TRAJBIND: IN-TRAJECTORY CONTENT BINDING FOR FORGERY-RESISTANT DIFFUSION WATERMARKING

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing diffusion watermarking methods mainly consider distortion and removal attacks, while forgery attacks pose a different threat: an adversary may transfer valid watermark evidence to content that was not generated by the watermarking system. We introduce TRAJBIND, an in-trajectory, content-bound watermarking method that ties the watermark pattern to the generated content. At a late denoising step, TRAJBIND extracts a compact content code from the model's predicted clean image and combines it with a secret key to determine the watermark pattern embedded during the remaining denoising steps. This design requires no auxiliary generation or external semantic models, while verification only requires the received image. We further provide a general formulation of diffusion-watermark forgery that covers both detection-level and exact-message attacks, together with a theoretical analysis that bounds detection-level forgery success in terms of content-code collision. Across five existing forgery attacks, TRAJBIND has lower worst-case forgery success than each of the 12 baseline watermarking methods while maintaining robust watermark detection and message recovery under benign transformations and removal attacks.

1 INTRODUCTION

Watermarks for diffusion models are intended to provide evidence that generated content comes from a particular system after it leaves the generation platform (Nguyen-Le et al., 2025). Compared with post-hoc watermarking, in-generation methods (Fernandez et al., 2023; Wen et al., 2023; Yang et al., 2024b; Gunn et al., 2025a; Nguyen-Le et al., 2026) embed signals directly into the generation process, using latent representations and denoising dynamics to improve robustness to image transformations. Existing studies have mainly considered *distortion attacks*, which apply content-preserving transformations such as compression or resizing, and *removal attacks*, which attempt to erase the watermark from a genuinely watermarked image (An et al., 2024). However, reliable provenance requires more than preserving watermark evidence under such attacks. The EU AI Act emphasizes effective, robust, and reliable marking of AI-generated content (European Parliament and Council of the European Union, 2024). Such reliability is compromised if valid watermark evidence can be transferred to content that the provider never generated.

Recently, ***forgery attacks*** have exposed this complementary threat. Rather than removing watermark evidence, these attacks transfer or reproduce it on attacker-chosen content that was never generated by the legitimate watermark system (Yang et al., 2024a; Souček et al., 2026; Dong et al., 2026; Müller et al., 2025; Jain et al., 2025). A successful forgery can therefore cause the watermark system to accept unauthentic content as watermarked, leading to false attribution to a legitimate provider or user. Existing attacks use different mechanisms, including averaging shared watermark patterns (Yang et al., 2024a), copying evidence from a single watermarked image (Souček et al., 2026; Jain et al., 2025), learning transferable watermark characteristics (Dong et al., 2026), and reusing watermark-bearing latent representations (Müller et al., 2025). Despite these differences, they exploit a common weakness of existing watermark methods: *watermark evidence is associated with a user or secret key but is insufficiently bound to the specific content being verified.* As a result, evidence extracted from legitimate watermarked images can potentially be reused across different images.

This observation suggests that forgery resistance requires the watermark pattern itself to depend on the image content. If images generated under the same secret key carry different content-dependent

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

watermark patterns, evidence obtained from one image should not remain valid for another image with different content. Recent methods pursue this direction by binding watermarks to image semantics or content features (Arabi et al., 2025; Zhang et al., 2026; Gao et al., 2026; Xie et al., 2026; Zhu et al., 2026). However, content binding introduces a key challenge for in-generation watermarking: the watermark pattern should depend on the final image, but the final image is not available when the watermark is embedded. Most existing methods resolve this by first producing a separate image and reading its content with an external model (Arabi et al., 2025; Gao et al., 2026; Zhang et al., 2026; Zhu et al., 2026); others rely on information stored outside the generation trajectory (Xie et al., 2026).

We address this challenge with a simple observation: *at a late denoising step, the model's predicted clean image already captures the main structure of the final output*. This intermediate prediction can therefore provide content information before generation is complete. Based on this observation, we introduce **TRAJBIND**, an in-trajectory, content-bound watermarking method that derives the binding information directly from the diffusion trajectory. TRAJBIND obtains a compact content code from the predicted clean image at an intermediate denoising step and combines this code with the secret key to determine the watermark pattern embedded during the remaining denoising steps. Consequently, images generated with the same secret key can carry different watermark patterns when their content differs. At verification, the receiver obtains a content-code estimate from the received image, constructs a small set of nearby candidate codes, and checks the watermark pattern associated with each candidate. It does not choose a single content code. Evidence copied from another watermarked image therefore becomes inconsistent with the target content unless a content code used by an observed watermarked image enters the candidate set of the received image. Importantly, this design performs content binding within a single generation trajectory, without an additional generation pass or external semantic model.

Contributions. Our main contributions are three-fold: (1) We provide a general formulation of diffusion-watermark forgery that covers both detection-level and exact-message attacks under clearly defined adversary capabilities; (2) We propose TRAJBIND, an in-trajectory, content-bound watermarking method that obtains a compact content code from the predicted clean image and uses it with the secret key to determine the watermark pattern within the same denoising trajectory; (3) We provide a theoretical analysis showing that, with a normalized verification score, detection-level forgery success is bounded by the probability of content-code collision plus a term that depends only on the candidate-list size and the threshold. Experiments against five forgery attacks and 12 watermarking methods show that TRAJBIND provides strong resistance to forgery attacks while maintaining robustness to common image transformations and removal attacks.

2 RELATED WORK

In-Generation Watermarking. In-generation watermarking embeds watermarks during image generation rather than after generation (Tancik et al., 2020). Existing methods modify model components (Fernandez et al., 2023) or the latent sampling and denoising process (Wen et al., 2023; Yang et al., 2024b; Gunn et al., 2025b). While these methods are robust to common distortions and removal attacks, their watermarks are generally independent of image content. Recent methods instead bind watermarks to image semantics or content features (Arabi et al., 2025; Zhang et al., 2026; Gao et al., 2026; Xie et al., 2026; Zhou et al., 2026). We provide a detailed comparison with these methods in Appendix B.

Forgery Attacks. Forgery attacks aim to place valid watermarks on adversary-chosen images. Existing attacks estimate shared watermark patterns from multiple watermarked images (Yang et al., 2024a), transfer watermarks from a single reference through latent optimization (Souček et al., 2026; Jain et al., 2025), reuse inverted noise for imprinting or reprompting (Müller et al., 2025), or learn transferable watermark patterns from watermarked samples (Dong et al., 2026). These attacks motivate binding the watermark to image content. TRAJBIND performs this binding within the same denoising trajectory by using a content code to determine the watermark pattern.

3 THREAT MODEL AND PROBLEM FORMULATION

3.1 FORMULATION OF WATERMARK FORGERY

Let $\mathcal{X} \subseteq [0,1]^{H\times W\times C}$ be the image space and $\mathcal{M} = \{0,1\}^k$ the watermark-message space. A watermarking system is $\mathsf{WM}_\theta = (\mathrm{Emb}_\theta, \mathrm{Ext}_\theta, \mathrm{Ver}_\theta)$, where $\mathrm{Ext}_\theta(x) \in \mathcal{M} \cup \{\perp\}$ and

$\mathrm{Ver}_\theta(x, m; \tau) = \mathbf{1}\{S_\theta(x, m) \geq \tau\}$, where $S_\theta(x, m)$ measures evidence that x carries message m, and τ is the detection threshold.

An adversary $\mathcal{A}$ attempts to forge evidence for a target watermark message $m^\star$ using observations $\mathcal{O}_\mathcal{A}$ available under an access policy Π, which specifies accessible samples, models, and generation or verification queries. Given a target image x^{tgt}, the adversary produces $x^f = \mathcal{A}(x^{\mathrm{tgt}}, m^\star; \mathcal{O}_\mathcal{A})$ to maximize watermark evidence while preserving the target content:

$$\max_{x^f \in \mathcal{X}} \quad S_\theta(x^f, m^\star) \quad \text{s.t.} \quad d_\mathcal{X}(x^f, x^{\mathrm{tgt}}) \leq \varepsilon \quad \text{and} \quad d_{\mathrm{sem}}\left(\phi_{\mathrm{sem}}(x^f), \phi_{\mathrm{sem}}(x^{\mathrm{tgt}})\right) \leq \gamma, \quad (1)$$

where $d_\mathcal{X}$ measures image distortion, d_{sem} measures semantic discrepancy, and $\phi_{\mathrm{sem}} : \mathcal{X} \to \mathbb{R}^{d_s}$ is a semantic feature extractor. A successful forgery satisfies $S_\theta(x^f, m^\star) \geq \tau$.

Without access to S_θ, the adversary constructs a surrogate $J_{\mathcal{O}_\mathcal{A}}$ and instead solves:

$$\max_{x^f \in \mathcal{X}} \quad J_{\mathcal{O}_\mathcal{A}}(x^f, m^\star) \quad \text{s.t.} \quad d_\mathcal{X}(x^f, x^{\mathrm{tgt}}) \leq \varepsilon \quad \text{and} \quad d_{\mathrm{sem}}\left(\phi_{\mathrm{sem}}(x^f), \phi_{\mathrm{sem}}(x^{\mathrm{tgt}})\right) \leq \gamma. \quad (2)$$

Existing forgery attacks mainly differ in how this surrogate is constructed and how target content is preserved or controlled (Yang et al., 2024a; Souček et al., 2026; Dong et al., 2026; Müller et al., 2025; Jain et al., 2025). Formulation details of these forgery attacks are provided in Appendix D.

We distinguish detection-level success from exact-message recovery:

$$\mathrm{ASR}_{\mathrm{det}}(\mathcal{A}) = \Pr\left[\mathrm{Ver}_\theta(x^f, m^\star; \tau) = 1\right], \qquad \mathrm{ASR}_{\mathrm{msg}}(\mathcal{A}) = \Pr\left[\mathrm{Ext}_\theta(x^f) = m^\star\right]. \quad (3)$$

3.2 Threat Model

Adversary knowledge and capabilities. The adversary knows the diffusion backbone, watermarking algorithm, selected DCT components, and decoding parameters. It may collect genuine watermarked outputs, choose targets and prompts, and train or optimize surrogate models. It does not know the secret key, the content-code parameters Θ_Φ, or the provider's intermediate generation states. Our primary setting excludes access to the private watermark verifier; query-enabled variants are specified through Π with a bounded query budget.

Adversary objective. The adversary seeks to forge $m^\star$ on an adversary-chosen image while preserving its content, according to Eq. 1, or Eq. 2 when the verifier is unavailable. Success is measured by $\mathrm{ASR}_{\mathrm{det}}$ and $\mathrm{ASR}_{\mathrm{msg}}$ in Eq. 3.

3.3 Forgery-Resistant Watermarking Formulation

We now formulate the objective of a watermarking system designed to resist the forgery attacks defined above. Let $\mathfrak{A}_\Pi$ denote the class of adversaries allowed by Π. A forgery-resistant watermarking system should provide low forgery success, reliable verification of legitimate watermarks, and limited image distortion.

Forgery resistance. Define the worst-case forgery risk by $\mathcal{R}_{\mathrm{forge}}(\mathsf{WM}_\theta; \Pi) = \sup_{\mathcal{A} \in \mathfrak{A}_\Pi} \max\left\{\mathrm{ASR}_{\mathrm{det}}(\mathcal{A}), \mathrm{ASR}_{\mathrm{msg}}(\mathcal{A})\right\}$. Consequently, robustness to forgery requires both $\mathrm{ASR}_{\mathrm{det}}$ and $\mathrm{ASR}_{\mathrm{msg}}$ to remain low for any adversary permitted by Π.

Watermark correctness. Let x^w be an image legitimately watermarked with message m, and let $\mathcal{T}_{\mathrm{ben}}$ be the supported benign transformations. We require $\inf_{T \in \mathcal{T}_{\mathrm{ben}}} \Pr\left[\mathrm{Ver}_\theta(T(x^w), m; \tau) = 1\right] \geq 1 - \beta_{\mathrm{det}}$, and $\inf_{T \in \mathcal{T}_{\mathrm{ben}}} \Pr\left[\mathrm{Ext}_\theta(T(x^w)) = m\right] \geq 1 - \beta_{\mathrm{msg}}$, where β_{det} and β_{msg} bound detection and message-recovery failure, respectively.

Watermark fidelity. For the corresponding unwatermarked output x^0, we require $\mathbb{E}\left[d_\mathcal{X}(x^w, x^0)\right] \leq \varepsilon_{\mathrm{wm}}$.

Combining these requirements, the defense problem is

$$\min_{\theta, \tau} \quad \mathcal{R}_{\mathrm{forge}}(\mathsf{WM}_\theta; \Pi) \quad \text{s.t.} \quad \text{watermark correctness}, \quad \text{watermark fidelity}, \text{ and } \Pr_{x \sim P_{\mathrm{clean}}}\left[\mathrm{Ver}_\theta(x, m; \tau) = 1\right] \leq \alpha, \quad (4)$$

where α is the upper bound on the false-positive rate.

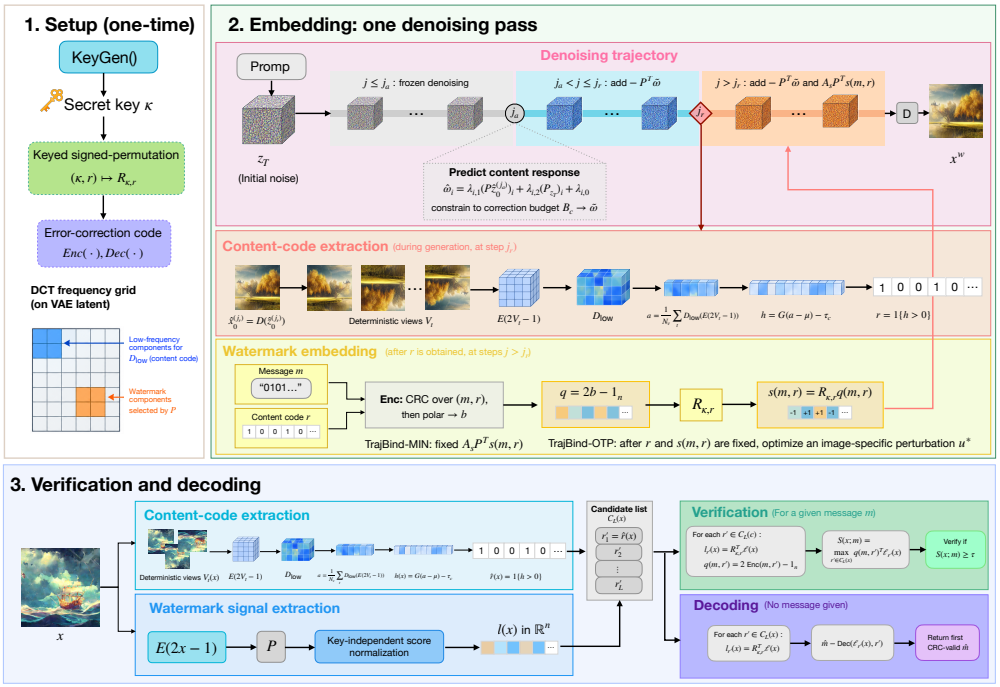


Figure 1: **Overview of TRAJBIND.** During generation, TRAJBIND first obtains a content code from an intermediate image, then uses this code to construct and embed the watermark in the remaining denoising steps. Verification checks whether the watermark matches the received image.

4 METHOD

The central idea of TRAJBIND is to derive a ***content-dependent watermark pattern within the same denoising trajectory*** used to generate the image. Before denoising is complete, the predicted clean image already shows the broad visual structure of the final output. We use this intermediate prediction to obtain a compact content code and then use that code, together with a secret key, to determine how the watermark message is placed during the remaining denoising steps. Throughout this section, m denotes the watermark message, r the content code, b the codeword, q its signed form, and s the signed sequence arranged in the selected DCT components. For simplicity, we write $S(x; m)$ for $S_\theta(x, m)$. Fig. 1 illustrates our method, while full algorithms are provided in Appendix A.6.

Let $m \in \{0,1\}^k$ denote the watermark message, $z_T \in \mathbb{R}^d$ the initial noise, and $\widehat{z}_0^{(j)}$ the predicted clean latent at denoising step j. We index the denoising steps in denoising order as $j = 0, \ldots, J-1$. Under this convention, the correction anchor j_a occurs before content-code extraction at j_r. The frozen VAE decoder D maps latent representations to images in $[0,1]$, whereas the deterministic encoder E takes images normalized to $[-1,1]$.

4.1 SINGLE-PASS CONTENT CODE EXTRACTION

At step j_r, we decode $\widehat{x}_0^{(j_r)} = D(\widehat{z}_0^{(j_r)})$. Its coarse structure provides the content information used for content binding, although subsequent denoising and image transformations can still change the extracted features. Therefore, we use multiple deterministic views and low-frequency DCT coefficients rather than relying on a single image view.

