astana, Kazakhstan, [5]School of Science and Technology, Örebro University, Örebro, Sweden

Critical infrastructures increasingly rely on AI-generated content (AIGC) for monitoring, decision support, and autonomous control. This dependence creates new attack surfaces: forged maintenance imagery, manipulated diagnostic scans, or spoofed sensor visualisations can trigger unsafe actions, regulatory violations, or systemic disruption. This paper proposes a post-quantum watermarking framework designed for critical infrastructure security. We embed robust provenance markers directly into the latent space of diffusion models, rather than at the pixel level, and reinforce them using error-correcting codes (ECC) to ensure watermark recoverability even after aggressive distortions such as compression, cropping, noise injection, and filtering. To secure watermark keys in transit and at rest, we integrate Kyber, a lattice-based key encapsulation mechanism standardised for post-quantum cryptography, to protect the watermark stream key against quantum-enabled interception. The resulting scheme (i) preserves visual fidelity, (ii) supports reliable forensic attribution and auditability under hostile conditions, and (iii) remains cryptographically secure in the post-quantum era. Experiments show that the proposed ECC-hardened latent watermarking achieves consistently high extraction accuracy across diverse attacks while maintaining image quality, outperforming state-of-the-art diffusion watermarking baselines. We position this watermarking–encryption pipeline as an enabling mechanism for privacy-aware traceability, zero-trust validation, and quantum-resilient content governance in next-generation critical infrastructure.

KEYWORDS

critical infrastructure security, diffusion model, diffusion models, error-correcting codes, infrastructure resilience, kyber, post-quantum cryptography, privacy and provenance

1 Introduction

In recent years, Diffusion Models (DM) [1–5] have garnered significant attention and emerged as a cornerstone technology in artificial intelligence (AI), owing to their ability to efficiently generate high-fidelity images [4–7]. When trained on large-scale datasets,

these models can synthesize high-resolution, high-quality images from text descriptions. Despite their utility in daily life and work, these technologies inevitably pose societal risks, including the dissemination of misinformation and copyright infringement [8,9]. In the context of critical infrastructure—such as energy grids, transportation systems, healthcare, finance, and communication networks—the misuse of AI-generated content (AIGC) could lead to severe operational disruptions, safety hazards, and legal accountability gaps. For instance, synthetically generated inspection reports, facility schematics, or sensor data simulations must be traceable to their origin to ensure integrity and compliance in regulated environments. Therefore, it is imperative to develop technical solutions capable of reliably identifying images synthesized by latent diffusion models and tracing their provenance, thereby enabling accountable data governance and forensic auditing in critical infrastructure applications.

Digital watermarking [10] has long provided a mechanism for copyright protection and content authentication. By embedding imperceptible identifiers into multimedia data, watermarking enables ownership verification and source tracking. For critical infrastructure, where data authenticity and non-repudiation are paramount, watermarking can establish a verifiable chain of custody for synthetic assets, such as diagnostic imagery or sensor visualizations, thereby supporting auditability throughout the content lifecycle. Nowadays, we can also embed watermark information into the generated image [11–17], allowing subsequent copyright authentication and tracking of false content. The existing watermarking methods for diffusion models can be divided into three categories. One is the post-processing watermark [18–21], which method usually adds watermarking information to the generated image by adjusting the image features, but this method may lead to the degradation of image quality. The other method [22–28] based combines the watermark embedding process within the image generation process, and embeds the watermark information into the image by fine-tuning the model. Although this method can avoid the degradation of the generated image quality, it also increases the computational cost and may affect the generation performance. These limitations become particularly acute in critical infrastructure environments, where watermarks must withstand not only common image manipulations but also domain-specific perturbations—such as compression in telemedicine systems or noise in industrial sensor networks—while maintaining strict performance and compliance standards.

To address these challenges, this study integrates error correction coding (ECC) [29] into the watermark embedding process of diffusion models. ECC augments watermark data with redundant bits, forming codewords that correct errors introduced by distortions like compression or cropping. By preprocessing watermark data with ECC schemes such as BCH [30] or LDPC [31], we enhance extraction accuracy and robustness without significant overhead.

Additionally, the Kyber algorithm [32]—a post-quantum key encapsulation mechanism (KEM) based on the Module Learning with Errors (MLWE) problem and standardized as FIPS 203—secures the stream key against quantum-era threats. Kyber ensures efficient, compact encryption, bolstering watermark confidentiality. This study proposes a robust digital watermarking framework for diffusion models with post-quantum integrity that significantly enhances watermark robustness while preserving the quality of generated images. Compared to the Gaussian Shading [12] technique—which involves repeatedly expanding watermark information to match latent feature dimensions and resampling initial latent features after stream key encryption—the proposed method reduces watermark-induced image quality degradation but suffers from limited extraction accuracy. To address this limitation, we introduce an error-correcting code (ECC) precoding mechanism. Prior to stream key encryption and distributed hold sampling, the watermark information undergoes ECC encoding to construct an error-correcting structure, significantly enhancing watermark robustness. Therefore, the contributions of this work can be summarized as follows:

1. Our method enhances robustness by distributing watermark information throughout the entire latent space using error correction codes. The error correction capability against other forms of attacks is also enhanced to varying degrees.
2. The Post-Quantum Kyber algorithm provides robust encryption for the watermark system's stream key, securing key transmission to significantly enhance watermark protection and augment the copyright protection capabilities of the diffusion model-based system. This is especially critical in distributed infrastructure networks where key exchange must remain resilient against eavesdropping and man-in-the-middle attacks.

The remainder of this paper is structured as follows. Section 2 reviews related work. Section 3 presents the watermark embedding and extraction framework. Section 4 details the experimental results and compares robustness and visual quality with existing techniques. Section 5 concludes this study.

2 Related work

2.1 Diffusion models

Diffusion models are a class of deep generative models grounded in non-equilibrium thermodynamics. They aim to synthesizing novel samples matching the original data distribution by learning the generative process. Specifically, the forward diffusion process gradually perturbs data into Gaussian noise through iterative noise addition, while the reverse denoising process trains a neural network to iteratively restore the data distribution, thereby yielding high-quality samples.

Diffusion models learn to approximate the target distribution $p_\theta(x_t)$ from the real data distribution $q(x)$ through forward and reverse Markov diffusion processes. Specifically, these models train a noise predictor $\epsilon_\theta(x_t, t)$ and generate images x_0 from Gaussian noise x_T through iterative noise estimation and T-step denoising. To accelerate generation [3], proposed the Denoising Diffusion Implicit Model (DDIM), which reduces the sampling steps from approximately 1000 to about 50. To further reduce computational costs while maintaining output quality, the Latent Diffusion Model (LDM) [33] performs the diffusion process in a compressed latent space, establishing the current mainstream paradigm for image generation with diffusion models. The

forward diffusion process of the diffusion model is defined as Markov:

$$q(x_{1:T}|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1}) \tag{1}$$

$$q(x_t|x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I\right) \tag{2}$$

$q(x_{1:T}|x_0)$ denotes the probability distribution of the noisy image x_T obtained from the original image x_0 via a T-step noise-addition process. It is a Gaussian distribution with mean $\sqrt{1-\beta_t}x_{t-1}$ and variance β_t, where β_t is a predetermined noise-variance coefficient and I is the identity matrix.

The goal of the reverse diffusion of LDM is to learn the joint distribution:

$$p_\theta(x_{0:T}) = p(x_T)\prod_{t=1}^{T} p_\theta(x_{t-1}|x_t) \tag{3}$$

where the prior distribution is given as $p(x_T) = N(0, I)$. The Latent Diffusion Model (LDM) generates images by executing the reverse diffusion process on a latent feature z_T.

$$z_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(z_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(z_t, t)\right) \quad (t = 1, 2, \ldots, T) \tag{4}$$

where $\alpha_t = \sqrt{1-\beta_t}$ and $\bar{\alpha}_t = \prod_{i=1}^{t}\alpha_i$, the noise predictor $\epsilon_\theta(z_t, t)$ is trained to estimate the noise introduced in the forward diffusion process. In this paper, we employ the classical Stable Diffusion model to illustrate our watermarking method. The dimensionalities of the original image x and the latent feature z_t are (3, 512, 512) and (1, 4, 64, 64), respectively.