Deterministic views and low-frequency DCT features. For an image x, we construct N_v deterministic views $\{\mathcal{V}_t(x)\}_{t=1}^{N_v}$, including the original image and its transformed versions. Each view is VAE-encoded, and $\mathcal{D}_{\text{low}}$ extracts the selected low-frequency DCT coefficients. We use the low-frequency region because it captures coarse image structure while being less sensitive to small local image changes. It also keeps the content features separate, at embedding time, from the higher-frequency DCT components used to carry the watermark. We average these features across views as $\overline{a}(x) = \frac{1}{N_v}\sum_{t=1}^{N_v} \mathcal{D}_{\text{low}}(E(2\mathcal{V}_t(x) - 1))$, which reduces sensitivity to any individual

view. The averaged feature is then projected as $h(x) = G(\overline{a}(x) - \mu) - \tau_c$, and the content code is obtained by thresholding each component, i.e., $\phi_c(x; \Theta_\Phi) = \mathbf{1}\{h(x) > 0\}$. Here, G projects the averaged features to k_r dimensions, μ is the feature mean, and τ_c contains the decision thresholds. The content-code parameters Θ_Φ also include the projection scales used to order candidate codes during verification. This construction uses the diffusion model's own VAE, without an external semantic model.

Content code. We set $r = \phi_c(\widehat{x}_0^{(j_r)}; \Theta_\Phi) \in \{0,1\}^{k_r}$ and keep it fixed during the remaining denoising steps. For a received image x, the same feature extraction gives $\widehat{r}(x) = \phi_c(x; \Theta_\Phi) = \mathbf{1}\{h(x) > 0\}$. Because r may differ from $\widehat{r}(x)$, the receiver searches nearby candidate codes. The content-code parameters are independent of κ.

4.2 Content-Dependent Watermark Embedding

Content-dependent watermark pattern. We encode the watermark message using a polar error-correction code (Arikan, 2009; Trifonov, 2012) as $b(m, r) = \mathrm{Enc}(m, r) \in \{0,1\}^n$, where the encoding depends on both the watermark message m and the content code r. We convert the encoded codeword to its signed form as $q(m, r) = 2b(m, r) - \mathbf{1}_n$. The secret key κ and content code r define a pseudorandom signed-permutation matrix $R_{\kappa,r} \in \mathbb{R}^{n\times n}$ (Bellare, 2006). The watermark pattern is $s(m, r) = R_{\kappa,r} q(m, r)$, where $R_{\kappa,r}^\top R_{\kappa,r} = I_n$. Consequently, the content code changes the DCT watermark pattern even when the watermark message is fixed. During extraction, the receiver reconstructs $R_{\kappa,r'}$ for each candidate code r' and uses $R_{\kappa,r'}^\top$ to place the receiver scores in codeword order before decoding.

Let $P \in \mathbb{R}^{n\times d}$ contain the normalized DCT basis patterns used for watermark embedding, with $PP^\top = I_n$. These DCT components are selected outside the low-frequency region used to compute the content code. Accordingly, $P^\top$ maps the watermark pattern into the latent space, whereas P extracts the corresponding DCT components during verification. The separation of the two frequency regions applies at embedding time and does not imply exact separation after the subsequent denoising steps and VAE processing.

Content correction. The image content naturally produces nonzero values in the selected DCT components, even before the watermark is embedded. These content-induced values can overlap with the watermark signal and make recovery less reliable. We therefore estimate and compensate for them before extracting the content code. At the correction anchor $j_a < j_r$, the content-induced response of component i is predicted as $\widehat{\omega}_i = \lambda_{i,1}(P\widehat{z}_0^{(j_a)})_i + \lambda_{i,2}(Pz_T)_i + \lambda_{i,0}$. The component-specific coefficients $\lambda_{i,0}, \lambda_{i,1}, \lambda_{i,2}$ are fitted on unwatermarked generations and then fixed. We constrain the norm of the predicted response to the correction budget B_c, yielding $\widetilde{\omega}$, and apply $-\widetilde{\omega}$ to reduce the content-induced responses in the selected DCT components.

Late-step embedding. Content correction starts after j_a. Once the content code r is obtained at j_r, we additionally insert the content-dependent watermark pattern. Specifically, the content-correction update is $\Delta z_{\mathrm{corr}} = -P^\top \widetilde{\omega}$, while the watermark update is $\Delta z_{\mathrm{wm}} = A_s P^\top s(m, r)$, where $A_s > 0$ controls the watermark strength. Importantly, the watermark update is introduced only after r has been obtained, so the content code is not computed from a latent that already contains the content-dependent watermark pattern.

Imperceptibility-aware embedding. TrajBind-Min uses a fixed perturbation that contains both the watermark update and the content correction that remains after r is obtained. Under our denoising schedule, the corresponding combined perturbation is $u_0 = A_s P^\top s(m, r) - \frac{1}{3} P^\top \widetilde{\omega}$. The fixed schedule distributes this perturbation over the remaining denoising steps without per-image optimization. To further reduce visual distortion, TrajBind-Opt optimizes an image-specific perturbation u as

$$u^\star = \arg\min_{u\in\mathcal{U}} \mathcal{D}(u) + \lambda_{\mathrm{wm}} \sum_{T\in\mathcal{G}} \left[\frac{\mathcal{L}_{\mathrm{wm}}(T(x(u)); m, r)}{\mathcal{L}_{\mathrm{wm}}(T(x(u_0)); m, r)} - 1 \right]_+^2, \tag{5}$$

where $x(u)$ is the output obtained by applying u over the remaining denoising steps, $\mathcal{D}(u)$ measures the visual difference from the corresponding unwatermarked image, and $\mathcal{L}_{\mathrm{wm}}$ measures the watermark-recovery loss. During optimization, r, $s(m, r)$, and the model parameters are fixed, while u is optimized over the full latent space. Among several candidate perturbations, we select the one

with the smallest $\mathcal{D}(u)$ that preserves message recovery relative to u_0; otherwise, we use u_0. Detailed settings are given in the App. A.4.

4.3 Watermark Verification

Given a received image x, the receiver obtains the content-code estimate $\widehat{r}(x)$, constructs a small candidate set $\mathcal{C}(x)$, and extracts soft watermark scores $\ell(x) \in \mathbb{R}^n$ from the selected DCT coefficients. The verification score is

$$S(x; m) = \max_{r' \in \mathcal{C}(x)} s(m, r')^\top \bar{\ell}(x), \qquad \bar{\ell}(x) = \frac{\ell(x)}{\|\ell(x)\|_2}, \tag{6}$$

and an image with $\ell(x) = 0$ is rejected. Since every $R_{\kappa,r'}$ is orthogonal, the normalization is the same for all candidates and uses no key, so it does not change which candidate attains the maximum. It makes S invariant to the scale of $\ell(x)$. Without it, the spread of the correlation between $\ell(x)$ and the pattern of a wrong candidate grows in proportion to $\|\ell(x)\|_2$, so an image with large energy in the selected DCT components could exceed a threshold set on unwatermarked images through a candidate whose pattern it does not carry. The watermark is verified when $S(x; m) \geq \tau$, where τ is selected using unwatermarked images. For message recovery, the receiver decodes over the candidate content codes and returns a valid message. Details of candidate generation, score computation, and message decoding are provided in App. A.5.

4.4 Theoretical Analysis

We study forgery success and indistinguishability without the secret key under the threat model in Sec. 3.2. All proofs and supporting results are in App. C.

The adversary observes N genuine watermarked images, whose selected content codes form $\mathcal{R}_{\text{obs}}$, but cannot query the private verifier. We use the verification candidate limit $|\mathcal{C}(x)| \leq L$ from App. A.5 and two properties: **(S1)** for fixed x, m, r, $\mathcal{C}(x)$, $\ell(x)$, and $q(m, r)$ use no keyed mappings and are independent of κ; Θ_Φ and τ are fixed independently of the mapping family. **(S2)** generation selects r before watermark insertion, uses only $R_{\kappa,r}$, and then depends on κ and m only through $s(m, r)$.

Assumption 1 (Pseudorandom content-dependent mappings (Luby & Rackoff, 1988; Goldreich et al., 1986))**.** *For every query $Q \geq 0$, consider any allowed computational procedure that queries $r \mapsto R_{\kappa,r}$ on Q distinct content codes. Let W be any event determined by the procedure. Under the idealized model, each distinct content code r is assigned an independent uniformly random signed-permutation matrix R_r, and the same matrix is reused whenever r appears again. Then $|\Pr[W] - \Pr_{\text{id}}[W]| \leq \epsilon_{\text{key}}(Q)$, where $\Pr_{\text{id}}$ denotes probability in the idealized model.*

Assumption 1 allows the keyed signed permutations used for at most Q distinct content codes to be replaced by independent uniform signed permutations, with probability gap at most $\epsilon_{\text{key}}(Q)$.

Theorem 1 (Detection-level forgery success bound)**.** *Under Assumption 1 and (S1)-(S2), every allowed adversary and every fixed threshold $\tau > 0$ satisfy*

$$\text{ASR}_{\text{det}}(\mathcal{A}) \leq \underbrace{\Pr_{\text{id}}\left[\mathcal{C}(x^f) \cap \mathcal{R}_{\text{obs}} \neq \varnothing\right]}_{(A)} + \underbrace{\min\left\{1, L \exp\left(-\frac{\tau^2}{2}\right)\right\}}_{(B)} + \underbrace{\epsilon_{\text{key}}(N + L)}_{(C)}, \tag{7}$$

where term (B) depends only on L and τ.

Term (A) is the probability that the forged image contains a candidate code from $\mathcal{R}_{\text{obs}}$. We refer to candidate codes not used in any adversary-observed watermarked generation as *unused content codes*. Term (B) bounds the probability that an unused code passes the verification threshold, while term (C) accounts for replacing the keyed mappings with random signed permutations over at most $N + L$ codes. Since Eq. 6 uses the unit vector $\bar{\ell}(x^f)$, term (B) is independent of the forged image; thus, success beyond (B) + (C) requires candidate-code reuse, as captured by term (A).

Theorem 2 (Content-code separation under bounded transformations)**.** *Fix a target image x^{tgt} and let $\mathcal{B}_{\varepsilon,\gamma}(x^{\text{tgt}})$ contain all $x \in \mathcal{X}$ satisfying the two constraints in Eq. 1. Suppose there is a set of bit positions $\mathcal{J} \subseteq \{1, \ldots, k_r\}$ for which $\widehat{r}_{\mathcal{J}}(x) = \widehat{r}_{\mathcal{J}}(x^{\text{tgt}})$ for every $x \in \mathcal{B}_{\varepsilon,\gamma}(x^{\text{tgt}})$. Also suppose that every $r' \in \mathcal{C}(x)$ satisfies $d_H(r', \widehat{r}(x)) \leq t_c$. If $\min_{r \in \mathcal{R}_{\text{obs}}} d_H\left(r_{\mathcal{J}}, \widehat{r}_{\mathcal{J}}(x^{\text{tgt}})\right) > t_c$, where the minimum is $+\infty$ when $\mathcal{R}_{\text{obs}} = \varnothing$, then $\mathcal{C}(x) \cap \mathcal{R}_{\text{obs}} = \varnothing$ for every $x \in \mathcal{B}_{\varepsilon,\gamma}(x^{\text{tgt}})$.*

The set $\mathcal{J}$ contains content-code bits that remain unchanged under all allowed target-image transformations. If every observed code differs on more than t_c bits in $\mathcal{J}$, none can enter the candidate list. App. C.2 gives sufficient conditions for identifying these stable bits.

Theorem 3 (Indistinguishability without the secret key)**.** *Let* Gen_κ *be the* TRAJBIND *generator and* $\mathsf{Gen}^{\mathrm{rand}}$ *be the same pipeline with* $s(m, r)$ *replaced by a fresh uniform pattern in* $\{-1, +1\}^n$ *for each generation. Under Assumption 1 and (S1)–(S2), any computational method* $\mathcal{D}$ *without access to* κ *or the verifier, using* N *generated images, satisfies*

$$\left|\Pr\left[\mathcal{D}(\mathsf{Gen}_\kappa) = 1\right] - \Pr\left[\mathcal{D}(\mathsf{Gen}^{\mathrm{rand}}) = 1\right]\right| \le \epsilon_{\mathrm{key}}(N) + \Pr_{\mathrm{rand}}\left[\mathrm{Rep}_N\right] \le \epsilon_{\mathrm{key}}(N) + \binom{N}{2} 2^{-H_\infty}, \tag{8}$$

where Rep_N *is the event that two generations select the same content code, and* H_∞ *is a lower bound on the conditional min-entropy of each selected code given prior key-independent information.*

If Gen^0 *is the unwatermarked generator, then*

$$\left|\Pr\left[\mathcal{D}(\mathsf{Gen}_\kappa) = 1\right] - \Pr\left[\mathcal{D}(\mathsf{Gen}^0) = 1\right]\right| \le \epsilon_{\mathrm{key}}(N) + \binom{N}{2} 2^{-H_\infty} + \Delta_N, \tag{9}$$

where Δ_N *is the maximum difference in output probabilities between* $\mathsf{Gen}^{\mathrm{rand}}$ *and* Gen^0 *over* N *generations and is independent of* κ *and* m*.*

This differs from the distribution-preserving construction used in pseudorandom-code methods (Christ & Gunn, 2024; Gunn et al., 2025a). For TRAJBIND, Δ_N captures the distribution difference between watermark embedding with random patterns and unwatermarked generation.

Table 1: **Watermark forgery attack success rates on MS-COCO.** $\mathrm{ASR}_{\mathrm{det}}$ reports detection-level forgery success, $\mathrm{ASR}_{\mathrm{msg}}$ message-level success. Lower is better. For $\mathrm{ASR}_{\mathrm{det}}$, the two values correspond to using real images and unwatermarked images as the reference sets, respectively. "–" indicates that $\mathrm{ASR}_{\mathrm{msg}}$ is not applicable to zero-bit watermarking methods.

Method	Averaging		Latent-Matching		Imprinting		Reprompting		WMCopier	
	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$
TreeRing	1.000/1.000	–	0.873/0.647	–	1.000/1.000	–	1.000/1.000	–	0.279/0.076	–
RingID	1.000/1.000	–	0.990/0.990	–	1.000/1.000	–	1.000/1.000	–	0.183/0.180	–
ShallowDiffuse	0.990/0.975	–	0.997/0.993	–	1.000/1.000	–	1.000/1.000	–	0.012/0.000	–
SERUM	1.000/1.000	–	0.727/0.490	–	0.150/0.100	–	1.000/1.000	–	0.005/0.000	–
SemBind-TR	0.300/0.283	–	0.033/0.027	–	0.050/0.020	–	0.000/0.000	–	0.020/0.000	–
StegaStamp	0.865/0.885	0.000	0.007/0.007	0.000	0.620/0.670	0.000	0.900/0.920	0.000	0.934/0.937	0.000
Gaussian-Shading	1.000/1.000	0.667	1.000/1.000	0.330	1.000/1.000	0.990	1.000/1.000	1.000	0.964/0.976	0.008
StableSignature	0.005/0.005	0.000	0.000/0.000	0.000	0.000/0.000	0.000	0.010/0.010	0.000	1.000/1.000	0.403
DiffMark	0.995/0.995	0.155	0.977/0.990	0.000	1.000/1.000	0.060	1.000/1.000	0.130	0.001/0.004	0.000
Spherical	1.000/1.000	0.000	0.993/0.993	0.080	1.000/1.000	0.730	1.000/1.000	1.000	0.000/0.000	0.000
PRC	0.000/0.000	0.000	0.020/0.057	0.000	1.000/1.000	1.000	1.000/1.000	1.000	0.000/0.000	0.000
SemBind-GS	0.633/0.650	0.000	0.163/0.170	0.000	0.330/0.340	0.000	0.110/0.110	0.000	0.000/0.000	0.000
TRAJBIND-MIN	0.000/0.017	0.000	0.000/0.010	0.000	0.000/0.020	0.000	0.100/0.100	0.030	0.000/0.000	0.000
TRAJBIND-OPT	0.000/0.000	0.000	0.000/0.000	0.000	0.000/0.000	0.000	0.031/0.031	0.031	0.000/0.000	0.000

5 EXPERIMENTS

Baselines. We compare TRAJBIND with 12 watermark methods: TreeRing (Wen et al., 2023), RingID (Ci et al., 2024), StegaStamp (Tancik et al., 2020), StableSignature (Fernandez et al., 2023), Gaussian-Shading (Yang et al., 2024b), DiffMark (Nguyen-Le et al., 2026), PRC (Gunn et al., 2025b), ShallowDiffuse (Li et al., 2025), SERUM (Kociszewski et al., 2026), Spherical (Hu et al., 2026), SemBind-TR and SemBind-GS (Zhang et al., 2026).

Datasets, Base models and Metrics. We evaluate on three primary datasets: Diffusion-DB (Wang et al., 2023), MS-COCO (Lin et al., 2014), and DALL-E3 (Betker et al., 2023). For watermark robustness, we report TPR@0.1%FPR under benign transformations. For forgery resistance, we report detection-level attack success rate ($\mathrm{ASR}_{\mathrm{det}}$) and exact-message attack success rate ($\mathrm{ASR}_{\mathrm{msg}}$), as defined in Eq. 3. We use Stable Diffusion 2.1 (SD 2.1) as the base model for all experiments.

5.1 Robustness to forgery attacks

We evaluate against 5 forgery attacks using $1,000$ prompts from MS-COCO, DiffusionDB, DALL-E3, including *Averaging* (Yang et al., 2024a), *Latent-Matching* (Jain et al., 2025), *Imprinting* and *Reprompting* (Müller et al., 2025), and *WMCopier* (Dong et al., 2026). Detailed formulations of these attacks are provided in App. D. **Results.** Tab. 1 shows that TrajBind has the lowest worst-case forgery success, measured by the largest $\mathrm{ASR}_{\mathrm{det}}$ or $\mathrm{ASR}_{\mathrm{msg}}$ over the five attacks, corresponding to $\mathcal{R}_{\mathrm{forge}}$ in Sec. 3.3. Most zero-bit baselines reach high $\mathrm{ASR}_{\mathrm{det}}$ under several attacks, while for multi-bit methods, high detection-level success does not always lead to exact-message recovery. SemBind improves over its Tree-Ring and Gaussian-Shading bases but remains vulnerable to Averaging. In contrast, TrajBind-OPT obtains zero $\mathrm{ASR}_{\mathrm{det}}$ and $\mathrm{ASR}_{\mathrm{msg}}$ under four attacks and only 0.031 under Reprompting. Results on DiffusionDB and DALL-E3 show the same trend, while App. F.4 shows that all exact-message forgeries and most detection-level acceptances occur with a content-code collision, consistent with Theorem 1.

5.2 Robustness to Removal and Distortion Attacks

Table 2: **Watermark robustness on MS-COCO under removal attacks.** Each entry reports TPR@0.1%FPR(↑) using real images / unwatermarked images as the respective reference sets.

Method	Clean	VAE1	Diff-40	Rinse-20	Rinse-60	Rinse-100	Average
TreeRing	1.000/1.000	0.994/0.984	1.000/0.998	1.000/0.996	0.986/0.954	0.953/0.879	0.989/0.969
RingID	1.000/1.000	0.999/0.999	1.000/1.000	1.000/1.000	0.999/0.999	0.995/0.995	0.999/0.999
ShallowDiffuse	1.000/1.000	0.954/0.942	1.000/1.000	1.000/1.000	0.998/0.993	0.965/0.951	0.986/0.981
SERUM	1.000/1.000	0.999/0.993	1.000/1.000	1.000/1.000	1.000/0.999	1.000/0.999	1.000/0.999
SemBind-TR	0.994/0.992	0.763/0.713	0.977/0.967	0.970/0.967	0.930/0.907	0.857/0.810	0.915/0.893
StegaStamp	1.000/1.000	1.000/1.000	0.994/0.996	0.994/0.995	0.884/0.906	0.523/0.590	0.899/0.915
Gaussian-Shading	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000
StableSignature	1.000/1.000	0.001/0.001	0.005/0.005	0.002/0.002	0.000/0.000	0.001/0.001	0.168/0.168
DiffMark	1.000/1.000	0.954/0.967	1.000/1.000	1.000/1.000	0.997/0.998	0.986/0.992	0.990/0.993
Spherical	1.000/1.000	0.946/0.951	1.000/1.000	1.000/1.000	0.998/0.998	0.984/0.984	0.988/0.989
PRC	1.000/1.000	0.560/0.610	0.987/0.987	0.940/0.950	0.847/0.867	0.520/0.597	0.809/0.835
SemBind-GS	0.998/0.998	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000
TrajBind-Min	0.989/0.986	0.987/0.987	0.968/0.968	0.970/0.966	0.942/0.936	0.925/0.921	0.964/0.961
TrajBind-OPT	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	0.998/0.998	0.992/0.992	0.998/0.998

We follow WAVES benchmark (An et al., 2024) for distortion and removal attacks, using the same settings. We test 5 distortions: Gaussian blur, brightness, contrast, JPEG compression, and Gaussian noise. For removal, we consider VAE reconstruction, diffusion regeneration, and repeated diffusion regeneration (Rinse), using VAE quality 1, Diff-40, and $2\times$ Rinse with 20, 60, and 100 steps. Full details are provided in App. E.1.

Reference sets. On MS-COCO, we set the detection threshold using both real images and unwatermarked companion images from the same generation pipeline: the former measures false positives on natural images, while the latter reduces the effect of differences between real and generated images. Since DiffusionDB and DALL-E3 do not provide corresponding real images, we use only unwatermarked references for these datasets. MS-COCO distortion results are provided in App. F.6.

Results. Tab. 2 shows that TrajBind-Opt remains highly robust across all removal attacks, while TrajBind-Min shows a larger drop under Rinse-100 attack. This stronger robustness comes from TrajBind-Opt's image-specific optimization over the full latent space, which preserves watermark recovery across the transformations considered during optimization. Moreover, TrajBind-Opt achieves removal robustness comparable to strong baselines such as Gaussian-Shading and SemBind-GS, while retaining the strong forgery resistance reported in Sec. 5.1. These results show that introducing content binding does not sacrifice robustness to watermark removal.

5.3 Image Quality

We evaluate image quality using 500 real images from MS-COCO and 500 unwatermarked images generated by SD 2.1 using MS-COCO prompts. Tab. 3 shows that TrajBind-OPT substantially improves image quality over TrajBind-MIN, e.g., CLIP-FID from 23.93 to 19.82. This agrees

Table 3: **Image quality on MS-COCO.** Each entry reports mean ± standard deviation. FID and CLIP-FID are computed using real and unwatermarked images as the respective reference sets.

Method	Bits	MS-COCO (real images)				MS-COCO (unwatermarked images)			
		SSIM↑	LPIPS↓	FID↓	CLIP-FID↓	SSIM↑	LPIPS↓	FID↓	CLIP-FID↓
TreeRing	0	0.4667 ± 0.0055	0.4798 ± 0.0024	146.26 ± 0.16	31.04 ± 0.81	0.4667 ± 0.0055	0.4798 ± 0.0024	119.38 ± 3.98	17.55 ± 0.49
RingID	0	0.3975 ± 0.0052	0.5336 ± 0.0027	149.29 ± 1.35	31.54 ± 0.64	0.3975 ± 0.0052	0.5336 ± 0.0027	121.98 ± 3.85	18.32 ± 0.44
ShallowDiffuse	0	0.9162 ± 0.0024	0.0273 ± 0.0007	145.55 ± 1.35	31.11 ± 0.81	0.9162 ± 0.0024	0.0273 ± 0.0007	117.94 ± 4.51	17.44 ± 0.65
SERUM	0	0.3110 ± 0.0038	0.6045 ± 0.0015	147.29 ± 0.53	31.25 ± 0.73	0.3110 ± 0.0038	0.6045 ± 0.0015	119.73 ± 3.25	17.80 ± 0.43
SemBind-TR	0	0.6084 ± 0.0025	0.3060 ± 0.0026	145.25 ± 0.71	32.24 ± 0.89	0.6084 ± 0.0025	0.3060 ± 0.0026	118.14 ± 3.71	17.31 ± 0.31
StegaStamp	100	0.8962 ± 0.0011	0.0993 ± 0.0009	184.44 ± 0.92	36.32 ± 0.98	0.8962 ± 0.0011	0.0993 ± 0.0009	181.24 ± 4.26	43.03 ± 0.65
Gaussian-Shading	256	0.1814 ± 0.0046	0.7234 ± 0.0015	144.08 ± 1.70	31.20 ± 0.78	0.1814 ± 0.0046	0.7234 ± 0.0015	119.30 ± 3.53	18.22 ± 0.47
StableSignature	48	0.8743 ± 0.0007	0.0387 ± 0.0007	147.10 ± 0.55	31.15 ± 0.76	0.8743 ± 0.0007	0.0387 ± 0.0007	119.12 ± 3.83	17.94 ± 0.47
DiffMark	64	0.8616 ± 0.0018	0.0699 ± 0.0012	148.95 ± 0.84	29.93 ± 0.68	0.8616 ± 0.0018	0.0699 ± 0.0012	123.49 ± 4.15	18.52 ± 0.59
Spherical	256	0.1679 ± 0.0035	0.7177 ± 0.0023	145.48 ± 0.59	31.19 ± 0.72	0.1679 ± 0.0035	0.7177 ± 0.0023	118.08 ± 3.74	17.56 ± 0.39
PRC	512	0.1886 ± 0.0071	0.7367 ± 0.0021	143.78 ± 1.36	32.25 ± 0.70	0.1886 ± 0.0071	0.7367 ± 0.0021	116.77 ± 3.75	17.27 ± 0.55
SemBind-GS	256	0.1790 ± 0.0063	0.7327 ± 0.0018	147.03 ± 1.15	32.02 ± 0.87	0.1790 ± 0.0063	0.7327 ± 0.0018	119.62 ± 3.35	17.47 ± 0.44
TRAJBIND-MIN	512	0.5973 ± 0.0012	0.2707 ± 0.0004	154.84 ± 0.54	31.03 ± 0.73	0.5973 ± 0.0012	0.2707 ± 0.0004	137.37 ± 2.91	23.93 ± 0.48
TRAJBIND-OPT	512	0.7672 ± 0.0011	0.0914 ± 0.0012	149.45 ± 0.92	30.78 ± 0.64	0.8067 ± 0.0011	0.0911 ± 0.0012	121.92 ± 3.06	19.82 ± 0.29

with the design of TRAJBIND-OPT, which optimizes an image-specific perturbation to reduce visual distortion while preserving watermark recovery. Despite embedding 512 bits and remaining robust to forgery, distortion, and removal attacks, TRAJBIND-OPT achieves image quality comparable to existing methods, including similar FID and CLIP-FID on real MS-COCO images. Better FID and CLIP-FID against real images indicate that TRAJBIND outputs remain close to the real-image distribution, while the relatively higher values against unwatermarked images reflect the distribution change introduced by watermark embedding. This is because TRAJBIND embeds the content-dependent watermark during the later denoising steps, after the content code has been obtained.

5.4 ANALYSIS OF DCT-BASED CONTENT BINDING

Fig. 2 analyzes the DCT design used for content binding. As shown in Fig. 2(a), we use low-frequency DCT components for the content code and separate components for watermark embedding, preventing the watermark update from directly changing the content features. Fig. 2(b) studies raw watermark-bit recovery without content correction or ECC. Recovery is poor in the low-frequency content region but becomes much more reliable in the watermark region, although about 3–5% errors remain in bins 5–15. These results support frequency separation, while also motivating content correction and ECC for the remaining errors in our method.

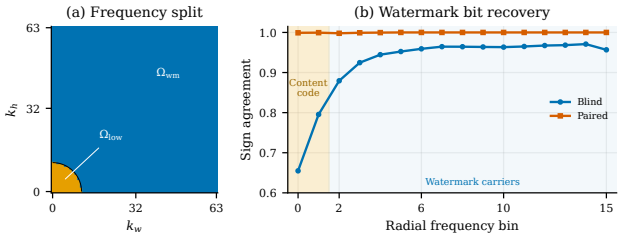


Figure 2: **DCT-based content binding.** (a) Low-frequency DCT components are used for the content code, while separate components carry the watermark. (b) Raw watermark-bit sign agreement across frequency bands at steps 45–49, without content correction or ECC. "Paired" subtracts the corresponding unwatermarked contribution.

6 CONCLUSION

We introduced TRAJBIND, an in-trajectory content-bound watermarking method for resisting forgery attacks on diffusion watermarks. TRAJBIND extracts a compact content code from the predicted clean image at an intermediate denoising step and combines it with the secret key to determine the watermark pattern embedded during the remaining denoising steps, without an additional generation pass or external semantic model. We also provide a general formulation of watermark forgery and a theoretical analysis that bounds detection-level forgery success in terms of content-code collision. Experiments against five forgery attacks show that TRAJBIND strongly reduces both detection-level and exact-message forgery success, while maintaining robustness to common image transformations and removal attacks and preserving good image quality. Together, these results show that content-dependent watermark patterns can be constructed within a single diffusion trajectory and provide an effective defense against existing watermark forgery attacks.

AI USE STATEMENT

In this work, we use Generative AI tools to support language editing and manuscript refinement, and to assist with implementation, debugging, and code refactoring. These tools were not used to develop the theoretical model or conceptual framework, formulate mathematical claims, or to generate experimental results. No other required AI-use disclosure category was applicable to this work.

All AI-assisted content was reviewed and verified by the authors. AI-assisted code was manually inspected, executed, and tested before use in the reported experiments. The authors take full responsibility for the final manuscript, including its text, technical claims, proofs, code, and experimental results.

ETHICS STATEMENT

This work studies watermarking for identifying AI-generated images and resisting watermark forgery. The forgery attacks in the paper are included to test watermark security and are run only in controlled experiments. These attack methods could also be misused to weaken or copy watermarks; thus, we present them in the context of measuring and improving watermark security. TRAJBIND only checks whether the recovered watermark is consistent with the image content and a secret key; it does not determine whether an image is truthful, harmful, or legally authorized. In deployment, watermark decisions should therefore be used together with other provenance information, and secret keys should be protected. Our experiments use existing popular public image datasets and generated images for testing purposes only, and we do not collect new human-subject data. Using and storing experimental data complies with relevant data protection regulations.

REPRODUCIBILITY STATEMENT

To support reproducibility, Secs. 4–5 describe the threat model and metrics, the method, datasets, base model, baselines, and the main experimental setup. Appendix A provides detailed method components and complete algorithms for key generation, embedding, extraction, decoding, and verification. Appendix C states the assumptions and gives the proofs; Appendix D gives the mathematical formulations of the forgery attacks; Appendix E gives the settings for distortion and removal attacks; and Appendix F provides white-box tests, ablations, additional forgery, distortion, and removal results, together with qualitative examples. We use the datasets, models, metrics, and attack settings stated in the paper, including the WAVES settings for distortion and removal attacks. Together, these details specify the method and experimental setup used for the reported results. Our source code will be released after acceptance.

REFERENCES

N. Ahmed, T. Natarajan, and K.R. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, C-23(1):90–93, January 1974. ISSN 2326-3814. doi: 10.1109/t-c.1974.223784. URL http://dx.doi.org/10.1109/t-c.1974.223784.

Bang An, Mucong Ding, Tahseen Rabbani, Aakriti Agrawal, Yuancheng Xu, Chenghao Deng, Sicheng Zhu, Abdirisak Mohamed, Yuxin Wen, Tom Goldstein, et al. Waves: Benchmarking the robustness of image watermarks. In *International Conference on Machine Learning*, pp. 1456–1492. PMLR, 2024.

Kasra Arabi, R Teal Witter, Chinmay Hegde, and Niv Cohen. Seal: Semantic aware image watermarking. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 16196–16205. IEEE, 2025.

Erdal Arikan. Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on information Theory*, 55(7): 3051–3073, 2009.

Severine Baudry, Jean-François Delaigle, Bülent Sankur, Benoıt Macq, and Henri Maıtre. Analyses of error correction strategies for typical communication channels in watermarking. *Signal Processing*, 81(6):1239–1250, 2001.

Mihir Bellare. *New Proofs for NMAC and HMAC: Security Without Collision-Resistance*, pp. 602–619. Springer Berlin Heidelberg, 2006. ISBN 9783540374336. doi: 10.1007/11818175_36. URL http://dx.doi.org/10.1007/11818175_36.

Mihir Bellare and Phillip Rogaway. *The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs*, pp. 409–426. Springer Berlin Heidelberg, 2006. ISBN 9783540345473. doi: 10.1007/11761679_25. URL http://dx.doi.org/10.1007/11761679_25.

James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023.

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration Inequalities - A Nonasymptotic Theory of Independence*. Oxford University Press, 2013. ISBN 978-0-19-953525-5. URL https://doi.org/10.1093/acprof:oso/9780199535255.001.0001.

D. Chase. Class of algorithms for decoding block codes with channel measurement information. *IEEE Transactions on Information Theory*, 18(1):170–182, January 1972. ISSN 1557-9654. doi: 10.1109/tit.1972.1054746. URL http://dx.doi.org/10.1109/tit.1972.1054746.