2.2 Watermarks for latent diffusion models

Latent Diffusion Models (LDMs) allow users to create style-specific images via training and fine-tuning. Yet these capabilities raise concerns about misuse, particularly the unauthorized commercial exploitation of LDM outputs that lack intrinsic copyright safeguards. Therefore, enhancing copyright protection and traceability for LDMs is crucial. Digital watermarking technology offers a proven approach to mitigate these issues by embedding imperceptible information into content. This technique involves embedding watermarks into generated images to enable source identification and verification. As shown in Figure 1, the existing digital watermarking methods for LDMs can be categorized into three types: postprocessing, generative, and latent feature-based watermarking.

Post-processing watermarking embeds watermarks after image synthesis to assert copyright. Representative schemes include DwtDct [18] and RivaGAN [19]. The approach is straightforward to integrate into open-source frameworks such as Stable Diffusion, enabling direct watermark injection into output images. However, its fundamental limitation lies in the direct modification of pixel data, which introduces artifacts or texture distortion and consequently degrades visual quality. Additionally, such watermarks remain vulnerable to targeted attacks (e.g., cropping, filtering), thereby compromising the reliability of copyright identification.

Generative watermarking integrates embedding with the generation pipeline, eliminating post-processing. Representative methods including Stable Signature [23] and AquaLora [24] enhance watermark concealment while preserving image generation quality. A key advantage of this approach is the deep fusion of watermarks with image content, resulting in significantly enhanced resistance to attacks compared to post-processing methods. However, limitations include the requirement for model retraining or fine-tuning, substantial computational overhead, and the necessity to repeat training processes when adapting to different style-specific models, consequently restricting flexibility.

Latent feature-based watermarking technology operates within the latent space of diffusion models, enabling watermark embedding without parameter modification. Representative approaches include Tree-ring [34], which encodes watermarks in the frequency domain of latent noise using ring-shaped patterns to achieve robust traceability. However, this approach does not incorporate user identity information, permitting only model origin verification rather than specific user tracking, thereby limiting capabilities for pursuing legal accountability. DiffuseTrace [35] employs an encoder to modify the initial latent noise. Gaussian Shading [12] maps watermarks to latent feature following Gaussian distributions. These methods avoid fine-tuning overhead and provide high deployment convenience, but face challenges in watermark robustness that require further optimization of interference resistance.

2.3 Error correcting code

Error-correcting codes (ECC) represent fundamental technologies in information theory and communications, employed to detect and correct errors during data transmission or storage through the introduction of redundancy. The fundamental principle involves encoding original information into codewords containing redundant bits using specific algorithms, utilizing the Hamming distance between codewords to detect and correct errors: a larger minimum Hamming distance corresponds to stronger error correction capabilities. ECC is primarily categorized into two types: block codes [30,31,36] and convolutional codes [37,38]. Block codes encode fixed-length data blocks independently, making them suitable for storage systems. Convolutional codes process continuous data streams through shift registers and utilize the Viterbi algorithm for soft-decision decoding, rendering them widely applicable in wireless communications. This study focuses on applying block codes to watermarking in diffusion models.

While numerous ECC families exist, this work concentrates on BCH and LDPC codes due to their inherent alignment with the characteristics of latent-space watermark embedding in diffusion models. First, the watermark is encoded as a binary sequence, and the perturbations introduced during latent inversion and image-level attacks predominantly manifest as independent bit flips rather than symbol-level erasures. BCH and LDPC codes operate natively in the binary domain, enabling them to directly address this bit-level distortion pattern. In contrast, Reed-Solomon codes [36] are symbol-oriented and optimized for burst errors over large finite fields. Consequently, they are less efficient against the sparse, randomly distributed distortions typical of latent representations.

Second, BCH and LDPC codes offer flexible code lengths and rates, which can be adapted to the spatial capacity constraints of latent feature maps. Their encoding and decoding processes incur low computational overhead and are compatible with iterative extraction pipelines. LDPC decoding performs well when soft information (e.g., log-likelihood ratios) is available, whereas BCH decoding provides deterministic algebraic correction that remains stable even under low initial bit reliability. Therefore, these two codes together facilitate a balanced comparative analysis of deterministic versus iterative decoding strategies under latent-space perturbations.

Therefore, BCH and LDPC codes represent complementary and practically deployable ECC structures for diffusion-model watermarking, making them well-suited for the comparative analysis conducted in this study.

2.3.1 BCH code

BCH codes [30] are a class of linear block codes in error-correcting coding theory. They can be integrated into watermarking systems to enhance robustness. For example, in video watermarking, adaptive BCH coding has been successfully combined with ring tensor features, markedly improving resilience against a range of attacks [39]. Their core principle involves constructing redundant parity bits through generator polynomials and performing error location and correction via algebraic operations over finite fields. The construction of BCH codes relies on finite fields (Galois fields, GF) and minimal polynomials. BCH code encoding involves the following main steps.

Consider a finite field $GF(q)$, where q represents a prime power. Binary BCH codes are most commonly defined over $GF(2)$. Their extension field $GF(2^m)$ can be constructed using primitive polynomials. The code length for such BCH codes is typically $n = 2^m - 1$.

The generator polynomial $g(x)$ is the fundamental component of a BCH code. It is defined as the least common multiple (LCM) of the minimal polynomials corresponding to a set of consecutive powers of the primitive element:

$$g(x) = LCM\{m_1(x), m_2(x), \ldots, m_{2t}(x)\} \tag{5}$$

where $m_i(x)$ denotes the minimal polynomial of element α^i over $GF(2)$, α is a primitive element of $GF(2^m)$, and t represents the error-correction capability of the code. The degree of the generator polynomial determines the number of check bits $r = n - k$, which satisfies $r \le mt$.

The encoding process of BCH codes systematically converts the information polynomial into a codeword polynomial. Let the information polynomial be denoted as:

$$u(x) = u_0 + u_1 x + \cdots + u_{k-1} x^{k-1} \tag{6}$$

The encoding operation is realized by generating polynomial $g(x)$:

$$c(x) = u(x) \cdot x^{n-k} + \left[u(x) \cdot x^{n-k} \bmod g(x)\right] \tag{7}$$

where the remainder term constitutes the check polynomial. This encoding can be efficiently implemented using a linear feedback shift register, which ensures all generated codewords maintain the necessary cyclic properties.

BCH decoding locates and corrects errors by adjoint. The process is as follows: The calculation of adjoint: accept vector $r(x) = c(x) + e(x)$ ($e(x)$ is the error polynomial), calculate the adjoint.

$$S_j = r\left(\alpha^j\right) = \sum_{i=0}^{n-1} r_i \cdot \left(\alpha^j\right)^i, \quad j = 1, 2, \ldots, 2t \tag{8}$$

if $S_j = 0$ holds for all j, then there is no error; otherwise enter the error correction process.

Error localization polynomial error correction: Using the Berlekamp-Massey (BM) algorithm to solve the error localization polynomial $U(x)$:

$$U(x) = \prod_{i=1}^{v} (1 - xX_i) = 1 + U_1 x + \cdots + U_v x^v \tag{9}$$

where $X_i = \alpha^{p_i}$ denotes the i-th error location and $v \le t$ represents the actual number of errors. By solving for the roots of $U(x)$, their reciprocals correspond to the error locations p_i. For binary BCH codes, error correction is accomplished by directly performing bit-flipping operations on the identified error positions.

2.3.2 LDPC code

Low-density parity-check (LDPC) codes [31] are a class of linear block codes characterized by a sparse parity-check matrix. The "low-density" property refers to the fact that the vast majority of entries in this matrix are zeros. Owing to their excellent error-correction performance, LDPC codes can be incorporated into watermarking systems to significantly improve robustness. For instance, semi-random LDPC codes have been integrated with a spatial-chromaticity Fourier transform to develop image watermarking schemes that achieve both high robustness and capacity [40]. LDPC codes are error-correcting codes based on sparse graphs, typically featuring large-sized low-density parity-check matrices, meaning that most elements in the matrix are 0 while only a few are 1. LDPC codes are primarily used to correct bit-level, random errors in codewords.

Let the information bit sequence be denoted by $\mathbf{u} = [u_1, u_2, \ldots, u_k]$ and the resulting codeword by $\mathbf{c} = [c_1, c_2, \ldots, c_n]$, where $n > k$ and the code rate is $R = k/n$. A valid codeword c must satisfy all constraints imposed by the parity-check matrix H; that is:

$$\mathbf{H}\mathbf{c}^T = \mathbf{0} \tag{10}$$

This equation represents the fundamental constraint for LDPC codes, where all operations are performed under modulo-2 arithmetic in the binary Galois Field $GF(2)$.