Miranda Christ and Sam Gunn. *Pseudorandom Error-Correcting Codes*, pp. 325–347. Springer Nature Switzerland, 2024. ISBN 9783031683916. doi: 10.1007/978-3-031-68391-6_10. URL http://dx.doi.org/10.1007/978-3-031-68391-6_10.

Hai Ci, Pei Yang, Yiren Song, and Mike Zheng Shou. Ringid: Rethinking tree-ring watermarking for enhanced multi-key identification. In *European conference on computer vision*, pp. 338–354. Springer, 2024.

I.J. Cox, J. Kilian, F.T. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Image Processing*, 6(12):1673–1687, December 1997. ISSN 1941-0042. doi: 10.1109/83.650120. URL http://dx.doi.org/10.1109/83.650120.

Ziping Dong, Chao Shuai, Zhongjie Ba, Peng Cheng, Zhan Qin, Qinglong Wang, and Kui Ren. Wmcopier: forging invisible watermarks on arbitrary images. volume 38, pp. 138035–138065, 2026.

European Parliament and Council of the European Union. Regulation (EU) 2024/1689 of the european parliament and of the council of 13 june 2024 laying down harmonised rules on artificial intelligence (artificial intelligence act). Official Journal of the European Union, 2024. Article 50(2) and Recital 133.

Pierre Fernandez, Guillaume Couairon, Hervé Jégou, Matthijs Douze, and Teddy Furon. The stable signature: Rooting watermarks in latent diffusion models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22409–22420. IEEE, 2023.

M.P.C. Fossorier and Shu Lin. Soft-decision decoding of linear block codes based on ordered statistics. *IEEE Transactions on Information Theory*, 41(5):1379–1396, 1995. ISSN 0018-9448. doi: 10.1109/18.412683. URL http://dx.doi.org/10.1109/18.412683.

Zheng Gao, Yifan Yang, Xiaoyu Li, Xiaoyan Feng, Haoran Fan, Yang Song, and Jiaojiao Jiang. Slice: Semantic latent injection via compartmentalized embedding for image watermarking. *arXiv preprint arXiv:2603.12749*, 2026.

Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, August 1986. ISSN 1557-735X. doi: 10.1145/6490.6503. URL http://dx.doi.org/10.1145/6490.6503.

Samuel Gunn, Xuandong Zhao, and Dawn Song. An undetectable watermark for generative image models. In *International Conference on Learning Representations*, volume 2025, pp. 6612–6637, 2025a.

Samuel Gunn, Xuandong Zhao, and Dawn Song. An undetectable watermark for generative image models. In *International Conference on Learning Representations*, volume 2025, pp. 6612–6637, 2025b.

Matthias Hein and Maksym Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/e077e1a544eec4f0307cf5c3c721d944-Paper.pdf.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, March 1963. ISSN 1537-274X. doi: 10.1080/01621459.1963.10500830. URL http://dx.doi.org/10.1080/01621459.1963.10500830.

Xiaoxiao Hu, Jiaqi Jin, Sheng Li, Wanli Peng, Xinpeng Zhang, and Zhenxing Qian. Spherical watermark: Encryption-free, lossless watermarking for diffusion models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=2eAGrunxVz.

Anubhav Jain, Yuya Kobayashi, Naoki Murata, Yuhta Takida, Takashi Shibuya, Yuki Mitsufuji, Niv Cohen, Nasir Memon, and Julian Togelius. Forging and removing latent-noise diffusion watermarks using a single image. *arXiv preprint arXiv:2504.20111*, 2(7), 2025.

Jan Kociszewski, Hubert Jastrzebski, Tymoteusz Stepkowski, Filip Manijak, Krzysztof Rojek, Franziska Boenisch, and Adam Dziedzic. SERUM: Simple, efficient, robust, and unifying marking for diffusion-based image generation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=AiBUm6iKBf.

Jiewei Lai, Lan Zhang, Chen Tang, Pengcheng Sun, Zhaopeng Zhang, Yunhao Wang, and Hui Jin. CSGuard: Toward forgery-resistant watermarking in diffusion models via compressed sensing constraint. *arXiv preprint arXiv:2605.01479*, 2026.

Wenda Li, Huijie Zhang, and Qing Qu. Shallow diffuse: Robust and invisible watermarking through low-dim subspaces in diffusion models. In *Advances in Neural Information Processing Systems 38*, NeurIPS 2025, pp. 170876–170906. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2025. doi: 10.52202/085713-5147. URL http://dx.doi.org/10.52202/085713-5147.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Qingyu Liu, Yitao Zhang, Zhongjie Ba, Chao Shuai, Peng Cheng, Tianhang Zheng, and Zhibo Wang. Attack-resistant watermarking for AIGC image forensics via diffusion-based semantic deflection. In *The Fourteenth International Conference on Learning Representations*, 2026.

Michael Luby and Charles Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM Journal on Computing*, 17(2):373–386, April 1988. ISSN 1095-7111. doi: 10.1137/0217022. URL http://dx.doi.org/10.1137/0217022.

Andreas Müller, Denis Lukovnikov, Jonas Thietke, Asja Fischer, and Erwin Quiring. Black-box forgery attacks on semantic watermarks for diffusion models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20937–20946. IEEE, 2025.

Hong-Hanh Nguyen-Le, Van-Tuan Tran, Thuc Nguyen, and Nhien-An Le-Khac. A survey on proactive deepfake defense: Disruption and watermarking. *ACM Computing Surveys*, 58(5):1–37, 2025.

Hong-Hanh Nguyen-Le, Van-Tuan Tran, Thuc D Nguyen, and Nhien-An Le-Khac. Transferable multi-bit watermarking across frozen diffusion models via latent consistency bridges. *arXiv preprint arXiv:2603.20304*, 2026.

Kai Niu and Kai Chen. Crc-aided decoding of polar codes. *IEEE communications letters*, 16(10): 1668–1671, 2012.

Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. 2004.

Tomáš Souček, Sylvestre-Alvise Rebuffi, Pierre Fernandez, Nikola Jovanović, Hady Elsahar, Valeriu Lacatusu, Tuan Tran, and Alexandre Mourachko. Transferable black-box one-shot forging of watermarks via image preference models. volume 38, pp. 50251–50274, 2026.

Ido Tal and Alexander Vardy. List decoding of polar codes. *IEEE transactions on information theory*, 61(5):2213–2226, 2015.

Matthew Tancik, Ben Mildenhall, and Ren Ng. Stegastamp: Invisible hyperlinks in physical photographs. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2114–2123. IEEE, 2020.

Peter Trifonov. Efficient design and decoding of polar codes. *IEEE transactions on communications*, 60(11):3221–3227, 2012.

Zijie J Wang, Evan Montoya, David Munechika, Haoyang Yang, Benjamin Hoover, and Duen Horng Chau. Diffusiondb: A large-scale prompt gallery dataset for text-to-image generative models. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 893–911, 2023.

Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-rings watermarks: Invisible fingerprints for diffusion images. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 58047–58063. Curran Associates, Inc., 2023. doi: 10.52202/075280-2529. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/b54d1757c190ba20dbc4f9e4a2f54149-Paper-Conference.pdf.

JinFeng Xie, Peipeng Yu, Jianwei Fei, Xiaoyu Zhou, and Zhihua Xia. Safetr: Verifiable semantic tree-ring watermark for diffusion model against forgery attacks. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 9505–9511. IEEE, 2026.

Pei Yang, Hai Ci, Yiren Song, and Mike Zheng Shou. Can simple averaging defeat modern watermarks? volume 37, pp. 56644–56673, 2024a.

Zijin Yang, Kai Zeng, Kejiang Chen, Han Fang, Weiming Zhang, and Nenghai Yu. Gaussian shading: Provable performance-lossless image watermarking for diffusion models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12162–12171. IEEE, 2024b.

Xin Zhang, Zijin Yang, Kejiang Chen, Linfeng Ma, Weiming Zhang, and Nenghai Yu. Sembind: Binding diffusion watermarks to semantics against black-box forgery attacks. In *Forty-third International Conference on Machine Learning*, 2026.

Tong Zhou, Ruyi Ding, Gaowen Liu, Charles Fleming, Ramana Rao Kompella, Yunsi Fei, Xiaolin Xu, and Shaolei Ren. Metaseal: Defending against image attribution forgery through content-dependent cryptographic watermarks. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=8i3ErmCfdJ.

Yifan Zhu, Yihan Wang, and Xiao-Shan Gao. Towards robust content watermarking against removal and forgery attacks. In *Findings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

ROADMAP

The appendix is organized as follows. Sec. A provides additional details of TRAJBIND, including the DCT-based content code, content-dependent permutation, ECC, embedding variants, verification, and full algorithms. Sec. B discusses additional related work. Sec. C gives the assumptions and proofs for the theoretical results in the main paper, while Sec. D gives the formulations of the forgery attacks. Sec. E provides implementation and attack settings. Sec. F presents additional experiments, including white-box attacks, ablations, forgery, distortion and removal results, and qualitative examples. Finally, Sec. G discusses the limitations of our method.

A ADDITIONAL DETAILS OF THE METHOD

A.1 WHY LOW-FREQUENCY DCT COEFFICIENTS ARE USED

The content code should meet two practical requirements. First, it should capture image-dependent information that is already available from the intermediate prediction, enabling the watermark embedder to determine the code before generation is complete. Second, it should remain sufficiently stable under benign image transformations, allowing the verifier to recover a consistent code from the received image. Therefore, we construct the content code from low-frequency DCT coefficients of the VAE latent and aggregate them across multiple deterministic views.

Let $Z \in \mathbb{R}^{C_z \times H_z \times W_z}$ be a VAE latent. For latent channel c and spatial frequency (k_h, k_w), the orthonormal two-dimensional DCT-II coefficient (Ahmed et al., 1974) is:

$$F_{c,k_h,k_w}(Z) = \rho_{k_h}^{(H_z)} \rho_{k_w}^{(W_z)} \sum_{h=0}^{H_z-1} \sum_{w=0}^{W_z-1} Z_{c,h,w} \times \cos\left(\frac{\pi(2h+1)k_h}{2H_z}\right) \cos\left(\frac{\pi(2w+1)k_w}{2W_z}\right), \quad (10)$$

where $\rho_0^{(N)} = 1/\sqrt{N}$ and $\rho_k^{(N)} = \sqrt{2/N}$ for $k > 0$. Small k_h and k_w correspond to low spatial frequencies, which summarize gradual spatial variations in the latent feature maps. In contrast, larger frequencies respond more strongly to local changes such as boundaries, texture, and fine detail. We select a fixed set Ω_{low} near the low-frequency corner and define $\mathcal{D}_{\text{low}}(Z) = \big(F_{c,k_h,k_w}(Z)\big)_{c=1,\ldots,C_z;\,(k_h,k_w)\in\Omega_{\text{low}}}$, using the same coefficient ordering for every image.

Low-frequency coefficients are suitable for content coding for three reasons. **First**, they summarize spatial variations over relatively broad regions of the latent rather than relying on fine local detail. This makes them informative about the visual structure already present in the late predicted image. **Second**, common transformations such as compression, mild filtering, and regeneration tend to alter fine details more strongly than broad spatial variations, making low-frequency features more repeatable between the transmitter prediction and the received image. **Third**, the content-code features can be separated from the DCT region used for watermark embedding. If Ω_{wm} denotes the DCT indices used for watermark embedding through P, we choose $\Omega_{\text{low}} \cap \Omega_{\text{wm}} = \varnothing$. This reduces direct overlap between the features used to determine the content code and those used to carry the watermark.

The resulting coefficients are not intended to represent object identities or high-level semantic labels. Instead, they provide a compact description of the image-dependent spatial structure needed for content binding. The separation between Ω_{low} and Ω_{wm} is exact at the selected latent coordinates, although subsequent denoising and VAE processing can mix information across frequencies.

A.2 CONTENT-DEPENDENT PERMUTATION

The secret key κ and content code r determine a permutation $\pi_{\kappa,r}$ and sign sequence $\xi_{\kappa,r} \in \{-1,+1\}^n$. Their direction follows the convention in the main paper. For codeword position i ($i = 1, \ldots, n$),

$$s_{\pi_{\kappa,r}(i)}(m,r) = \xi_{\kappa,r,i}\, q_i(m,r). \quad (11)$$

Table 4: **Example of the content-dependent permutation.** The permutation maps each signed codeword value to a selected DCT component, while ξ_i may reverse its sign.

i	q_i	$\pi(i)$	ξ_i	Assigned value
1	$+1$	4	$+1$	$s_4 = +1$
2	-1	1	-1	$s_1 = +1$
3	$+1$	6	$+1$	$s_6 = +1$
4	$+1$	2	-1	$s_2 = -1$
5	-1	5	$+1$	$s_5 = -1$
6	-1	3	-1	$s_3 = +1$

Thus, $q_i(m, r)$ is assigned to DCT component $\pi_{\kappa,r}(i)$ after multiplication by the corresponding sign.

Tab. 4 gives a six-element example. For readability, the κ and r subscripts are omitted. For this example, $q = (+1, -1, +1, +1, -1, -1)$ and $s = (+1, -1, +1, +1, -1, +1)$, where s is written in DCT-component order. Given a candidate content code r', the verifier reverses the mapping as $\widehat{q}_i = \xi_{\kappa,r',i}\, \widehat{s}_{\pi_{\kappa,r'}(i)}$. Because the permutation is one-to-one and $\xi^2_{\kappa,r',i} = 1$, this operation is information preserving. Its role is to make the physical watermark pattern depend jointly on the secret key and the content code. It does not provide error correction by itself.

A.3 The role of Error-Correction Code

The error-correcting code (ECC) in TrajBind is used to improve ***message recovery*** when the extracted watermark information contains errors. Specifically, we use a polar code (Arikan, 2009; Tal & Vardy, 2015; Niu & Chen, 2012) to introduce redundancy into the message before watermark embedding. Such redundancy is commonly used in watermarking to improve message recovery under distortions that perturb the extracted watermark information (Baudry et al., 2001; Hu et al., 2026). Given a message m and content code r, the encoder produces $b(m, r) = \mathrm{Enc}(m, r)$, which is converted to the signed codeword $q(m, r) = 2b(m, r) - \mathbf{1}_n$. During decoding, the aligned watermark information provides soft values for recovering the encoded message.

The ECC and the content-dependent signed permutation have different roles. The ECC provides redundancy for robust message recovery, whereas $R_{\kappa,r}$ determines how the encoded message is mapped to the watermark pattern. In particular, the signed permutation only rearranges and changes the signs of the codeword components. This means that it does not provide error correction. At the receiver, reversing this mapping restores the codeword ordering before ECC decoding.

ECC decoding is required when the embedded message is unknown and must be recovered. In contrast, for verification of a given message m, TrajBind directly constructs the corresponding encoded message for each candidate content code and measures its consistency with the extracted watermark information. Therefore, the ECC mainly improves the reliability of message recovery and should not be regarded as the mechanism that provides content binding. Content binding is instead introduced by the dependence of the watermark pattern on the content code and the secret key.

A.4 Imperceptibility-Aware Embedding Variants

We propose two embedding variants that share the same content code r, encoded message $b(m, r)$, signed codeword $q(m, r)$, content-dependent watermark pattern $s(m, r)$, and content-interference correction. They differ in how the watermark perturbation is determined during the final denoising steps.

TrajBind-Min: fixed-strength embedding. TrajBind-Min directly embeds the content-dependent watermark pattern with a fixed strength. Given $s(m, r)$, the latent watermark perturbation is $\Delta z_{\mathrm{wm}} = P^\top (A_s s(m, r))$,where A_s determines the embedding strength and $P^\top$ maps the selected DCT coefficients back to the latent space. This perturbation is constructed in the same way for all images and is not adapted to their individual content, making the method simple and single-pass but not explicitly optimized for imperceptibility.

TrajBind-Opt: imperceptibility-aware optimization. Because the remaining denoising process and VAE decoder transform latent perturbations nonlinearly, reducing A_s alone does not provide the best perceptual quality for a given level of watermark reliability. Therefore, TrajBind-Opt starts from the fixed-strength embedding of TrajBind-Min and optimizes an image-specific latent perturbation u. The content code r, watermark pattern $s(m, r)$, and diffusion-model parameters remain fixed throughout the optimization.

Let $x(u)$ denote the final generated image under perturbation u, and let x^0 denote the corresponding image generated without watermark embedding. We measure perceptual distortion as

$$\mathcal{D}(u) = \mathrm{LPIPS}\left(x(u), x^0\right) + \lambda_{\mathrm{pix}} \frac{\|x(u) - x^0\|_2^2}{HWC}, \tag{12}$$

where the first term measures perceptual difference and the second penalizes pixel-level deviation.

To preserve watermark recovery, we define a soft reliability loss using the receiver outputs under the content code r:

$$\mathcal{L}_{\mathrm{wm}}(v; m, r) = \frac{1}{n} \sum_{i=1}^{n} \frac{\log\left(1 + \exp\left[-q_i(m,r)\, \ell_{r,i}(v)\right]\right)}{\log 2}, \tag{13}$$

where $\ell_{r,i}(v)$ is the receiver value for codeword position i. A smaller value indicates stronger agreement between the received evidence and the intended signed codeword. Since $\ell_r(v) = R_{\kappa,r}^\top \ell(v)$ and $s(m,r) = R_{\kappa,r} q(m,r)$, we have $q(m,r)^\top \ell_r(v) = s(m,r)^\top \ell(v)$. Hence,

$$\mathcal{L}_{\mathrm{wm}}(v; m, r) = \frac{1}{n \log 2} \sum_{j=1}^{n} \log\left(1 + \exp[-s_j(m,r)\ell_j(v)]\right). \tag{14}$$

Therefore, with fixed optimization settings, the watermark loss depends on κ and m only through $s(m,r)$, as required by (S2).

Let u_0 denote the fixed perturbation from TRAJBIND-MIN, and let $\mathcal{G}$ denote a set of image transformations considered during optimization. Both the loss and any recovery checks during this optimization use the same fixed generation-time code r; they do not run the receiver's candidate-code search. We optimize

$$u^* = \arg\min_{u \in \mathcal{U}} \left[\mathcal{D}(u) + \lambda_{\mathrm{wm}} \sum_{T \in \mathcal{G}} \left[\frac{\mathcal{L}_{\mathrm{wm}}(T(x(u)); m, r)}{\mathcal{L}_{\mathrm{wm}}(T(x(u_0)); m, r)} - 1 \right]_+^2 \right], \tag{15}$$

where $[a]_+ = \max(a, 0)$ and $\mathcal{U}$ constrains the allowable latent perturbation. The second term penalizes solutions whose watermark-recovery loss becomes worse than that of TRAJBIND-MIN. Thus, TRAJBIND-OPT searches for an image-specific perturbation that reduces perceptual distortion while retaining the watermark evidence provided by the fixed-strength embedding.

The optimization procedure, transformation set $\mathcal{G}$, perturbation constraint $\mathcal{U}$, and associated hyperparameters are described in the Implementation Details section.

A.5 WATERMARK EXTRACTION AND VERIFICATION

Candidate content codes. Let $\mathcal{C}_L(x)$ denote the ordered list of at most L distinct candidate content codes constructed from the received image and the fixed content-code parameters, without using keyed watermark scores or decoding outcomes. Verification uses $\mathcal{C}(x) = \mathcal{C}_{L_{\mathrm{ver}}}(x)$; throughout the verification analysis, $L = L_{\mathrm{ver}}$. The message decoder may use a different limit L_{dec}.

For the candidate-list construction analyzed in Appendix C.2, candidates are indexed by sets of flipped bits $F \subseteq \{1, \ldots, k_r\}$ as $r_F = \widehat{r}(x) \oplus \mathbf{1}_F$. They are ordered by $C_x(F) = \sum_{i \in F} w_i(x)$, with key-independent weights $w_i(x) \geq 0$, and ties place a strict subset before its superset. For $1 \leq L \leq 2^{k_r}$, retaining the first L distinct candidates implies $d_H(r', \widehat{r}(x)) \leq \lfloor \log_2 L \rfloor$ for every $r' \in \mathcal{C}_L(x)$. This is the condition supporting the explicit separation certificate; candidate-list size alone does not imply it.

Verification score. The verifier obtains the selected DCT coefficients as $y(x) = PE(2x - 1)$ and applies a fixed key-independent score transformation to obtain $\ell(x) \in \mathbb{R}^n$, in the selected DCT-component order. For each $r' \in \mathcal{C}(x)$, it forms $\ell_{r'}(x) = R_{\kappa,r'}^\top \ell(x)$, which places the soft watermark values in codeword order. For $\ell(x) \neq 0$, the score is:

$$S(x; m) = \max_{r' \in \mathcal{C}(x)} \frac{q(m,r')^\top \ell_{r'}(x)}{\|\ell(x)\|_2} = \max_{r' \in \mathcal{C}(x)} s(m,r')^\top \bar{\ell}(x), \qquad \bar{\ell}(x) = \frac{\ell(x)}{\|\ell(x)\|_2}, \tag{16}$$

and $S(x; m) = -\infty$ when $\ell(x) = 0$, so such an image is rejected. Then $\mathrm{Ver}_\theta(x, m; \tau) = \mathbf{1}\{S(x; m) \geq \tau\}$. Because $R_{\kappa,r'}^\top R_{\kappa,r'} = I_n$, we have $\|\ell_{r'}(x)\|_2 = \|\ell(x)\|_2$ for every candidate. The

normalization is therefore computed once, uses no key, and does not change which candidate attains the maximum. Since $\ell(x)$ is key-independent, so is $\bar{\ell}(x)$, and property (S1) continues to hold.

In the experiments, the threshold is selected on unwatermarked images using the same verification procedure. The fixed-threshold theoretical results assume that τ is independent of the keyed mapping family. When the deployment key is used to select the threshold, the codes and mappings used in that procedure require additional accounting in both the used-code set and the mapping budget.

Watermark message decoding. For unknown-message recovery, the receiver follows Algorithm 3. With the system parameters fixed, its Decode procedure corresponds to Ext_θ in Sec. 3.1. The helper $\mathrm{Extract}$ only returns candidate content codes and $\ell(x)$; it does not itself recover a message. The displayed decoder returns the first CRC-valid message. The exact-message bound in Corollary 1(ii) additionally requires verification of the returned message, which is not a condition of the displayed decoder.

A.6 Details of the TrajBind Algorithms

This section gives the algorithms of TrajBind. The system consists of 6 main procedures: key generation, content-dependent encoding, in-trajectory embedding, watermark extraction, message decoding, and verification. The diffusion backbone, VAE encoder–decoder (E, D), DCT basis P, content-code parameters Θ_Φ, polar code, correction predictor, embedding schedules, and detection threshold are fixed before deployment. The secret key κ is used only to construct the content-dependent signed permutation.

Algorithm 1: Key generation and content-dependent encoding. KeyGen samples the secret key used to construct the content-dependent signed permutation. Given a message m and content code r, Encode_κ first applies the content-conditioned error-correction encoding, where $\mathrm{Enc}(m, r)$ includes the CRC computed from m and r, and then constructs the keyed signed permutation associated with r.

Algorithm 1 TrajBind key generation and content-dependent encoding

function KeyGen
  Sample a secret key κ
  return κ
end function

function Encode(κ, m, r)
  $b \leftarrow \mathrm{Enc}(m, r) \in \{0,1\}^n$ ▷ CRC over (m, r) followed by polar encoding
  $q \leftarrow 2b - \mathbf{1}_n$
  Derive $\pi_{\kappa,r}$ and $\xi_{\kappa,r} \in \{-1, +1\}^n$ from κ and r
  Construct $R_{\kappa,r}$ such that $(R_{\kappa,r})_{\pi_{\kappa,r}(i),i} = \xi_{\kappa,r,i}$ for $i = 1, \ldots, n$
  $s \leftarrow R_{\kappa,r} q$
  return s
end function

Algorithm 2: In-trajectory embedding. Embed_κ first estimates the content-interference correction at step j_a. The correction is applied during subsequent denoising steps. At step j_r, the predicted clean image determines the content code r and hence the content-dependent watermark pattern. The watermark perturbation is applied only in the denoising steps after this commitment.

The content-dependent watermark perturbation is therefore constructed only after the content code has been determined. In the implemented schedule, the content code is committed at step 44, and watermark embedding begins at step 45. TrajBind-Min uses the prescribed fixed perturbation during the remaining steps. TrajBind-Opt shares the same trajectory through the content-code commitment, keeps r and $s(m, r)$ fixed, and replaces the fixed post-commitment control with the optimized image-specific perturbation.

Algorithm 2 TRAJBIND in-trajectory watermark embedding

Require: Watermark message m, secret key κ, initial noise z_T, generation condition
Ensure: Watermarked image x^w
Initialize the diffusion trajectory from z_T
for $j = 0, \ldots, J-1$ **do**
  if $j > j_a$ **then**
    Apply the prescribed correction slice to the current latent
  end if
  if $j > j_r$ **then**
    Apply the prescribed watermark slice to the current latent
  end if
  Run the frozen denoising step and obtain $\widehat{z}_0^{(j)}$ and the next latent
  if $j = j_a$ **then**
    for $i = 1, \ldots, n$ **do**
      $\widehat{\omega}_i \leftarrow \lambda_{i,1}(P\widehat{z}_0^{(j_a)})_i + \lambda_{i,2}(Pz_T)_i + \lambda_{i,0}$
    end for
    Constrain $\widehat{\omega}$ to the correction budget B_c to obtain $\widetilde{\omega}$
    $\Delta z_{\mathrm{corr}} \leftarrow -P^\top \widetilde{\omega}$
  end if
  if $j = j_r$ **then**
    $\widehat{x}_0^{(j_r)} \leftarrow D(\widehat{z}_0^{(j_r)})$
    $r \leftarrow \phi_c(\widehat{x}_0^{(j_r)}; \Theta_\Phi)$
    $s \leftarrow \textsc{Encode}(\kappa, m, r)$
    $\Delta z_{\mathrm{wm}} \leftarrow A_s P^\top s$
  end if
end for
Decode the final latent with D to obtain x^w
return x^w

Algorithm 3: Extraction and verification. Algorithm 3 summarizes the receiver-side procedures. $\mathrm{Extract}$ constructs an ordered list of candidate content codes and obtains the key-independent vector $\ell(x)$. The secret key is used only when aligning this vector under a particular candidate content code.

Here, $\mathrm{Dec}(\ell_{r'}(x), r')$ denotes SCL decoding with the candidate content code r' used as the CRC context; it returns $\perp$ if no CRC-valid path is found. Decoding examines candidate content codes in their ranked order and returns the first valid payload. Verification does not perform message decoding: for the specified message m, it maximizes the consistency score over the candidate list, divides the maximum by $\|\ell(x)\|_2$, and compares the resulting score with τ.

B ADDITIONAL RELATED WORK

B.1 CONTENT-BOUND WATERMARK METHODS

Content-bound watermarking methods connect the watermark with image content through semantic or feature-dependent signals. SEAL (Arabi et al., 2025) first generates a proxy image, extracts its semantic representation through captioning and semantic embedding, and then applies locality-sensitive hashing to derive content-dependent keys for diffusion initialization. SLICE (Gao et al., 2026) decomposes image semantics into multiple factors using a vision-language model and binds them to different latent regions for localized verification. SafeTR (Xie et al., 2026) incorporates semantic verification into Tree-Ring watermarking. MetaSeal (Zhou et al., 2026) is a post-hoc method: it signs a caption of the image with ECDSA, encodes the caption and signature as a QR code, and embeds the code with an invertible network. SemBind (Zhang et al., 2026) and ISTS (Zhu et al., 2026) explicitly target forgery attacks. SemBind generates an auxiliary clean image and uses a trained semantic masker to obtain a binary semantic code, which modulates the initial latent of a base watermark such as Tree-Ring or Gaussian Shading. ISTS first generates an unwatermarked

Algorithm 3 TRAJBIND extraction, decoding, and verification

function EXTRACT(x, L)
  Compute $\overline{a}(x)$ and $h(x)$ from the deterministic views
  $\widehat{r}(x) \leftarrow \mathbf{1}\{h(x) > 0\}$
  Construct the ordered candidate list $\mathcal{C}_L(x)$
  $y(x) \leftarrow PE(2x - 1)$
  Obtain the key-independent vector $\ell(x)$ from $y(x)$
  return $\mathcal{C}_L(x), \ell(x)$
end function

function DECODE($\kappa, x, L_{\mathrm{dec}}$)
  $(\mathcal{C}_{L_{\mathrm{dec}}}(x), \ell(x)) \leftarrow$ EXTRACT(x, L_{dec})
  for each $r' \in \mathcal{C}_{L_{\mathrm{dec}}}(x)$ in candidate order **do**
    Reconstruct $R_{\kappa,r'}$ from κ and r'
    $\ell_{r'}(x) \leftarrow R_{\kappa,r'}^{\top}\ell(x)$
    $\widehat{m} \leftarrow \mathrm{Dec}(\ell_{r'}(x), r')$
    if $\widehat{m} \neq \perp$ **then**
      return $\widehat{m}$
    end if
  end for
  return $\perp$
end function

function VERIFY($\kappa, x, m, \tau, L_{\mathrm{ver}}$)
  $(\mathcal{C}_{L_{\mathrm{ver}}}(x), \ell(x)) \leftarrow$ EXTRACT(x, L_{ver})
  if $\ell(x) = 0$ **then**
    return 0
  end if
  $S \leftarrow -\infty$
  for each $r' \in \mathcal{C}_{L_{\mathrm{ver}}}(x)$ **do**
    Reconstruct $R_{\kappa,r'}$ from κ and r'
    $\ell_{r'}(x) \leftarrow R_{\kappa,r'}^{\top}\ell(x)$
    $q(m, r') \leftarrow 2\,\mathrm{Enc}(m, r') - \mathbf{1}_n$
    $S \leftarrow \max\{S, q(m, r')^{\top}\ell_{r'}(x)\}$
  end for
  $S \leftarrow S/\|\ell(x)\|_2$
  return $\mathbf{1}\{S \geq \tau\}$
end function

image, extracts its CLIP features, and uses a clustering-based selector to choose the injection step and pattern position of a zero-bit ring watermark; detection uses DDIM inversion.

Two further methods act on the denoising trajectory itself. CSGuard (Lai et al., 2026) computes $y = A\widehat{z}_0^{(t)}$ from the predicted clean latent at an intermediate step with a secret matrix A, and projects every later clean-latent prediction onto $\{z : Az = y\}$. The watermark itself is an existing initial-latent watermark such as Gaussian Shading under a key that does not depend on content; verification recomputes y from the encoded received image and runs a DDIM inversion constrained to the same set. PAI (Liu et al., 2026) steers the denoising trajectory with a deflection conditioned on the user key, and detection uses DDIM inversion.

Difference from TRAJBIND. SemBind (Zhang et al., 2026) is the closest published method that defends against watermark forgery through content binding, and we evaluate both of its variants in all experiments. SemBind, SEAL, SLICE, and ISTS obtain the content information from a separate image and an external model, and verify through DDIM inversion. Instead of using auxiliary clean generation or an additional trained semantic network, TRAJBIND derives the content code from the intermediate clean-image prediction $\widehat{x}_0^{(j_r)}$ of the trajectory that produces the delivered image, using the diffusion model's own VAE, and determines the watermark before generation is completed.

Verification uses only VAE encodings of the received image and no UNet evaluation. CSGuard also reads an intermediate prediction of the same trajectory, but uses it as a constraint on later predictions, and its watermark bits do not depend on content. In TRAJBIND, the content code selects the keyed signed permutation $R_{\kappa,r}$ and is part of the polar encoding $b(m, r)$, so both detection and exact-message decoding depend on it. In addition, our theoretical analysis bounds detection-level forgery success in terms of content-code collision and gives conditions under which content-altering forgeries cannot reuse an observed content code.

B.2 UNDETECTABLE WATERMARKING METHODS

A separate line of work studies *undetectable* watermarks, whose objective is to prevent any probabilistic polynomial-time adversary without the secret key from distinguishing watermarked generations from ordinary generations. Pseudorandom error-correcting codes (PRCs) (Christ & Gunn, 2024) generate codewords that appear random to anyone without the secret key, while still allowing the intended decoder to recover the encoded information even when some bits are corrupted. Given a PRC codeword $c \in \{-1, +1\}^n$, the signs of the initial diffusion latent are determined by c, while their magnitudes follow those of Gaussian samples. Informally, the resulting latent can be written as $z_i = c_i|g_i|$, where $g_i \sim \mathcal{N}(0, 1)$. Replacing the PRC codeword with uniformly random signs recovers the standard Gaussian latent distribution. Since PRC codewords are computationally indistinguishable from random signs, the resulting watermarked generations are likewise computationally indistinguishable from ordinary generations, while error correction enables robust detection with controlled false-positive probability.

Difference in principle. PRC and TRAJBIND address different security properties. PRC is designed primarily to make the *presence* of the watermark computationally undetectable without the secret key, while retaining robust keyed detection. Its pseudorandom codeword determines the initial latent but is not derived from the semantic content of the particular image that will be generated. Consequently, undetectability alone does not establish that watermark evidence observed on one image cannot be transferred to different content. TRAJBIND instead targets this forgery problem directly: the watermark pattern is indexed jointly by the secret key and the content code r, so different content is intended to induce different keyed watermark patterns even for the same message.