To achieve systematic coding, where the codeword directly contains the original information bits, the corresponding generator matrix G is derived from the parity-check matrix H, with dimensions $k \times n$. Through algorithms like Gaussian elimination, H can be transformed into systematic form via row operations:

$$\mathbf{H} = \left[\mathbf{P}^T | \mathbf{I}_m\right] \tag{11}$$

where I is an $m \times m$ identity matrix and P is a $k \times m$ dense matrix. The corresponding generator matrix G can then be constructed as:

$$\mathbf{G} = \left[\mathbf{I}_k | \mathbf{P}\right] \tag{12}$$

The encoding operation involves the matrix multiplication of the information vector u and the generator matrix G, expressed as:

$$\mathbf{c} = \mathbf{u} \cdot \mathbf{G} = [\mathbf{u}|\mathbf{uP}] \tag{13}$$

The resulting codeword c is systematically composed of the original information bits u and the calculated check bits $\mathbf{p} = \mathbf{uP}$, and therefore inherently satisfies the constraint $\mathbf{Hc}^T = \mathbf{0}$.

Decoding is performed using the min-sum algorithm [41]. First, the log-likelihood ratios are initialized according to the received vector y, yielding $L_n^{(0)}$:

$$L_n^{(0)} = \ln \frac{P(y_n|c_n = 0)}{P(y_n|c_n = 1)} \tag{14}$$

Each check node is updated and forwards the least-reliable adjacent information to its connected variable nodes, prioritizing adjustment of the least-reliable bits:

$$L_{m\to n}^{(L)} \approx \left(\prod_{n' \in N(m)\backslash n} \operatorname{sgn}\left(L_{n'\to m}^{(L-1)}\right) \right) \cdot \min_{n' \in N(m)\backslash n} \left|L_{n'\to m}^{(L-1)}\right| \tag{15}$$

Variable point update, if most check nodes support the current bit value, enhance its confidence; if there is a conflict, the reliability of the current value is weakened:

$$L_{n\to m}^{(L)} = L_n^{(0)} + \sum_{m' \in M(n)\backslash m} L_{m'\to n}^{(L)} \tag{16}$$

where $N(m)$ denotes the set of variable nodes adjacent to the m-th check node, and $M(n)$ denotes the set of check nodes adjacent to the n-th variable node.

Finally, the judgment process is performed:

$$c_n = \begin{cases} 0 & \text{if } L_n^{(l_{\max})} \geq 0 \\ 1 & \text{otherwise} \end{cases} \tag{17}$$

if the verification matrix multiplied by c equals 0 or the maximum number of iterations is reached, decoding is terminated and the decoded information is output.

2.4 Post-quantum key encapsulation mechanism

Kyber [32] is a post-quantum key encapsulation mechanism (KEM) whose security is based on the hardness of the Module Learning with Errors (MLWE) problem, providing a rigorous foundation in lattice-based cryptography. In a related advancement [42], pioneered a white-box watermarking signature scheme, demonstrating the practical synergy between post-quantum KEMs and watermarking for enhanced model copyright protection. Algorithmically, Kyber employs a matrix-vector arithmetic structure over a polynomial ring, achieving a provable security guarantee while maintaining high computational efficiency. The three core algorithmic components are detailed in the following subsections.

2.4.1 KeyGen

The key generation phase produces a key pair: a public key for encryption and a private key for decryption. During initialization, security parameters including the dimension nk and the polynomial ring are defined R_q, and the public matrix $A \in R_q^{b\times b}$ is randomly generated, where b depends on the security level. Subsequently, a secret vector $d \in R_q^b$ and an error vector $e \in R_q^b$ are sampled, with their coefficients drawn from a centered binomial distribution or a discrete Gaussian distribution to maintain the requisite small-norm properties. Finally, the public key pk = (A, dt) and the private key sk = d are computed as shown in Equation 18.

$$dt = A \cdot d + e \in R_q^b \tag{18}$$

2.4.2 Encapsulate

During the encapsulation phase, a sender can encapsulate a random session key using the recipient's public key. The vectors $g \in R_q^k$, $e1 \in R_q^b$, and $e_2 \in R_q$ are randomly generated, with their coefficients drawn from a small-error distribution. The two ciphertext components are then computed using Equations 19, 20.

$$du = A^T \cdot g + e1 \in R_q^b \tag{19}$$

$$dv = A^T \cdot g + e2 + m \in R_q \tag{20}$$

where $km \in R_q$ represents the encoded message. The shared session key K is then derived via $K = H(km)$, producing the final ciphertext $C = (du, dv)$ and session key K.

This construction guarantees ciphertext security. Even if an adversary obtains the public values A and dt, the secret values g and km cannot be efficiently recovered due to the computational hardness of the underlying MLWE problem.

2.4.3 Decapsulate

During decryption, the recipient employs their private key to recover the session key from the ciphertext. Using Equation 21, error compensation yields w, which is then decoded via Equation 22 to obtain km'. The shared key K' is subsequently reconstructed using the hash function $K' = H(km')$.

$$w = dv - d^T \cdot du \in R_q \tag{21}$$

$$km' = dv - d^T \cdot du = \left(d^T \cdot A \cdot g + d^T \cdot e1 + e2 + km\right) - d^T \cdot \left(A^T \cdot g + e1\right) \approx km \tag{22}$$

The Kyber algorithm implements secure key exchange through the aforementioned three-phase procedure. Its design incorporates both post-quantum security and practical deployment efficiency. This combination makes it suitable for diverse secure communication scenarios, including stream-key protection within this paper's watermarking framework.

3 Methods

This section details the robust image watermarking method with error correction coding technology intergrated into the diffution model. The complete process consists of two parts: watermark embedding and watermark extraction.

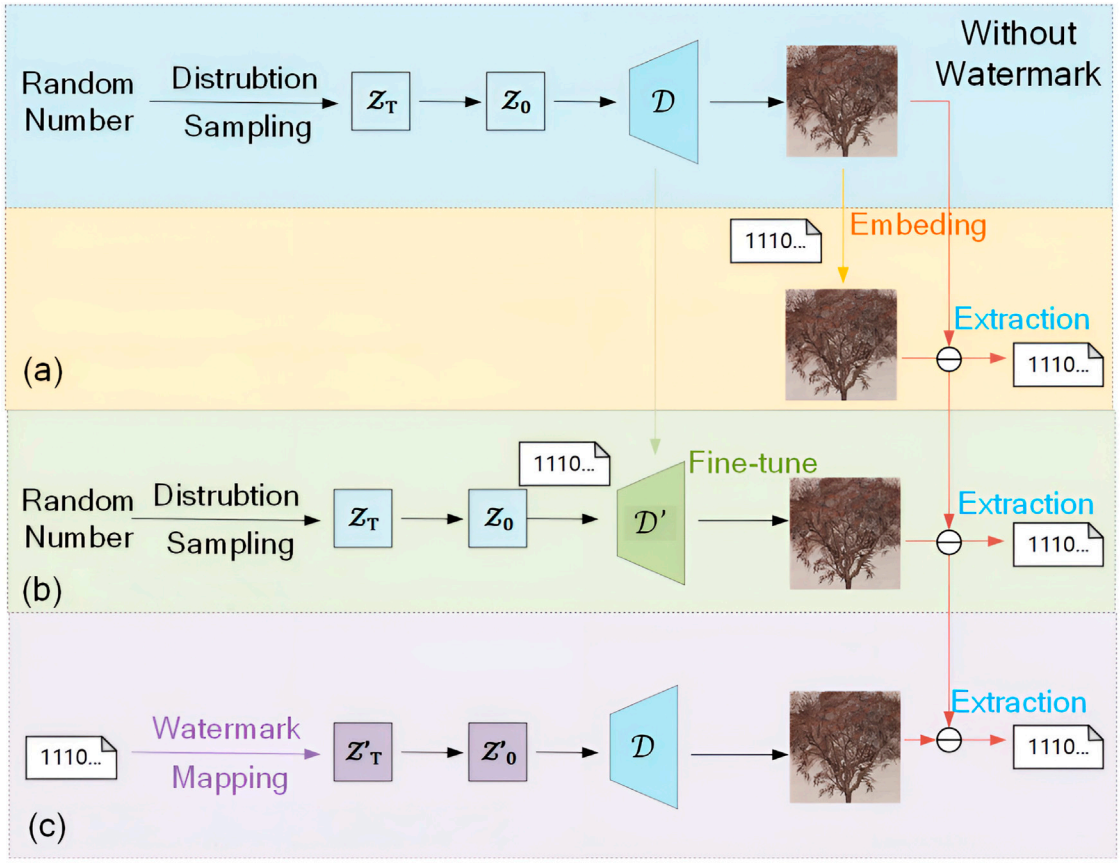

FIGURE 1
Existing watermarking frameworks are broadly categorized into three types: post-processing-based, model fine-tuning-based, and latent feature-based methods. Our latent-representation approach significantly outperforms prior schemes in robustness. **(a)** Post-processing-based. **(b)** Fine-tuning-based. **(c)** Latent-representation-based.

3.1 Watermark embedding

The watermark embedding process aims to achieve robustness, security, and statistical imperceptibility within latent diffusion models, and therefore, it integrates error correction, cryptographic protection, and distribution-aware sampling into a unified embedding process: (1) the original watermark sequence is encoded using ECC to bolster robustness against noise and distortions; (2) the watermark security against unauthorized extraction is guaranteed via a post-quantum cryptographic mechanism; (3) the encrypted watermark is embedded into latent feature using a distribution-preserving sampling strategy, ensuring alignment with the original latent prior. Collectively, these steps form the complete embedding pipeline, whose architecture is illustrated in Figure 2 and detailed in Sections 3.1.1–3.1.3.

3.1.1 Error correcting schemes for robust watermark

Before formal embedding and encryption, it is necessary to implement Error Eorrection measures on the original watermark information u so as to ensure that the original watermark u can be recovered even after critical infrastructures undergo attacks. To maximize the robustness, we designed a two-layer error correction scheme: the first layer applies a block-based ECC to impose global structural constraints on the watermark sequence; the second layer introduces repetition coding to mitigate fine-grained, random bit flips induced by stochastic sampling and quantization. This two-layer strategy provides significantly higher robustness than single-layer schemes.

In the first encoding layer, the original binary watermark sequence $u \in \{0,1\}^l$ is encoded using linear block codes, such as BCH, LDPC or their cascaded combinations, producing an intermediate sequence $c_1 \in \{0,1\}^{l_1}$. Such block-level coding corrects correlated or structured errors that may accumulate during diffusion sampling and latent inversion. As detailed in Section 2.3, BCH and LDPC codes are particularly suited for this role due to their binary-domain operation and efficacy in correcting multiple distributed bit errors.

In the second encoding layer, repetition coding is applied to strengthen the intermediate sequence c_1 into c. In the layer of repetition coding, each bit is replicated r times to form the final encoded sequence $c \in \{0,1\}^{l_2}$, which is embedded into independent latent variables using the distribution-preserving sampling strategy from Section 3.1.3. This layer primarily counteracts localized random bit flips caused by sampling stochasticity and quantization uncertainty. Distributing replicas across independent

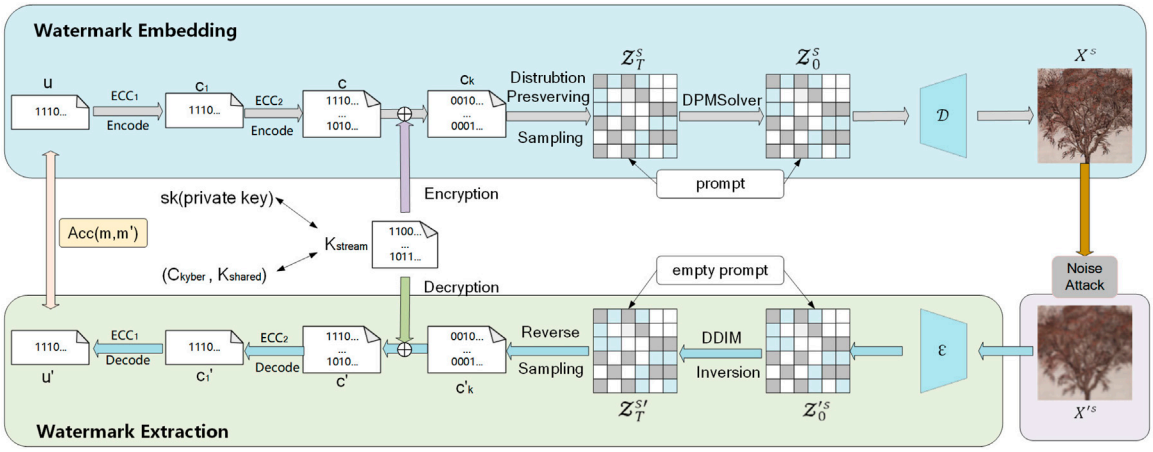

FIGURE 2
Error-correcting code watermarking framework. A binary sequence u represents the watermark. It is first encoded by ECC_1 to yield c_1, then by ECC_2 to yield c. After encryption, c_k is mapped to the initial latent feature via distribution-preserving sampling. Denoising produces the watermarked image X^s. Extraction reverses the DDIM inversion and subsequent steps.

latent positions ensures that random perturbations affect them independently.

When attacked, the second layer's majority voting across the bit repetition suppresses dominant random errors, yielding a stabilized estimate of c_1. The block-level ECC decoder subsequently corrects residual inconsistencies to recover the original watermark. Thus, the hierarchical design allocates roles clearly: repetition coding handles prevalent random noise, while block-based ECC performs precise correction of the global structure.

By incorporating the two-layer Error Correcting scheme, the proposed encoding strategy achieves a level of robustness unattainable by either repetition coding or block-level ECC alone. This two-layer framework constitutes a core component of the watermarking method and directly explains the robustness gains demonstrated experimentally.

3.1.2 Apply kyber algorithm

First, a Kyber key pair (pk, sk) is generated. The public key, pk, is used to encrypt the stream key, while the private key, sk, is retained by the authorized party for decryption. The randomly generated stream key is then encapsulated using the Kyber algorithm.

$$\left(C_{kyber}, K_{shared}\right) = \text{Kyber.Encapsulate}\left(\text{pk}\right) \tag{23}$$

Using the shared key K_{shared} to encrypt the stream key:

$$K_{enc} = AES\left(K_{shared}, K_{stream}\right) \tag{24}$$

where AES denotes the symmetric encryption algorithm.

During decapsulation, the private key sk and the ciphertext C_{kyber} are employed to recover the shared key.

$$K_{shared} = \text{Kyber.Decapsulate}\left(C_{kyber}, \text{sk}\right) \tag{25}$$

Following decryption, the original stream key K_{stream} is recovered and utilized for subsequent watermark decoding operations.

$$K_{stream} = AES^{-1}\left(K_{shared}, K_{enc}\right) \tag{26}$$

The security of the proposed framework relies on the Kyber algorithm, whose architecture is illustrated in Figure 3. Even if an adversary acquires the ciphertext C_{kyber} and the encapsulated key K_{enc}, the original stream key cannot be recovered without the private key sk. This ensures the confidentiality of the embedded watermark information.

3.1.3 Distribution preserving sampling

This section presents a watermark-guided sampling strategy for initial latent features, designed to address a core challenge in latent-space watermarking: the embedding of discrete watermark bits while preserving consistency with the continuous prior distribution (e.g., a standard Gaussian, $\mathcal{N}(0, I)$) assumed by the diffusion model. Direct modification of latent variables to encode watermark information can introduce detectable distributional shifts, compromising both the quality of generated images and the stealth of the watermark. To overcome this, we introduce a distribution-preserving sampling strategy that ensures watermark embedding does not alter the underlying latent distribution. Specifically, the method employs a deterministic probabilistic mapping mechanism, guaranteeing that the latent features carrying the embedded watermark remain conformant to the original Gaussian prior $\mathcal{N}(0, I)$.

First, a random binary key matching the dimensions of the carrier signal is combined with the encoded watermark information to produce a randomized watermark. The encrypted data follows a discrete uniform distribution. Let the latent feature space follow $Z \sim \mathcal{N}(0, I)$, with probability density function $f(x)$, cumulative distribution function $cdf(x)$, and quantile function $ppf(p)$.