Difference in theoretical guarantees. The corresponding theoretical analyses have different goals. PRC-style analysis shows that pseudorandom encoded signs induce a sampling process computationally indistinguishable from ordinary Gaussian sampling, yielding computational undetectability between watermarked and unwatermarked generations under the underlying cryptographic assumptions (Christ & Gunn, 2024; Gunn et al., 2025b). Its detection analysis further establishes a controlled false-positive probability and robustness provided by the error-correcting code.

In contrast, we use pseudorandomness to analyze ***forgery resistance*** rather than watermark undetectability. Specifically, the keyed construction makes the watermark mapping associated with an unseen content code computationally indistinguishable from an independently random mapping, even after observing watermarked images associated with other content codes. Based on this property, Theorem 1 bounds forgery success through two mechanisms: reuse of a previously observed content code and accidental correlation with the watermark pattern associated with an unseen code. Theorem 2 further gives sufficient conditions preventing a forged image from reusing an observed content code. Thus, while PRC uses pseudorandomness to hide the presence of a watermark, TRAJBIND uses it to limit the transferability of watermark evidence across different content.

Theorem 3 considers a complementary question: whether an observer without the secret key can distinguish TRAJBIND generations from ordinary generations. Unlike PRC, TRAJBIND does not assume that watermark embedding preserves the original generation distribution. Our analysis proceeds in two steps. First, we show that the keyed TRAJBIND generator is close to an otherwise identical generator that uses fresh random watermark patterns, up to the pseudorandom-mapping error and the probability of content-code reuse. Second, comparing this random-pattern generator with the unwatermarked generator introduces the additional term Δ_N, which captures the distribution shift caused by watermark embedding itself. Therefore, PRC-style undetectability and our result provide different guarantees: PRC uses a distribution-preserving latent construction to hide the presence of

the watermark, whereas our analysis separates the secrecy of the keyed content-dependent mapping from the distributional effect of watermark embedding.

C Proofs and Supporting Analysis

This appendix proves the results in Sec. 4.4. We first explain why mappings for unused content codes can be treated as random in the idealized model and how the resulting bounds apply to the keyed construction. We then connect changes in image features to stable content-code bits and candidate coverage. Finally, we prove the forgery bounds, the bounds between the TrajBind generator and the random-pattern generator, the bounds between the random-pattern generator and the unwatermarked generator, and sufficient conditions for verifying genuine watermarked images. The analysis uses pseudorandom functions (Goldreich et al., 1986; Bellare, 2006), concentration bounds for random signed sums (Hoeffding, 1963), margin-based robustness certificates (Hein & Andriushchenko, 2017), and correlation-based spread-spectrum watermarking (Cox et al., 1997).

C.1 Notation Summary

Table 5 summarizes the notation used in the analysis and its proofs.

Throughout this appendix, $\mathcal{B}_{\varepsilon,\gamma}(x^{\mathrm{tgt}})$ denotes all images satisfying both constraints in Eq. 1. The fixed-threshold results assume that τ is independent of the keyed mapping family. If keyed mappings are used to select τ, their codes must be included in the used-code set and their mappings counted in the budget, as described in App. A.5.

Table 5: Summary of key notation used in the theoretical analysis.

Notation	Meaning
κ, m, $m^\star$, r	Secret key, message, target message, and content code.
x, x^f, x^{tgt}	Image, forged image, and target image.
$\widehat{r}(x)$, $\mathcal{C}(x)$, $\mathcal{C}_L(x)$, L	Estimated code, candidate set, top-L candidates, and candidate limit.
$q(m,r)$, $R_{\kappa,r}$, $s(m,r)$	Signed codeword, keyed signed permutation, and watermark pattern.
$\ell(x)$, $\ell_r(x)$	Receiver vector in DCT order and its content-aligned form.
$S(x;m)$, τ, $\bar{\ell}(x)$	Verification score, threshold, and normalized receiver vector $\ell(x)/\|\ell(x)\|_2$.
$\mathcal{R}_{\mathrm{obs}}$, N, Q_v	Codes used in observed generations, number of observed generations, and verification queries.
$\mathcal{B}_{\varepsilon,\gamma}(x^{\mathrm{tgt}})$	Allowed target-image transformations under the image and semantic constraints.
$\overline{a}(x)$, $h_i(x)$	Average content feature and score for the ith content-code bit.
$\mathcal{J}_{\eta_a}(x^{\mathrm{tgt}})$, D_r	Stable-bit set and mismatch count with code r.
$\mathcal{S}_n^{\pm}$, $\Pr_{\mathrm{id}}$, $\mathbb{E}_{\mathrm{id}}$	Signed-permutation space, probability, and expectation in the idealized model.
$\epsilon_{\mathrm{key}}(Q)$	Keyed-to-idealized approximation error for at most Q distinct mappings.
Gen_κ, $\mathsf{Gen}^{\mathrm{rand}}$, Gen^0	Keyed, random-pattern, and unwatermarked generators.
Rep_N, $p_{\max}$, H_∞, Δ_N	Code repetition, maximum code probability, code min-entropy, and generator distribution gap.
a_{eff}, $a_{\min}$, e_ℓ, ν_ℓ	Effective amplitude, its lower bound, residual vector, and residual-norm bound.
β_{cov}, β_{ch}	Candidate-coverage and effective-channel failure probabilities.

For a content code r, the signed-permutation matrix is defined by $(R_{\kappa,r})_{\pi_{\kappa,r}(i),i} = \xi_{\kappa,r,i}$ for $i = 1,\dots,n$. Thus, $R_{\kappa,r}^\top R_{\kappa,r} = I_n$ and $s(m,r) = R_{\kappa,r}q(m,r)$. As defined in Sec. 4.3, the verifier first computes $y(x) = PE(2x-1)$ and then applies the key-independent score transformation to obtain $\ell(x)$. Its entries follow the selected DCT-component order. The vector $\ell_r(x) = R_{\kappa,r}^\top \ell(x)$ reverses the permutation and signs for code r and places the entries in codeword order. Therefore, $q(m,r)^\top \ell_r(x) = s(m,r)^\top \ell(x)$ and $S(x;m) = \max_{r' \in \mathcal{C}(x)} s(m,r')^\top \bar{\ell}(x)$ with $\bar{\ell}(x) = \ell(x)/\|\ell(x)\|_2$ for $\ell(x) \neq 0$, and $S(x;m) = -\infty$ for $\ell(x) = 0$, with $|\mathcal{C}(x)| \leq L$.

Let $\mathcal{S}_n^{\pm}$be the set of $n \times n$ signed-permutation matrices. In the idealized model, each distinct code r has an independent uniform matrix $R_r \in \mathcal{S}_n^{\pm}$. The same matrix is reused whenever that code

appears again. We write $\Pr_{\mathrm{id}}$ and $\mathbb{E}_{\mathrm{id}}$ for probability and expectation in this model. By Assumption 1, replacing the keyed mappings with these random mappings changes the probability of any event that an allowed computation can check by at most $\epsilon_{\mathrm{key}}(Q)$ when at most Q distinct mappings are used. A code is called unused when its mapping has not yet been used to construct a watermark pattern or compute a verification score. $\Pr_{\mathrm{id}}$ and $\mathbb{E}_{\mathrm{id}}$.

C.2 Additional Assumptions and Lemmas

The first two lemmas explain why a watermark pattern for an unused code is random in the idealized model and how to transfer the resulting probability bound to the keyed construction. The remaining lemmas connect bounded feature changes to stable content-code bits and candidate coverage.

Assumption 2 (Bounded variation of the averaged low-frequency DCT features)**.** *For a target image x^{tgt}, there is $\eta_a \geq 0$ such that $\|\overline{a}(x) - \overline{a}(x^{\mathrm{tgt}})\|_2 \leq \eta_a \quad$ for every $x \in \mathcal{B}_{\varepsilon,\gamma}(x^{\mathrm{tgt}})$.*

Lipschitz continuity gives one sufficient condition for this assumption. Since $\mathcal{D}_{\mathrm{low}}$ selects coordinates of an orthonormal DCT, it is 1-Lipschitz in ℓ_2. If $d_{\mathcal{X}}(x, x') = \|x - x'\|_2$ and the encoder E and views $\mathcal{V}_t$ are Lipschitz, then $\|\overline{a}(x) - \overline{a}(x')\|_2 \leq L_a\|x - x'\|_2$, with $L_a \leq \frac{2}{N_v}\sum_{t=1}^{N_v} \mathrm{Lip}(E)\,\mathrm{Lip}(\mathcal{V}_t)$. Every allowed image transformation satisfies $\|x - x^{\mathrm{tgt}}\|_2 \leq \varepsilon$, so $\eta_a = L_a\varepsilon$ satisfies Assumption 2.

Lemma 1 (Random watermark patterns)**.** *Let R be a uniformly random signed-permutation matrix and let $q \in \{-1, +1\}^n$ and $v \in \mathbb{R}^n$ be fixed vectors. Then Rq is uniform on $\{-1, +1\}^n$. For $\tau > 0$ and $v \neq 0$, $\Pr[q^\top R^\top v \geq \tau] \leq \exp\left(-\frac{\tau^2}{2\|v\|_2^2}\right)$. When $v = 0$, this probability is zero.*

The lemma bounds the probability of a large correlation with a random watermark pattern. It is a concentration bound for a sum with independent random signs (Hoeffding, 1963; Boucheron et al., 2013).

Proof. Choose a uniform permutation π of $\{1, \ldots, n\}$. Also choose n independent uniform signs $\xi_i \in \{-1, +1\}$, independently of π. The $n!2^n$ equally likely choicesgive all signed-permutation matrices. Write $s = Rq$, with $s_{\pi(i)} = \xi_i q_i$.

Fix π and $u \in \{-1, +1\}^n$. Since $q_i^2 = 1$, the equality $s = u$ holds exactly when $\xi_i = u_{\pi(i)}q_i$ for every i. This is one of the 2^n equally likely sign sequences. Therefore, $\Pr[s = u \mid \pi] = 2^{-n}$. Averaging over π gives $\Pr[s = u] = 2^{-n}$, so the entries of s are independent uniform signs.

Let $Z = s^\top v = \sum_{j=1}^n s_j v_j$. For any $\lambda > 0$, independence gives

$$\mathbb{E}[e^{\lambda Z}] = \prod_{j=1}^n \mathbb{E}[e^{\lambda s_j v_j}] = \prod_{j=1}^n \frac{e^{\lambda v_j} + e^{-\lambda v_j}}{2} = \prod_{j=1}^n \cosh(\lambda v_j) \leq \underbrace{\exp\left(\frac{\lambda^2}{2}\sum_{j=1}^n v_j^2\right)}_{(*)}. \tag{17}$$

The term (*) uses $\cosh t \leq e^{t^2/2}$, which follows from $(2j)! \geq 2^j j!$:

$$\cosh t = \sum_{j=0}^{\infty} \frac{t^{2j}}{(2j)!} \leq \sum_{j=0}^{\infty} \frac{(t^2/2)^j}{j!} = e^{t^2/2}. \tag{18}$$

Markov's inequality gives:

$$\Pr[Z \geq \tau] \leq e^{-\lambda\tau}\mathbb{E}[e^{\lambda Z}] \leq \exp\left(-\lambda\tau + \frac{\lambda^2\|v\|_2^2}{2}\right). \tag{19}$$

For $v \neq 0$, choose $\lambda = \tau/\|v\|_2^2$ to obtain $\Pr[q^\top R^\top v \geq \tau] \leq \exp[-\tau^2/(2\|v\|_2^2)]$. If $v = 0$, then $Z = 0$ and the probability is zero because $\tau > 0$. □

Lemma 2 (Random-mapping approximation)**.** *Under Assumption 1: **(i)** if an allowed computation uses mappings for Q distinct codes and can check an event W, then $|\Pr[W] - \Pr_{\mathrm{id}}[W]| \leq \epsilon_{\mathrm{key}}(Q)$; **(ii)** in the idealized model, after conditioning on the key-independent parameters, the random choices, the previous requests and outputs, and the codes and mappings already used, the mappings for all unused codes remain independent and uniform over $\mathcal{S}_n^{\pm}$.Here, the conditioned information does not depend on any unused mapping.*

Part (i) transfers a probability bound from the idealized model to the keyed construction. Part (ii)shows that earlier observations do not change the distribution of a mapping that has not been used (Bellare & Rogaway, 2006; Shoup, 2004). Therefore, it allows Lemma 1 to be applied to an unused candidate code even when that code is chosen using earlier observations. The random object is the mapping, not the content code itself.

Proof. **(i)** By Assumption 1, for any event W using at most Q distinct mappings, $|\Pr[W] - \Pr_{\mathrm{id}}[W]| \le \epsilon_{\mathrm{key}}(Q)$.

(ii) In the idealized model, each content code r is assigned an independent uniform matrix $R_r \sim \mathrm{Unif}(\mathcal{S}_n^{\pm})$, and the same matrix is reused whenever r appears again. Let $\mathcal{H}$ denote all earlier requests, outputs, and used mappings. Then, for any distinct unused codes $r_1, \dots, r_k$ and any $A_1, \dots, A_k \in \mathcal{S}_n^{\pm}$,

$$\Pr_{\mathrm{id}}\left[R_{r_1} = A_1, \dots, R_{r_k} = A_k \,|\, \mathcal{H}\right] = \prod_{j=1}^{k} |\mathcal{S}_n^{\pm}|^{-1} = |\mathcal{S}_n^{\pm}|^{-k}. \tag{20}$$

Therefore, the mappings for unused codes remain jointly independent and uniform over $\mathcal{S}_n^{\pm}$ after conditioning on the earlier requests, outputs, and used mappings. The same conclusion holds when an unused code is selected from the earlier requests and outputs. □

Let $g_i^\top$ be the ith row of G and let $h_i(x)$ be the ith component of $h(x)$. Then $h_i(x) = g_i^\top(\overline{a}(x) - \mu) - (\tau_c)_i$ and $\widehat{r}_i(x) = \mathbf{1}\{h_i(x) > 0\}$. Under Assumption 2, define the stable-bit set and the number of differences from code r on these bits by

$$\mathcal{J}_{\eta_a}(x^{\mathrm{tgt}}) = \left\{i : |h_i(x^{\mathrm{tgt}})| > \|g_i\|_2 \eta_a\right\}, \qquad D_r = d_H\left(r_{\mathcal{J}_{\eta_a}(x^{\mathrm{tgt}})}, \widehat{r}_{\mathcal{J}_{\eta_a}(x^{\mathrm{tgt}})}(x^{\mathrm{tgt}})\right). \tag{21}$$

Lemma 3 (High-margin bit stability)**.** *If $\|\overline{a}(x) - \overline{a}(x_0)\|_2 \le \eta$ with $\eta \ge 0$, then $\widehat{r}_i(x) = \widehat{r}_i(x_0)$ whenever $|h_i(x_0)| > \|g_i\|_2\eta$.*

The score $|h_i(x_0)|$ measures the distance from the bit's decision threshold. As in margin-based robustness certificates (Hein & Andriushchenko, 2017), the bit stays unchanged when the maximum score change is smaller than this margin.

Proof. Subtracting $h_i(x_0)$ from $h_i(x)$ gives:

$$\begin{aligned} h_i(x) - h_i(x_0) &= g_i^\top\left(\overline{a}(x) - \overline{a}(x_0)\right) \\ |h_i(x) - h_i(x_0)| &\le \|g_i\|_2 \|\overline{a}(x) - \overline{a}(x_0)\|_2 \\ &\le \|g_i\|_2 \eta \quad \text{(By Cauchy–Schwarz and Assumption 2)} \end{aligned} \tag{22}$$

If $h_i(x_0) > \|g_i\|_2\eta$, then $h_i(x) \ge h_i(x_0) - \|g_i\|_2\eta > 0$, so both estimated bits are one. If $h_i(x_0) < -\|g_i\|_2\eta$, then $h_i(x) \le h_i(x_0) + \|g_i\|_2\eta < 0$, so both estimated bits are zero. These are thetwo cases covered by $|h_i(x_0)| > \|g_i\|_2\eta$. □

The strict inequality prevents the score from reaching zero. This matters because the rule $\widehat{r}_i = \mathbf{1}\{h_i > 0\}$ assigns a zero score to bit zero.

Lemma 4 (Candidate-list bit-lip bound)**.** *Let $1 \le L \le 2^{k_r}$. For $\widehat{r}(x)$, index candidate codes by sets of flipped bits $F \subseteq \{1, \dots, k_r\}$, with $r_F = \widehat{r}(x) \oplus \mathbf{1}_F$. Suppose $\mathcal{C}_L(x)$ contains the codes corresponding to the first L distinct sets in increasing cost $C_x(F) = \sum_{i \in F} w_i(x)$, where $w_i(x) \ge 0$. When costs are equal, a set is placed before any larger set that contains it. Then every subset of a retained set is also retained, and each retained candidate satisfies $d_H(r_F, \widehat{r}(x)) = |F| \le \lfloor \log_2 L \rfloor$.*

The construction follows soft-decision decoding, which forms candidate codewords by flipping less reliable bits (Chase, 1972; Fossorier & Lin, 1995). The bound follows because retaining a set of d flipped bits also requires retaining all 2^d of its subsets.