In the initial stage of sampling, the watermark information c_k with length of k after error correction coding and random stream key encryption is divided into k groups. Each group is mapped to an integer value $y \in [0, 2^k - 1]$, which follows a discrete uniform distribution, i.e., $p(y = i) = \frac{1}{2^k} \quad (i = 0, 1, \ldots, 2^k - 1)$. The standard Gaussian distribution is partitioned into 2^k intervals of equal probability. When $y = i$, the watermarked latent feature z_T^s is sampled

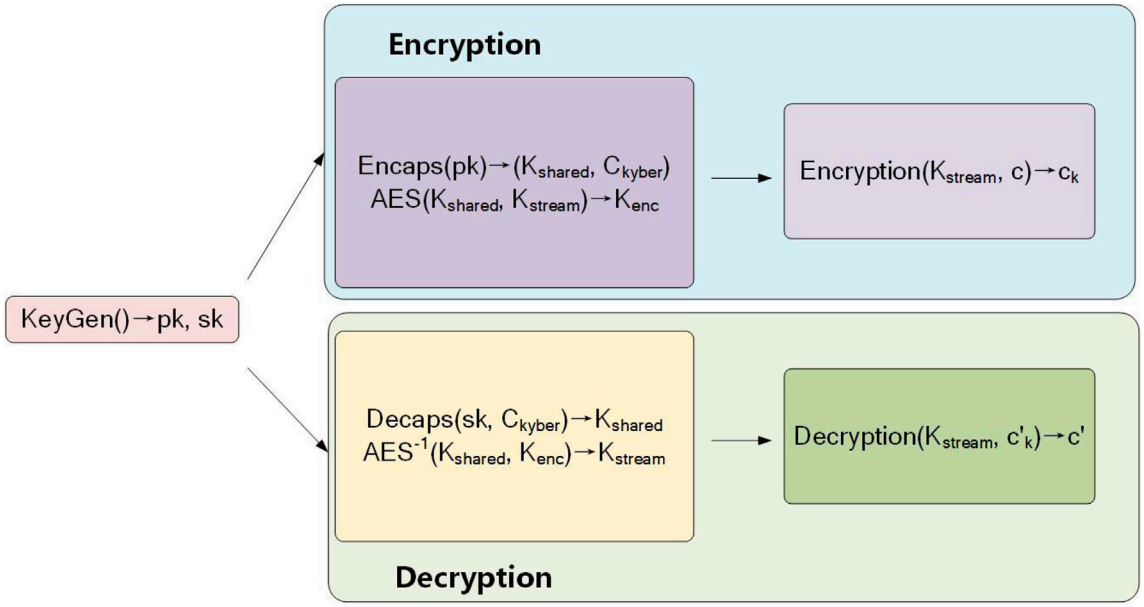

FIGURE 3
Schematic of the watermark encryption and decryption process based on the Kyber key-encapsulation mechanism. During encryption, a key pair (pk,sk) is first generated via Kyber.KeyGen. The stream key K_{stream} is then protected by Kyber. Encaps to produce a ciphertext C_{kyber} and a shared key K_{shared}; the latter is used in a subsequent *AES* symmetric encryption to obtain K_{enc}. Finally, K_{stream} and encoded watermark c are combined via a bitwise XOR operation to produce the secure watermark data c_k. The decryption phase reverses these steps: Kyber.Decaps recovers K_{shared} from C_{kyber} using the private key sk, after which AES^{-1} reconstructs K_{stream}. The watermark is then retrieved by decrypting c'_k with the stream key K_{stream} via a bitwise XOR operation. The resulting noisy codeword c' is then output for subsequent error correction and decoding.

from the conditional distribution of the i-th interval:

$$p(z_T^s|y=i)=\begin{cases}2^k\cdot f(z_T^s) & ppf\left(\frac{i}{2^k}\right)\le z_T^s\le ppf\left(\frac{i+1}{2^k}\right)\\ 0 & \text{otherwise}\end{cases} \tag{27}$$

The probability distribution of z_T^s is given by the following formula:

$$p(z_T^s)=\sum_{i=0}^{2^k-1}p(z_T^s|y=i)P(y=i)=f(z_T^s) \tag{28}$$

Equation 28 shows that z_T^s obeys the same distribution of the random sampling potential representation $z_T\sim\mathcal{N}(0,I)$. Next, we describe the sampling method in detail.

We can get the cumulative distribution function of the equation according to the above:

$$F(z_T^s|y=i)=\begin{cases}0 & z_T^s<ppf\left(\frac{i}{2^k}\right)\\ 2^k\cdot cdf(z_T^s)-i & ppf\left(\frac{i}{2^k}\right)\le z_T^s\le ppf\left(\frac{i+1}{2^k}\right)\\ 1 & z_T^s>ppf\left(\frac{i+1}{2^k}\right)\end{cases} \tag{29}$$

When the condition $y=i$ holds, the latent feature z_T^s is sampled from the interval $\left[ppf\left(\frac{i}{2^k}\right),ppf\left(\frac{i+1}{2^k}\right)\right]$. Sampling from the conditional cdf $F(z_T^s\mid y=i)$ provides a direct method for obtaining z_T^s. Since $F(z_T^s\mid y=i)$ is $[0,1]$ takes values in $[0,1]$, sampling from it is equivalent to sampling from a standard uniform distribution, denoted as $u=F(z_T^s\mid y=i)\sim U(0,1)$. By shifting each term of Equation 29 and considering the inverse function relationship between the cumulative distribution function and the probability density function, Equation 30 describes the sampling process of the hidden representation z_T^s of the watermark driven by the random watermark c_1.

$$z_T^s=ppf\left(\frac{u+i}{2^l}\right) \tag{30}$$

3.2 Watermark extraction

The watermark extraction process is designed to reliably recover the embedded watermark from potentially distorted images by systematically inverting the embedding operations. It proceeds through three sequential stages: latent inversion, distribution-preserving bit recovery, and layered error-correction decoding. These stages progressively suppress distortions to restore the original watermark sequence.The details are as follows.

First, the watermarked image X'^s is encoded into the latent space using the Stable Diffusion encoder $\mathcal{E}$, yielding the latent feature $z_0'^s=\mathcal{E}(X'^s)$. Subsequently, the DDIM inversion process is applied to predict the cumulative noise added during the forward process. This method deterministically reconstructs the noise-addition path, ensuring that the final latent feature approximates the original, i.e., $z_T'^s\approx z_T^s$.

Watermark information extraction from latent features: After obtaining $z_T'^s$, the watermark integer value is first recovered through the inverse quantile function operation:

$$i=\left[2^k\cdot cdf(z_T'^s)\right] \tag{31}$$

where i denotes the watermark bit recovered after decryption and reverse diffusion.

Following the transformation of the message into binary format, a majority voting scheme is employed to decode the repetition-encoded information. The value for each bit position is ascertained based on the majority value across all repetition instances. This process yields the reconstructed repetition-encoded information c'.

For BCH-based decoding, the received codeword c' is represented as the polynomial $c'(x)$ and may contain errors. On the finite field $GF(2^m)$, the syndromes are first calculated using Equation 8. Based on these computed values, it is determined whether solving the error-locator polynomial is necessary in the subsequent step. If required, error correction is performed using Equation 9, ultimately yielding the decoded watermark information u'.

When LDPC decoding is adopted, the received encoded information c', which may contain errors, is first converted to LLR values using Equation 14. These LLR values are stored in a message matrix that shares the same dimensions as the parity-check matrix H, serving as the initial inputs to the variable nodes. The algorithm then enters an iterative decoding process. Each iteration comprises two stages: check node updating and variable node updating. During the check node update phase, each check node j gathers messages from all connected variable nodes except the target node i (denoted as the set $N(j)\backslash i$. It then performs minimum value filtering and sign propagation: Firstly, the minimum ($\min_1$) and the second minimum ($\min_2$) of the absolute value of the adjacent message are found. If the target node i corresponds to $\min_1$, the output magnitude is set to $\min_2$; otherwise, it is set to $\min_1$. Concurrently, the output sign is determined by taking the product of the signs of all incoming messages, as specified in Equation 15. During the variable node update phase, each variable node i combines the initial channel LLR with messages from all connected check nodes except the target node j, as described by Equation 16. After either completing the predetermined number of iterations or meeting the convergence criteria, the decoding process proceeds to the decision phase, implemented using Equation 17. Finally, the decoded watermark information u' is recovered based on the decision values obtained. For the combined LDPC and BCH decoding scheme, the LDPC decoding process is first performed to obtain the intermediate decoded information c'_1. This output then serves as the input to the BCH decoder, ultimately yielding the final recovered watermark information u'.

4 Experiments

This section presents the experimental analysis, which comprises the experimental setup, a comprehensive evaluation of the proposed approach, comparisons with baseline methods, and a performance analysis of different error correction codes.

4.1 Experimental settings

This section outlines the experimental design for evaluating the robustness, visual quality, and security of the proposed watermarking framework. The specific configurations detailed include the diffusion model architecture, sampling strategy, dataset selection, and watermark payload size. These established settings provide a consistent baseline for analyzing the impact of different ECC schemes and embedding mechanisms on watermark extraction accuracy and image fidelity.

Implemental Settings: We employed the Stable Diffusion model to evaluate the efficacy of the watermarking method. The generated watermark images had a resolution of 512×512 pixels, with a latent spatial dimension of $4 \times 64 \times 64$. We employed the Stable-Diffusion-Prompt dataset and the DPM-solver scheduler to perform sampling for a total of 50 steps. During the extraction stage, DDIM inversion was performed using the same number of steps and empty text prompts.

Robustness Evaluation against Image Processing: We evaluated the robustness of various watermarking methods against common image distortions. The evaluation was conducted on 1000 generated watermarked images. The specific parameters for each distortion method are detailed in Figure 4.

For baseline comparisons, this study selected the following representative methods: including the post-processing techniques DwtDct, DwtDctSvd [18], and RivaGAN [19]; the generative method Stable Signature [23]; and the latent feature-based technique Latent Watermark [43] and Gaussian Shading [12]. These baselines were compared against the concatenated BCH-repetition code scheme proposed herein. To ensure fair comparison, we standardize the watermark capacity to 256 bits.

4.2 Evaluation metrics

We use the average watermark extraction bit accuracy of all the extracted watermark samples as the watermark accuracy performance index to evaluate our method. To assess the quality of the generated watermarked images, the Fréchet Inception Distance (FID) [44] and CLIP Score [45] were employed as primary metrics. FID assesses the fidelity and variation of generated images by measuring the divergence between their feature distributions and those of real images. The CLIP Score quantifies the degree of semantic alignment between an image and its corresponding text prompt.

4.3 Comparison with baselines

To provide a more rigorous and balanced comparison, we further analyze the robustness performance of the proposed method against baseline approaches under a variety of common image distortions. In addition to reporting average extraction accuracy, we emphasize scenario-specific behaviors to better reflect practical robustness characteristics.

As shown in Table 1, the proposed method achieves overall competitive performance across most attack types. In certain mild distortion scenarios, such as low-intensity compression or resizing, the robustness of our method is comparable to that of the strongest baseline. This observation indicates that the proposed framework does not sacrifice general robustness in favor of specific attack resilience.

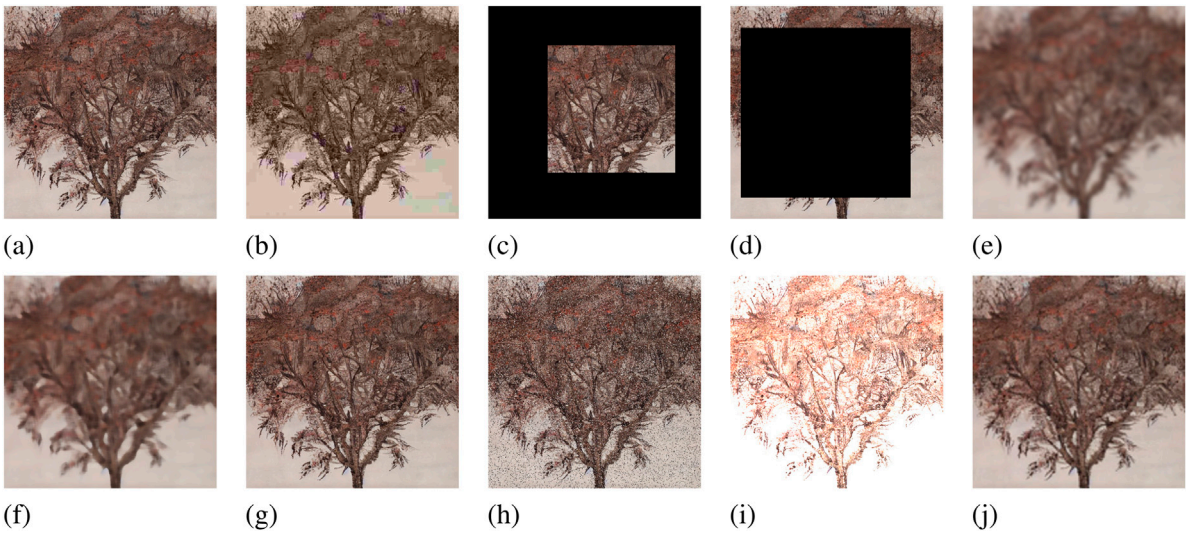

FIGURE 4
Watermarked image is attacked by different noise. **(a)** Watermarked image. **(b)** JPEG, $QF = 10$. **(c)** 60% area Random Crop (RandCr). **(d)** 80% area Random Drop (RandDr). **(e)** Gaussian Blur, $r = 6$ (GauBlur). **(f)** Median Filter, $k = 11$ (MedFilter). **(g)** Gaussian Noise, $\mu = 0$, $\sigma = 0.1$ (GauNoise). **(h)** Salt and Pepper Noise, $p = 0.05$ (S& PNoise). **(i)** Brightness, *factor* = 2.5. **(J)** Resize and restore, *factor* = 0.3.

More importantly, under stochastic and noise-dominated perturbations, the proposed method exhibits consistent and statistically significant improvements over competing approaches. Specifically, under Gaussian blur, Gaussian noise, random drop, and salt-and-pepper noise, the extraction accuracy of our method surpasses the baselines by a noticeable margin. These gains can be attributed to the two-stage error correction mechanism introduced in Section 3.1.1, which combines block-level ECC with repetition coding to jointly address both large-scale corruption and fine-grained random bit fluctuations.

In contrast, baseline methods that rely on single-layer error correction or direct latent perturbation are more sensitive to random and impulsive noise. While they may perform competitively in structured or deterministic distortions, their robustness degrades more rapidly when confronted with unpredictable bit-level perturbations. This difference explains why competing methods may match or slightly outperform our approach in some controlled settings, yet fall behind in noise-intensive scenarios.

Overall, this analysis highlights that the primary advantage of the proposed framework lies not merely in average performance, but in its robustness stability across challenging and highly stochastic attack conditions, which are common in real-world image dissemination pipelines. These results support the claim that the proposed method offers a more reliable watermarking solution under practical and adverse conditions.

4.4 Comparison with other ECC

The performance of three concatenated schemes—BCH-repetition, LDPC-repetition, and a hybrid LDPC-BCH-repetition structure—was compared and analyzed. Their anti-distortion robustness, extraction accuracy, and impact on generated image quality were systematically evaluated.

As indicated in Table 2, the BCH-repetition code scheme maintains superior watermark extraction bit accuracy across various distortion conditions. Under Gaussian blur distortion, its extraction accuracy exceeds that of the LDPC-BCH-repetition method by 3.96%. This enhanced performance stems from the statistical nature of errors in the latent space: perturbations caused by diffusion inversion and image-level attacks predominantly manifest as random, sparse bit flips, not as burst or symbol-level errors. In such an error regime, BCH codes offer a distinct advantage. Their compact codewords enable a higher repetition factor, effectively reducing the bit-error rate prior to decoding. In contrast, LDPC-based schemes generate longer codewords. This limits the available repetition count and increases the probability that the initial error rate will exceed the threshold necessary for iterative convergence. Moreover, BCH decoding employs deterministic algebraic correction for up to terrors, whereas LDPC decoding relies on belief propagation. The latter can become unstable when the initial log-likelihood ratios are unreliable. Consequently, the BCH-repetition scheme is inherently better suited to the error patterns in latent diffusion spaces, achieving an optimal balance between redundancy efficiency and robust error correction.

4.4.1 Quality assessment

As summarized in Table 3, the achieved FID and CLIP scores closely approach the watermark-free baseline values, with the minimal discrepancy between these metrics indicating that the method does not substantially compromise image quality.

4.5 Unauthorized extraction attack

Beyond evaluations of robustness, we designed security experiments to test resistance against key-less watermark extraction. This involved a large-scale unauthorized extraction attempt on

TABLE 1 Watermark extraction bit accuracy under different distortion conditions. Although the error-correcting code-encoded watermark slightly underperforms the Gaussian Shading technique in certain individual distortion cases, it demonstrates superior overall robustness and stability when considering all noise types collectively.