Proof. Let $\mathcal{F}_L(x)$ contain the first L retained sets of flipped bits. For any $F \in \mathcal{F}_L(x)$ and $A \subsetneq F$,

$$C_x(F) - C_x(A) = \sum_{i \in F \setminus A} w_i(x) \ge 0. \tag{23}$$

Hence, $C_x(A) \le C_x(F)$. If the inequality is strict, A appears before F because it has a lower cost. If the costs are equal, the ordering rule still places A first. Hence $A \in \mathcal{F}_L(x)$.

A set F has $2^{|F|}$ distinct subsets, including itself. All are retained, so

$$2^{|F|} \le |\mathcal{F}_L(x)| = L. \tag{24}$$

It follows that $|F| \le \lfloor \log_2 L \rfloor$. Since r_F differs from $\widehat{r}(x)$ exactly on F, this proves the Hamming-distance bound. □

Lemma 5 (Candidate coverage from affine margins)**.** *Let $r = \phi_c(\widehat{x}_0^{(j_r)}; \Theta_\Phi)$ be the content code selected during generation, and let x be a received genuine watermarked image. Suppose $\eta_i \ge 0$ and $|g_i^\top(\overline{a}(x) - \overline{a}(\widehat{x}_0^{(j_r)}))| \le \eta_i$ for every i. Define $U(x) = \{i : |h_i(x)| \le \eta_i\}$. If $\mathcal{C}(x)$ contains every code obtained from $\widehat{r}(x)$ by flipping a subset of $U(x)$, then $r \in \mathcal{C}(x)$.*

This result identifies which bits may differ between the generation code r and the estimated content code $\widehat{r}(x)$ of the received image. The coverage condition requires all combinations of these uncertain bits to be included; a list limited to L candidates does not automatically satisfy it.

Proof. The definition of h_i gives:

$$\left| h_i(x) - h_i(\widehat{x}_0^{(j_r)}) \right| = \left| g_i^\top \left(\overline{a}(x) - \overline{a}(\widehat{x}_0^{(j_r)}) \right) \right| \le \eta_i. \tag{25}$$

For $i \notin U(x)$, we have $|h_i(x)| > \eta_i$. If $h_i(x) > \eta_i$, then $h_i(\widehat{x}_0^{(j_r)}) \ge h_i(x) - \eta_i > 0$. If $h_i(x) < -\eta_i$, then $h_i(\widehat{x}_0^{(j_r)}) \le h_i(x) + \eta_i < 0$. Thus, $r_i = \widehat{r}_i(x)$ for every $i \notin U(x)$. The two codes can differ only on $U(x)$, so r is obtained by flipping a subset of $U(x)$. By the condition in Lemma 5 that $\mathcal{C}(x)$ contains every such code, we have $r \in \mathcal{C}(x)$. □

C.3 Proofs of Theorems

C.3.1 Proof of Theorem 1

Proof. In the idealized random-mapping model, let $W = \{S(x^f; m^\star) \ge \tau\}$ be the event that the forged image passes verification, and let $O = \{\mathcal{C}(x^f) \cap \mathcal{R}_{\text{obs}} \ne \varnothing\}$ be the event that a candidate code was used in an observed generation. If $\ell(x^f) = 0$, the verifier rejects x^f and W does not occur. Otherwise, write $v = \bar{\ell}(x^f) = \ell(x^f)/\|\ell(x^f)\|_2$, so that $\|v\|_2 = 1$, and define $p_\tau = \min\{1, L\exp(-\tau^2/2)\}$.

Given T, the forged image x^f, target message $m^\star$, receiver vector v, candidate set $\mathcal{C}(x^f)$, and event O are fixed. On O^c, every candidate code is unused. Lemma 2(ii) leaves each of its mappings uniform and independent of the conditioned information, while (S1) fixes the signed codeword $q(m^\star, r')$ without using that mapping. If $\ell(x^f) \ne 0$, Lemma 1 with $\|v\|_2 = 1$ and a union bound give, on O^c,

$$\begin{aligned} \Pr_{\text{id}}[W \mid \mathsf{T}] &\le \sum_{r' \in \mathcal{C}(x^f)} \Pr_{\text{id}}\left[q(m^\star, r')^\top R_{r'}^\top v \ge \tau \mid \mathsf{T}\right] \\ &\le |\mathcal{C}(x^f)| \exp\left(-\frac{\tau^2}{2}\right) \le L \exp\left(-\frac{\tau^2}{2}\right). \end{aligned} \tag{26}$$

If $\ell(x^f) = 0$, W does not occur. A probability is also at most one. Therefore, $\Pr_{\text{id}}[W \cap O^c \mid \mathsf{T}] \le \mathbf{1}_{O^c} p_\tau$. Averaging over T gives:

$$\begin{aligned} \Pr_{\text{id}}[W] &\le \Pr_{\text{id}}[O] + \mathbb{E}_{\text{id}}[\mathbf{1}_{O^c} p_\tau] \\ &\le \Pr_{\text{id}}[O] + p_\tau. \end{aligned} \tag{27}$$

The observed generations and final verification together use mappings for $N + L$ distinct codes: at most one per generation and at most L in the final candidate list. Applying Lemma 2(i) once to W therefore gives

$$\text{ASR}_{\text{det}}(\mathcal{A}) \le \Pr_{\text{id}}[O] + \min\left\{1, L\exp\left(-\frac{\tau^2}{2}\right)\right\} + \epsilon_{\text{key}}(N + L), \tag{28}$$

which is Eq. 7.

The bound on term (B) holds for every forged image. Unlike a bound for the unnormalized score, it requires no assumption on $\|\ell(x^f)\|_2$. □

Corollary 1 (False-positive threshold)**.** *Assume the conditions of Theorem 1.*

For an image x generated independently of κ and a fixed message m, $\Pr[\mathrm{Ver}_\theta(x, m; \tau) = 1] \le L \exp\left(-\frac{\tau^2}{2}\right) + \epsilon_{\mathrm{key}}(L)$. For $\alpha \in (0, 1)$, choosing $\tau_\alpha = \sqrt{2 \ln(L/\alpha)}$ gives a false-positive bound of $\alpha + \epsilon_{\mathrm{key}}(L)$ without using the secret key to determine the threshold and without any assumption on the distribution of x. For a fixed set of M messages, replace L with LM in the exponential term and in τ_α. The number of distinct mappings remains L.

This corollary concerns verification only. The decoder in Algorithm 3 may return a CRC-valid message without requiring $S(x; m) \ge \tau$. Therefore, we do not assume $\mathrm{ASR}_{\mathrm{msg}} \le \mathrm{ASR}_{\mathrm{det}}$ and do not use Theorem 1 as a bound on $\mathrm{ASR}_{\mathrm{msg}}$.

Proof. In the idealized model, condition on x and the fixed key-independent parameters. The vector $\bar{\ell}(x)$ and candidate set $\mathcal{C}(x)$ are then fixed independently of the random mappings. Lemma 1 with $v = \bar{\ell}(x)$ and $\|v\|_2 = 1$ bounds each candidate's probability of reaching τ. For $\ell(x) \neq 0$, a union bound gives

$$\Pr_{\mathrm{id}}[S(x; m) \ge \tau \mid x, \Theta_\Phi] \le L \exp\left(-\frac{\tau^2}{2}\right). \tag{29}$$

If $\ell(x) = 0$, the acceptance probability is zero. Average over x and apply Lemma 2(i) with L distinct mappings. This adds $\epsilon_{\mathrm{key}}(L)$. Substituting τ_α makes the idealized bound equal to α. Because the conditional bound holds for every x, averaging over x uses no property of its distribution.

Let $\mathcal{M}_0$ be a fixed set of M messages. Applying the same bound to each pair $(m, r') \in \mathcal{M}_0 \times \mathcal{C}(x)$ gives

$$\Pr_{\mathrm{id}}\left[\max_{m \in \mathcal{M}_0} \max_{r' \in \mathcal{C}(x)} q(m, r')^\top R_{r'}^\top \bar{\ell}(x) \ge \tau \middle| x, \Theta_\Phi\right] \le LM \exp\left(-\frac{\tau^2}{2}\right). \tag{30}$$

There are LM scores, but each candidate's mapping is reused across the M messages. The approximation error therefore remains $\epsilon_{\mathrm{key}}(L)$, while the threshold uses LM instead of L. □

Corollary 2 (Detection-level forgery bound with verification queries)**.** *After observing N genuine watermarked generations, allow the adversary to make Q_v verification queries (x_j, m_j) and receive accept-or-reject decisions. Each query may depend on the earlier answers. Suppose $\tau > 0$. Under Assumption 1 and (S1)–(S2),*

$$\mathrm{ASR}_{\mathrm{det}}(\mathcal{A}) \le \Pr_{\mathrm{id}}\left[\mathcal{C}(x^f) \cap \mathcal{R}_{\mathrm{obs}} \neq \varnothing\right] + (Q_v + 1) L \exp\left(-\frac{\tau^2}{2}\right) + \underbrace{\epsilon_{\mathrm{key}}(N + (Q_v + 1)L)}_{(*)}. \tag{31}$$

The term (*) counts N mappings from generation and L from each verification query and the final check.

Proof. Consider the idealized random-mapping model. After the observed generations, define a restricted verifier that checks only codes in $\mathcal{R}_{\mathrm{obs}}$:

$$\overline{\mathrm{Ver}}(x, m) = \mathbf{1}\left\{\max_{r \in \mathcal{C}(x) \cap \mathcal{R}_{\mathrm{obs}}} q(m, r)^\top R_r^\top \bar{\ell}(x) \ge \tau\right\}, \tag{32}$$

where the maximum over an empty set is $-\infty$, and x is rejected when $\ell(x) = 0$, as by the full verifier. For comparison, consider a second run of the same adversary with the same internal randomness, but with its verification queries answered by the restricted verifier. Let $(\bar{x}_j, \bar{m}_j)$ be its jth query, and include its final forgery as check $Q_v + 1$ by setting $(\bar{x}_{Q_v+1}, \bar{m}_{Q_v+1}) = (\bar{x}^f, m^\star)$.

Let T_j contain all information before check j: the fixed key-independent parameters, adversary randomness, generation requests and outputs, the codes and mappings in $\mathcal{R}_{\mathrm{obs}}$, and earlier restricted-verifier requests and answers. Since the restricted verifier uses only mappings in $\mathcal{R}_{\mathrm{obs}}$, conditioning

on T_j fixes $(\bar{x}_j, \bar{m}_j)$, while Lemma 2(ii) implies that all mappings outside $\mathcal{R}_{\text{obs}}$ remain independent and uniform.

Let B_j be the event that some $r \in \mathcal{C}(\bar{x}_j) \setminus \mathcal{R}_{\text{obs}}$ has score at least τ for $\bar{m}_j$. By (S1), Lemma 1, and the bound on the candidate-list size, with $\|\bar{\ell}(\bar{x}_j)\|_2 = 1$ whenever $\ell(\bar{x}_j) \neq 0$, we have:

$$\Pr_{\text{id}}[B_j \mid \mathsf{T}_j] \leq L \exp\left(-\frac{\tau^2}{2}\right). \tag{33}$$

Averaging and taking a union bound over all $Q_v + 1$ checks gives:

$$\begin{aligned} \Pr_{\text{id}}\left[\bigcup_{j=1}^{Q_v+1} B_j\right] &\leq \sum_{j=1}^{Q_v+1} \mathbb{E}_{\text{id}}\left[\Pr_{\text{id}}[B_j \mid \mathsf{T}_j]\right] \\ &\leq (Q_v+1)L \exp\left(-\frac{\tau^2}{2}\right). \end{aligned} \tag{34}$$

Compare the full and restricted verifiers under the same mappings and adversary randomness. The restricted verifier can accept only through codes in $\mathcal{R}_{\text{obs}}$, so every restricted-verifier acceptance is also a full-verifier acceptance. If their verification results first differ at query j, then the full verifier accepts through a code outside $\mathcal{R}_{\text{obs}}$, and hence B_j occurs.

Let W denote successful final verification by the full verifier, and let $O = \mathcal{C}(x^f) \cap \mathcal{R}_{\text{obs}} \neq \varnothing$. If no earlier verification results differ, both runs produce the same forged image. On O^c, the restricted verifier rejects the final image, so W implies B_{Q_v+1}. Hence, $W \cap O^c \subseteq \bigcup_{j=1}^{Q_v+1} B_j$. Consequently, we obtain:

$$\Pr_{\text{id}}[W] \leq \Pr_{\text{id}}[O] + (Q_v+1)L \exp\left(-\frac{\tau^2}{2}\right). \tag{35}$$

At most $N + (Q_v + 1)L$ distinct content-code mappings are instantiated. Applying Lemma 2(i) once adds $\epsilon_{\text{key}}(N + (Q_v + 1)L)$, proving Eq. 31. □

C.3.2 Proof of Theorem 2

Proof. If $\mathcal{R}_{\text{obs}} = \varnothing$, then $\mathcal{C}(x) \cap \mathcal{R}_{\text{obs}} = \varnothing$ for every x. Otherwise, fix $x \in \mathcal{B}_{\varepsilon,\gamma}(x^{\text{tgt}})$ and $r \in \mathcal{R}_{\text{obs}}$. Counting differences on all bit positions gives at least as many as counting only those in $\mathcal{J}$. Since the estimated bits on $\mathcal{J}$ remain unchanged throughout the feasible set,

$$d_H(r, \widehat{r}(x)) \geq d_H(r_{\mathcal{J}}, \widehat{r}_{\mathcal{J}}(x)) = d_H\left(r_{\mathcal{J}}, \widehat{r}_{\mathcal{J}}(x^{\text{tgt}})\right) > t_c. \tag{36}$$

Every candidate $r' \in \mathcal{C}(x)$ has distance at most t_c from $\widehat{r}(x)$, so $r \notin \mathcal{C}(x)$. This holds for every observed code and every image in the feasible set. Thus, $\mathcal{C}(x) \cap \mathcal{R}_{\text{obs}} = \varnothing$ throughout that set. □

Corollary 3 (Explicit separation certificate)**.** *Suppose Assumption 2 holds for x^{tgt}, and construct $\mathcal{C}_L$ as in Lemma 4.Define the minimum over an empty $\mathcal{R}_{\text{obs}}$ as $+\infty$. For every $x \in \mathcal{B}_{\varepsilon,\gamma}(x^{\text{tgt}})$,*

$$\mathcal{C}_L(x) \cap \mathcal{R}_{\text{obs}} \neq \varnothing \quad \Longrightarrow \quad \min_{r \in \mathcal{R}_{\text{obs}}} D_r \leq \lfloor \log_2 L \rfloor, \tag{37}$$

where D_r is defined in Eq. 21. If this minimum is larger than $\lfloor \log_2 L \rfloor$, no observed code appears in the candidate list anywhere in the feasible set. If the adversary's output always belongs to this set and the verifier uses $\mathcal{C} = \mathcal{C}_L$, term (A) of Theorem 1 is at most

$$\Pr_{\text{id}}\left[\min_{r \in \mathcal{R}_{\text{obs}}} D_r \leq \lfloor \log_2 L \rfloor\right]. \tag{38}$$

Proof. For $x \in \mathcal{B}_{\varepsilon,\gamma}(x^{\text{tgt}})$ and $i \in \mathcal{J}_{\eta_a}(x^{\text{tgt}})$,

$$|h_i(x) - h_i(x^{\text{tgt}})| \leq \|g_i\|_2 \|\overline{a}(x) - \overline{a}(x^{\text{tgt}})\|_2 \leq \|g_i\|_2 \eta_a < |h_i(x^{\text{tgt}})|. \tag{39}$$

Therefore, Lemma 3 gives $\widehat{r}_{\mathcal{J}_{\eta_a}(x^{\text{tgt}})}(x) = \widehat{r}_{\mathcal{J}_{\eta_a}(x^{\text{tgt}})}(x^{\text{tgt}})$.