Methods	Noise										
	None	GauBlur	GauNoise	JPEG	MedFilter	RandDr	RandCr	Brightness	S and PNoise	Resize	Average
DwtDct [18]	0.8056	0.5001	0.4696	0.5021	0.5096	0.5541	0.7790	0.5097	0.5723	0.5189	0.5721
DwtDctSvd [18]	0.9995	0.5667	0.5241	0.4917	0.7968	0.5862	0.8247	0.5093	0.5191	0.8892	0.6707
RivaGAN [19]	0.9868	0.5983	0.7143	0.5865	0.8879	0.9685	0.9752	0.8411	0.8558	0.9736	0.8388
Stable Signature [23]	0.9983	0.4079	0.5389	0.5724	0.5223	0.9766	0.9926	0.9533	0.6743	0.5893	0.7226
Latent Watermark [43]	0.9996	0.8163	0.6869	0.8780	0.9209	0.5965	0.6328	0.9703	0.7320	0.9852	0.8219
Gaussian Shading [12]	0.9999	0.9192	0.8619	0.9448	0.9839	0.9671	0.9795	0.9724	0.9360	0.9958	0.9563
ECC (the proposed)	0.9999	**0.9627**	**0.8736**	**0.9488**	0.9829	**0.9778**	0.9794	0.9724	**0.9477**	0.9961	0.9641

Bolded to indicate optimal performance.

TABLE 2 The watermark extraction bit accuracy achieved by different error-correcting and repetition code concatenation schemes was compared. Results demonstrate that BCH–repetition achieves the highest overall robustness across distortions.

Methods	Noise										
	None	GauBlur	GauNoise	JPEG	MedFilter	RandDr	RandCr	Brightness	S and PNoise	Resize	Average
LDPC	0.9999	0.9167	0.8658	0.9485	0.9832	0.9602	0.9583	0.9721	0.9399	0.9951	0.9540
LDPC + BCH	0.9999	0.9231	0.8705	0.9508	0.9834	0.9608	0.9622	0.9727	0.9399	0.9956	0.9559
BCH	0.9999	**0.9627**	**0.8736**	**0.9488**	0.9829	**0.9778**	0.9794	0.9724	**0.9477**	0.9961	0.9641

Bolded to indicate optimal performance.

10.3389/fphy.2025.1750515

TABLE 3 FID and CLIP Scores assess the image quality of the error-correcting watermarking scheme.

Methods	FID	CLIP-score
Stable diffusion	25.23	0.3629
LDPC	24.50	0.3642
LDPC + BCH	24.78	0.3640
BCH	24.82	0.3608

N = 1000 images. The adversary in this simulation is granted full system knowledge (the diffusion model, DDIM inversion, and ECC parameters) but is denied access to the Kyber private key.

For each image, the attacker first applies DDIM inversion to obtain the latent feature $z_T'^s$. They then attempt to infer the watermark bits directly by mapping the latent values to binary sequences, thus trying to circumvent the cryptographic protection entirely. The recovered bit sequences are compared with the corresponding ground-truth watermark bits.

$$\mathbf{ACC} = 0.5017 \pm 0.0083 \tag{32}$$

The watermark extraction bit accuracy remained statistically indistinguishable from the random-guess baseline of 0.50. The negligible variance observed across 1000 trials indicates no measurable information leakage from the encrypted latent watermark. These results collectively demonstrate that, even under a strong inversion-based attack, unauthorized extraction remains infeasible without the requisite cryptographic key.

4.6 Discussion on limitations

Despite its demonstrated effectiveness and robustness, the proposed framework has several limitations to discuss.

First, computational overhead presents a practical constraint. The framework relies on diffusion-based image generation and DDIM inversion, processes that are computationally more intensive than conventional spatial-domain watermarking. Specifically, the watermark extraction requires a DDIM inversion step, which introduces non-negligible latency. While acceptable for offline verification or forensic analysis, this overhead necessitates further optimization for real-time or large-scale deployment.

Second, the framework's security guarantees depend on specific assumptions. Although post-quantum security is provided via the Kyber key encapsulation mechanism, the system's security critically depends on secure key management and trusted distribution channels. Compromise or improper management of private keys could therefore undermine the cryptographic protection layer. Moreover, the current security analysis focuses on unauthorized extraction and does not address active adversaries who might attempt to forge watermarks using compromised credentials.

Finally, the current design emphasizes robustness and security at the expense of payload capacity. The redundancy introduced by error-correcting and repetition coding limits the maximum watermark length that can be embedded without degrading visual quality. Consequently, developing more efficient coding strategies to improve this robustness–capacity trade-off remains an open challenge.

These limitations point to clear directions for future work: computational optimization, stronger adversarial security models, and more efficient encoding schemes.

5 Conclusion

This work presents a unified post-quantum–resilient watermarking framework for diffusion models that integrates latent-space watermark embedding, error-correcting codes, and secure key encapsulation. The proposed method embeds watermark signals directly into the latent features of diffusion models, thereby avoiding pixel-level degradation and maintaining high visual fidelity. By incorporating ECC into the embedding pipeline, the system gains strong resilience against a wide spectrum of distortions—including noise, compression, and spatial manipulation—without requiring model retraining or architectural modification. In parallel, the adoption of the Kyber key encapsulation mechanism ensures secure watermark key management under quantum-era threat models.

Extensive experiments demonstrate that the framework achieves high watermark extraction accuracy across challenging attack scenarios while preserving image quality comparable to unwatermarked outputs. The comparative analysis further shows that different ECC configurations offer distinct performance trade-offs, with BCH-based schemes providing particularly strong robustness in latent-space perturbation regimes.

Overall, this study highlights the feasibility and effectiveness of combining latent watermarking with post-quantum cryptography and classical error-correction coding. The resulting system provides a practical pathway for trustworthy AIGC governance, secure provenance tracking, and resilient content authentication in critical infrastructure applications. Future work may explore adaptive ECC allocation, content-aware embedding strategies, and broader integration with multimodal generative models.

Future work will focus on two key directions to extend this research. First, a systematic evaluation of alternative secure key-exchange mechanisms is warranted. This includes hybrid lattice schemes, code-based KEMs, and lightweight authenticated protocols. Such a study would elucidate their deployment trade-offs in heterogeneous infrastructure environments. Second, the security architecture of large-scale generative systems could be strengthened by extending the analysis of watermark-encryption strategies. Promising extensions include adaptive key rotation, hierarchical multi-party key management, and layered encryption models. Pursuing these directions will advance watermarking pipelines toward more comprehensive, resilient, and scalable protection for next-generation AI-generated content ecosystems.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

XH: Conceptualization, Funding acquisition, Methodology, Supervision, Writing – original draft, Writing – review and editing. BZ: Conceptualization, Methodology, Software, Writing – original draft. MA-D: Funding acquisition, Validation, Writing – review and editing. NE-G: Funding acquisition, Validation, Writing – review and editing. MR: Validation, Writing – review and editing. AK: Methodology, Validation, Writing – original draft, Writing – review and editing.

Funding

The author(s) declared that financial support was received for this work and/or its publication. This article has been supported by the start-up funding from Guangdong Polytechnic Normal University with Grant No. 2023SDKYA004. The current work was assisted financially by the Dean of Scientific Research at King Khalid University via the Large Group Project under grant number RGP. 2/23/46.

Acknowledgements

The authors extend their appreciation to the Deanship of Scientific Research at King Khalid University for funding this work through large Groups Project under grant number RGP. 2/23/46.

Conflict of interest

The author(s) declared that this work was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement

The author(s) declared that generative AI was not used in the creation of this manuscript.

Any alternative text (alt text) provided alongside figures in this article has been generated by Frontiers with the support of artificial intelligence and reasonable efforts have been made to ensure accuracy, including review by the authors wherever possible. If you identify any issues, please contact us.

Publisher's note

References

1. Ho J, Jain A, Abbeel P. Denoising diffusion probabilistic models. In: *Proceedings of the 34th international conference on neural information processing systems*. Red Hook, NY, USA: Curran Associates Inc. (2020). NIPS '20.

2. Sohl-Dickstein J, Weiss E, Maheswaranathan N, Ganguli S. Deep unsupervised learning using nonequilibrium thermodynamics. In: F Bach, D Blei, editors. *Proceedings of the 32nd international conference on machine learning. Lille, France: PMLR), vol. 37 of proceedings of machine learning research* (2015). p. 2256–65.

3. Song J, Meng C, Ermon S (2020). Denoising diffusion implicit models. *ArXiv* abs/2010.02502

4. Song Y, Ermon S. Generative modeling by estimating gradients of the data distribution. In: *Neural information processing systems* (2019).

5. Song Y, Sohl-Dickstein J, Kingma DP, Kumar A, Ermon S, Poole B. Score-based generative modeling through stochastic differential equations. In: *International conference on learning representations* (2021).

6. Xu T, Zhang P, Huang Q, Zhang H, Gan Z, Huang X, et al. Attngan: fine-grained text to image generation with attentional generative adversarial networks. In: *2018 IEEE/CVF conference on computer vision and pattern recognition* (2018). p. 1316–24. doi:10.1109/CVPR.2018.00143

7. Qiao T, Zhang J, Xu D, Tao D. Mirrorgan: learning text-to-image generation by redescription. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)* (2019).

8. Guo D, Chen H, Wu R, Wang Y. Aigc challenges and opportunities related to public safety: a case study of chatgpt. *J Saf Sci Resilience* (2023) 4:329–39. doi:10.1016/j.jnlssr.2023.08.001

9. Frolov S, Hinz T, Raue F, Hees J, Dengel A. Adversarial text-to-image synthesis: a review. *Neural Networks* (2021) 144:187–209. doi:10.1016/j.neunet.2021.07.019

10. van Schyndel R, Tirkel A, Osborne C. A digital watermark. *Proc 1st Int Conf Image Process* (1994) 2:86–90. doi:10.1109/ICIP.1994.413536

11. Yuan Z, Li L, Wang Z, Zhang X. Watermarking for stable diffusion models. *IEEE Internet Things J* (2024) 11:35238–49. doi:10.1109/JIOT.2024.3434656

12. Yang Z, Zeng K, Chen K, Fang H, Zhang WM, Yu N. Gaussian shading: provable performance-lossless image watermarking for diffusion models. *CoRR abs/2404* (2024):04956.

13. Li Y, Liao X, Wu X. Screen-shooting resistant watermarking with grayscale deviation simulation. *IEEE Trans Multimedia* (2024) 26:10908–23. doi:10.1109/TMM.2024.3415415

14. He W, Cai Z, Wang Y. High-fidelity reversible image watermarking based on effective prediction error-pairs modification. *IEEE Trans Multimedia* (2021) 23:52–63. doi:10.1109/TMM.2020.2982042

15. Qin C, Li X, Zhang Z, Li F, Zhang X, Feng G. Print-camera resistant image watermarking with deep noise simulation and constrained learning. *IEEE Trans Multimedia* (2024) 26:2164–77. doi:10.1109/TMM.2023.3293272

16. Zhong X, Huang P-C, Mastorakis S, Shih FY. An automated and robust image watermarking scheme based on deep neural networks. *IEEE Trans Multimedia* (2021) 23:1951–61. doi:10.1109/TMM.2020.3006415

17. Fang H, Jia Z, Qiu Y, Zhang J, Zhang W, Chang E-C. De-end: decoder-driven watermarking network. *IEEE Trans Multimedia* (2023) 25:7571–81. doi:10.1109/TMM.2022.3223559

18. Cox I, Miller M, Bloom J, Fridrich J, Kalker T. *Digital watermarking and steganography*. 2 edn. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. (2007).

19. Zhang KA, Xu L, Cuesta-Infante A, Veeramachaneni K (2019). Robust invisible video watermarking with attention. *arXiv e-prints*.

20. Yin Z, Yin H, Zhang X. Neural network fragile watermarking with no model performance degradation. In: *2022 IEEE international conference on image processing (ICIP)* (2022). p. 3958–62. doi:10.1109/ICIP46576.2022.9897413

21. [Dataset] Bui T, Yu N, Collomosse J (2022). Repmix: representation mixing for robust attribution of synthesized images, 146, 63. doi:10.1007/978-3-031-19781-9_9

22. Cui Y, Ren J, Xu H, He P, Liu H, Sun L, et al. Diffusionshield: a watermark for data copyright protection against generative diffusion models. *SIGKDD Explor Newsl* (2025) 26:60–75. doi:10.1145/3715073.3715079

23. Fernandez P, Couairon G, Jégou H, Douze M, Furon T. The stable signature: rooting watermarks in latent diffusion models. In: *2023 IEEE/CVF international conference on computer vision (ICCV)* (2023). p. 22409–20. doi:10.1109/ICCV51070.2023.02053

24. Feng W, Zhou W, He J, Zhang J, Wei T, Li G, et al. Aqualora: toward white-box protection for customized stable diffusion models via watermark lora. In: *Proceedings of the 41st international conference on machine learning*. JMLR.org (2024). ICML'24.

25. Kishore V, Chen X, Wang Y, Li B, Weinberger KQ. Fixed neural network steganography: train the images, not the network. In: *International conference on learning representations* Vienna, Austria: JMLR.org (2022).

26. Liu Y, Li Z, Backes M, Shen Y, Zhang Y. Watermarking diffusion model. *CoRR abs/2305* (2023):12502. doi:10.48550/ARXIV.2305.12502

27. Xiong C, Qin C, Feng G, Zhang X. Flexible and secure watermarking for latent diffusion model. In: *Proceedings of the 31st ACM international conference on multimedia*. New York, NY, USA: Association for Computing Machinery), MM '23 (2023). p. 1668–76. doi:10.1145/3581783.3612448

28. Zhao Y, Pang T, Du C, Yang X, Cheung N, Lin M. A recipe for watermarking diffusion models. *CoRR abs/2303* (2023):10137. doi:10.48550/ARXIV.2303.10137

29. Moon TK (2005). *Error correction coding: mathematical methods and algorithms/T.K. moon* (error correction coding: mathematical methods and algorithms)

30. Clark and GeorgeC (1981). *Error-correction coding for digital communications* (Error-correction coding for digital communications)

31. Cai Z, Hao J, Tan P, Sun S, Chin P. Efficient encoding of ieee 802.11n ldpc codes. *Electronics Lett* (2006) 42:1471–2. doi:10.1049/el:20063126

32. Bos JW, Ducas L, Kiltz E, Lepoint T, Lyubashevsky V, Schanck JM, et al. Crystals - kyber: a cca-secure module-lattice-based kem. In: *2018 IEEE European symposium on security and privacy (EuroS&P)* (2017). p. 353–67.

33. Rombach R, Blattmann A, Lorenz D, Esser P, Ommer B. High-resolution image synthesis with latent diffusion models. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)* (2022). p. 10684–95.

34. Wen Y, Kirchenbauer J, Geiping J, Goldstein T (2023). Tree-ring watermarks: fingerprints for diffusion images that are invisible and robust. *ArXiv* abs/2305.20030.

35. Lei L, Gai K, Yu J, Zhu L. Diffusetrace: a transparent and flexible watermarking scheme for latent diffusion model. *CoRR abs/2405* (2024):02696. doi:10.48550/ARXIV.2405.02696

36. Chen X, Reed IS. *Error-control coding for data networks*. USA: Kluwer Academic Publishers (1999).

37. Johannesson R, Zigangirov KS. *Fundamentals of convolutional coding*. John Wiley and Sons (2015).

38. Dholakia A. *Introduction to convolutional codes with applications*. Springer Science and Business Media (1994).

39. Wang J, Zhao J, Li L, Wang Z, Wu H, Wu D. Robust blind video watermarking based on ring tensor and bch coding. *IEEE Internet Things J* (2024) 11:40743–56. doi:10.1109/JIOT.2024.3453960

40. Ghouti L, Landolsi MA. Robust color image watermarking using the spatio-chromatic fourier transform and semi-random ldpc codes. In: *2012 international conference on computer and communication engineering (ICCCE)* (2012). p. 349–53. doi:10.1109/ICCCE.2012.6271209

41. Maammar NE, Bri S, Foshi J. Layered offset min-sum decoding for low density parity check codes. In: *2018 international symposium on advanced electrical and communication technologies (ISAECT)* (2018). p. 1–5. doi:10.1109/ISAECT.2018.8618780

42. [Dataset Kitagawa F, Nishimaki R. *White-box watermarking signatures against quantum adversaries and its applications*. Cryptology ePrint Archive (2025). Paper 2025/265.

43. Meng Z, Peng B, Dong J. Latent watermark: inject and detect watermarks in latent diffusion space. *IEEE Trans Multimedia* Vienna, Austria: JMLR.org (2025) 27:3399–410. doi:10.1109/TMM.2025.3535300

44. Heusel M, Ramsauer H, Unterthiner T, Nessler B, Hochreiter S. Gans trained by a two time-scale update rule converge to a local nash equilibrium, *Proc 31st Int Conf Neural Inf Process Syst*. 17. (2017). p. 6629–40.

45. Radford A, Kim JW, Hallacy C, Ramesh A, Goh G, Agarwal S, et al. Learning transferable visual models from natural language supervision. In: M Meila, T Zhang, editors. *Proceedings of the 38th international conference on machine learning. (PMLR), vol. 139 of proceedings of machine learning research* (2021). p. 8748–63.