By Lemma 4, every $r' \in \mathcal{C}_L(x)$ satisfies $d_H(r', \widehat{r}(x)) \leq \lfloor \log_2 L \rfloor$. Apply Theorem 2 with $\mathcal{J} = \mathcal{J}_{\eta_a}(x^{\text{tgt}})$ and $t_c = \lfloor \log_2 L \rfloor$, and use the definition of D_r, the condition $D_r > t_c$ for every $r \in \mathcal{R}_{\text{obs}}$ is established. Equivalently, for any allowed forged image, we obtain:

$$\left\{\mathcal{C}_L(x^f) \cap \mathcal{R}_{\text{obs}} \neq \varnothing\right\} \subseteq \left\{\min_{r \in \mathcal{R}_{\text{obs}}} D_r \leq \lfloor \log_2 L \rfloor\right\}. \tag{40}$$

Taking probabilities in the idealized model proves the bound on term (A). □

C.3.3 PROOF OF THEOREM 3

Generator definitions. For a prompt c and message m, $\mathsf{Gen}_\kappa(c, m)$ uses new independent generation randomness and the content-dependent watermark pattern $s(m, r)$. The generator $\mathsf{Gen}^{\mathrm{rand}}$ follows the same pipeline but replaces this pattern with a new independent uniform vector in $\{-1, +1\}^n$ in each generation. The generator Gen^0 is the corresponding pipeline without watermark embedding.

Let $\Pr_{\mathrm{rand}}$ denote probability under $\mathsf{Gen}^{\mathrm{rand}}$. Define

$$\Delta_N = \sup_{\mathcal{D}} \left| \Pr\left[\mathcal{D}(\mathsf{Gen}^{\mathrm{rand}}) = 1\right] - \Pr\left[\mathcal{D}(\mathsf{Gen}^0) = 1\right] \right|,$$

where the supremum is over the allowed computational methods using at most N generation queries. Each query may depend on earlier generated images. Thus, Δ_N bounds the difference in output probabilities between $\mathsf{Gen}^{\mathrm{rand}}$ and the unwatermarked generator Gen^0. It is independent of κ and, by (S2), of the watermark messages. An additional argument is needed to show that Δ_N is small.

Content-code entropy. Let $r^{(j)}$ denote the code selected during generation j under $\mathsf{Gen}^{\mathrm{rand}}$. Let T_j contain all key-independent parameters available before generation j, including the previous prompts, messages, outputs, and codes $r^{(1)}, \ldots, r^{(j-1)}$, together with the current prompt and message. Assume a uniform upper bound $p_{\max} = 2^{-H_\infty}$ such that

$$\Pr_{\mathrm{rand}}[r^{(j)} = r \mid \mathsf{T}_j] \le p_{\max} = 2^{-H_\infty} \quad \text{for all } r \in \{0,1\}^{k_r},\ j \le N. \tag{41}$$

This bound must hold for every allowed prompt and every conditioning value for which the conditional probability is defined. It is a separate condition on the content-code distribution and does not follow from independent generation randomness or Assumption 1. The bound $\Pr[\mathrm{Rep}_N] \le \binom{N}{2} 2^{-H_\infty}$ is small when $N \ll 2^{H_\infty/2}$. The code length k_r alone is insufficient since $H_\infty \le k_r$.

Proof. Replace the keyed matrices by the idealized family $\{R_r\}_{r \in \{0,1\}^{k_r}}$. By (S1)–(S2), at most N generations use at most N distinct mappings. Lemma 2(i) gives

$$\left| \Pr[\mathcal{D}(\mathsf{Gen}_\kappa) = 1] - \Pr_{\mathrm{id}}[\mathcal{D} = 1] \right| \le \epsilon_{\mathrm{key}}(N), \tag{42}$$

where $\Pr_{\mathrm{id}}[\mathcal{D} = 1]$ refers to the method's output when the generator uses the idealized mappings.

Compare the idealized generator and $\mathsf{Gen}^{\mathrm{rand}}$ under the same randomness. If their first $j-1$ outputs are identical, they make the same jth request. By (S2), $r^{(j)}$ is selected before watermark insertion and is therefore identical in both generators.

If $r^{(j)}$ has not appeared earlier, its mapping is unused. Conditioning on all earlier information and on the current code and signed codeword, Lemmas 2(ii) and 1 imply that $R_{r^{(j)}} q(m^{(j)}, r^{(j)})$ is an independent uniform pattern. Using the same pattern in $\mathsf{Gen}^{\mathrm{rand}}$ and the same remaining randomness therefore gives the same image by (S2). Applying this step successively keeps the generated images identical until the first repeated code, which appears at the same generation in both cases. Therefore,

$$\Pr_{\mathrm{id}}[\mathrm{Rep}_N] = \Pr_{\mathrm{rand}}[\mathrm{Rep}_N], \qquad \left| \Pr_{\mathrm{id}}[\mathcal{D} = 1] - \Pr[\mathcal{D}(\mathsf{Gen}^{\mathrm{rand}}) = 1] \right| \le \Pr_{\mathrm{rand}}[\mathrm{Rep}_N]. \tag{43}$$

Together with Eq. 42, this proves the first inequality in Eq. 8.

The earlier codes are fixed given T_j. By Eq. 41,

$$\Pr_{\mathrm{rand}}\left[r^{(j)} \in \{r^{(1)}, \ldots, r^{(j-1)}\} \,\middle|\, \mathsf{T}_j \right] \le \sum_{i=1}^{j-1} \Pr_{\mathrm{rand}}[r^{(j)} = r^{(i)} \mid \mathsf{T}_j] \le (j-1) 2^{-H_\infty}. \tag{44}$$

Averaging over T_j and taking a union bound over generations gives

$$\Pr_{\mathrm{rand}}[\mathrm{Rep}_N] \le \sum_{j=2}^{N} (j-1) 2^{-H_\infty} = \binom{N}{2} 2^{-H_\infty}. \tag{45}$$

This proves the second inequality in Eq. 8.

Finally, the triangle inequality and the definition of Δ_N give

$$\begin{aligned}\left|\Pr[\mathcal{D}(\mathsf{Gen}_\kappa)=1]-\Pr[\mathcal{D}(\mathsf{Gen}^0)=1]\right| &\le \left|\Pr[\mathcal{D}(\mathsf{Gen}_\kappa)=1]-\Pr[\mathcal{D}(\mathsf{Gen}^{\mathrm{rand}})=1]\right|+\Delta_N \\ &\le \epsilon_{\mathrm{key}}(N)+\binom{N}{2}2^{-H_\infty}+\Delta_N,\end{aligned} \tag{46}$$

which is Eq. 9. □

C.4 Verification of genuine watermarked images

The following result gives sufficient conditions for accepting a genuine watermarked image, including after a benign transformation.

Theorem 4 (Verification of genuine watermarked images)**.** *Let r be the content code selected during generation, and let x be a received genuine watermarked image. Suppose $r \in \mathcal{C}(x)$ and there exist a_{eff} and e_ℓ such that*

$$\ell(x) = a_{\mathrm{eff}}\, s(m,r) + e_\ell, \qquad a_{\mathrm{eff}} \ge a_{\min} > 0, \qquad \|e_\ell\|_2 \le \nu_\ell. \tag{47}$$

If $\nu_\ell < \sqrt{n}\, a_{\min}$, then

$$S(x;m) \ge \sigma_{\min}, \qquad \sigma_{\min} = \sqrt{n}\,\frac{\sqrt{n}\, a_{\min} - \nu_\ell}{\sqrt{n}\, a_{\min} + \nu_\ell}. \tag{48}$$

If this lower bound is at least τ, the verifier accepts. More generally, suppose $\Pr[r \notin \mathcal{C}(x)] \le \beta_{\mathrm{cov}}$ and $\Pr\left[\nexists\, a_{\mathrm{eff}} \ge a_{\min}, e_\ell : \ell(x) = a_{\mathrm{eff}}\, s(m,r) + e_\ell,\ \|e_\ell\|_2 \le \nu_\ell\right] \le \beta_{\mathrm{ch}}$. Whenever $\nu_\ell < \sqrt{n}\, a_{\min}$ and $\sigma_{\min} \ge \tau$, the detection probability is at least $1 - \beta_{\mathrm{cov}} - \beta_{\mathrm{ch}}$. The two failure events need not be independent.

Writing $\rho_\ell = \nu_\ell/(\sqrt{n}\, a_{\min})$ for the relative residual, $\sigma_{\min} = \sqrt{n}\,(1-\rho_\ell)/(1+\rho_\ell)$. The bound depends on n and the relative residual, and not on the amplitude of $\ell(x)$.

Proof. Since $q(m,r) \in \{-1,+1\}^n$ and $R_{\kappa,r}^\top R_{\kappa,r} = I_n$, we have: $\|s(m,r)\|_2^2 = q(m,r)^\top R_{\kappa,r}^\top R_{\kappa,r} q(m,r) = n$. Eq. 47 gives

$$\begin{aligned} s(m,r)^\top \ell(x) &= n a_{\mathrm{eff}} + s(m,r)^\top e_\ell \\ &\ge n a_{\mathrm{eff}} - \|s(m,r)\|_2 \|e_\ell\|_2 \text{ (By Cauchy-Schwarz)} \\ &\ge \sqrt{n}\,(\sqrt{n}\, a_{\min} - \nu_\ell) > 0, \end{aligned} \tag{49}$$

where the last step uses $a_{\mathrm{eff}} \ge a_{\min}$ and $\nu_\ell < \sqrt{n}\, a_{\min}$. In particular, $\ell(x) \ne 0$. The triangle inequality gives $\|\ell(x)\|_2 \le a_{\mathrm{eff}}\|s(m,r)\|_2 + \|e_\ell\|_2 \le \sqrt{n}\, a_{\mathrm{eff}} + \nu_\ell$. Because $r \in \mathcal{C}(x)$ and the numerator is positive,

$$S(x;m) \ge \frac{s(m,r)^\top \ell(x)}{\|\ell(x)\|_2} \ge \frac{n a_{\mathrm{eff}} - \sqrt{n}\,\nu_\ell}{\sqrt{n}\, a_{\mathrm{eff}} + \nu_\ell} \ge \sigma_{\min}, \tag{50}$$

where the last step holds because $t \mapsto \sqrt{n}\,(t-\nu_\ell)/(t+\nu_\ell) = \sqrt{n}\,[1 - 2\nu_\ell/(t+\nu_\ell)]$ is nondecreasing for $t > 0$, applied at $t = \sqrt{n}\, a_{\mathrm{eff}} \ge \sqrt{n}\, a_{\min}$. Thus, the verifier accepts whenever $\sigma_{\min} \ge \tau$.

Under this threshold condition, rejection can occur only if the candidate set misses r or the conditions in Eq. 47 do not hold. Explicitly,

$$\{S(x;m) < \tau\} \subseteq \{r \notin \mathcal{C}(x)\} \cup \left\{\nexists\, a_{\mathrm{eff}} \ge a_{\min},\, e_\ell : \ell(x) = a_{\mathrm{eff}}\, s(m,r) + e_\ell,\ \|e_\ell\|_2 \le \nu_\ell\right\}, \tag{51}$$

The union bound then gives

$$\begin{aligned} \Pr[\mathrm{Ver}_\theta(x,m;\tau)=1] &= 1 - \Pr[S(x;m) < \tau] \\ &\ge 1 - \Pr[r \notin \mathcal{C}(x)] \\ &\quad - \Pr[\nexists\, a_{\mathrm{eff}} \ge a_{\min},\, e_\ell : \ell(x) = a_{\mathrm{eff}}\, s(m,r) + e_\ell,\ \|e_\ell\|_2 \le \nu_\ell] \\ &\ge 1 - \beta_{\mathrm{cov}} - \beta_{\mathrm{ch}}. \end{aligned} \tag{52}$$

□

D FORMULATION FOR FORGERY ATTACKS

We formulate the five attacks evaluated in the main paper using the notation of Sec. 3.1. The unwatermarked target image is x^{tgt}, while x^w and $\{x_i^w\}_{i=1}^N$ denote one or multiple observed watermarked references. For multi-bit target-message copying, these references carry $m^\star$, although the adversary need not know its bits. All observations and surrogate models must be allowed by Π. Under the verifier-free policy, the adversary cannot select attacks or decide when to stop based on private verification results.

For attacks that preserve the perceptual content of the target image, we use the feasible set defined in Sec. 3.1,

$$\begin{aligned}\mathcal{B}_{\varepsilon,\gamma}(x^{\mathrm{tgt}}) = \big\{x \in \mathcal{X} : d_{\mathcal{X}}(x, x^{\mathrm{tgt}}) \le \varepsilon,\\ d_{\mathrm{sem}}\big(\phi_{\mathrm{sem}}(x), \phi_{\mathrm{sem}}(x^{\mathrm{tgt}})\big) \le \gamma\big\}.\end{aligned} \tag{53}$$

Averaging. Averaging (Yang et al., 2024a) estimates a reusable additive component from watermarked images and a clean reference pool $\{x_j^0\}_{j=1}^{N_0}$. The unpaired estimator and the forged image are defined as:

$$\begin{aligned}\widehat{\delta} &= \frac{1}{N}\sum_{i=1}^{N} x_i^w - \frac{1}{N_0}\sum_{j=1}^{N_0} x_j^0,\\ x^f &= \mathrm{clip}_{[0,1]}\big(x^{\mathrm{tgt}} + \lambda_{\mathrm{avg}}\widehat{\delta}\big), \qquad \lambda_{\mathrm{avg}} > 0.\end{aligned} \tag{54}$$

With paired clean companions, $N_0 = N$ and $\widehat{\delta} = N^{-1}\sum_{i=1}^N (x_i^w - x_i^0)$. Such pairs require additional access under Π; they are not implied by observing watermarked outputs. Unpaired averaging can also retain differences between the clean-content distributions of the two pools. The attack transfers the estimated component without recovering the secret key. Clipping enforces the pixel range, whereas the transfer strength controls distortion.

Latent-Matching. Let $\widetilde{E}$ and $\widetilde{D}$ denote a surrogate VAE encoder and decoder, using the main paper's convention that the encoder receives images normalized to $[-1, 1]$ and the decoder returns images in $[0, 1]$. Set $\widetilde{z}_0^w = \widetilde{E}(2x^w - 1)$. Latent-Matching (Jain et al., 2025) modifies image pixels to approach this reference in the clean VAE latent space:

$$\min_{\delta:\, x^{\mathrm{tgt}}+\delta\in\mathcal{X}} \left\| \widetilde{E}\big(2(x^{\mathrm{tgt}} + \delta) - 1\big) - \widetilde{z}_0^w \right\|_2 + \lambda_{\mathrm{LM}}\|\delta\|_2. \tag{55}$$

Here $\lambda_{\mathrm{LM}} > 0$ penalizes pixel distortion, and the returned perturbation gives $x^f = x^{\mathrm{tgt}} + \delta$. The negative latent-distance term supplies the surrogate objective $J_{\mathcal{O}_{\mathcal{A}}}$. This attack requires only the surrogate encoder, not a diffusion denoiser or diffusion inversion. Its hard-constrained counterpart restricts $x^{\mathrm{tgt}} + \delta$ to the set in Eq. 53; this restriction is not implied by the regularization term.

Imprinting. Unlike Latent-Matching, Imprinting (Müller et al., 2025) compares initial-noise representations and differentiates through diffusion inversion. Let $\widetilde{I}$ denote surrogate DDIM inversion from a clean VAE latent to initial diffusion noise, with fixed inversion conditioning, such as an empty prompt. Define $\widetilde{z}_0^{\mathrm{tgt}} = \widetilde{E}(2x^{\mathrm{tgt}} - 1)$ and $\widetilde{z}_T^w = \widetilde{I}(\widetilde{z}_0^w)$. Imprinting optimizes a latent perturbation rather than an image-space perturbation:

$$\min_{\delta_z} \left\| \widetilde{I}\big(\widetilde{z}_0^{\mathrm{tgt}} + \delta_z\big) - \widetilde{z}_T^w \right\|_2, \qquad x^f = \widetilde{D}\big(\widetilde{z}_0^{\mathrm{tgt}} + \delta_z^{\mathrm{imp}}\big), \tag{56}$$

where δ_z^{imp} is the perturbation returned by a finite optimization run initialized at $\delta_z = 0$. The negative inversion-distance term is the optimization surrogate. The loss is applied directly to the optimized latent, without assuming that VAE decoding and re-encoding are exact inverses.

WMCopier. WMCopier (Dong et al., 2026) trains an unconditional diffusion model on the observed watermarked images $\{x_i^w\}_{i=1}^N$. Let ϵ_ψ denote its noise predictor. The training objective is

$$\min_{\psi} \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{t,\zeta}\left[\left\| \epsilon_\psi\left(\sqrt{\bar{\alpha}_t}\, x_i^w + \sqrt{1-\bar{\alpha}_t}\,\zeta, t\right) - \zeta \right\|_2^2 \right], \tag{57}$$

where $\zeta \sim \mathcal{N}(0, I)$, t is sampled from the training schedule, and $\bar{\alpha}_t$ is the cumulative signal coefficient.

Given a target image x^{tgt}, WMCopier applies shallow diffusion inversion followed by denoising with the learned model. Let $\widetilde{I}_\psi(\cdot; t_s)$ and $\widetilde{G}_\psi(\cdot; t_s)$ denote the inversion and denoising maps at a shallow noise level $0 < t_s < T$. The initial copied image is $x^{(0)} = \widetilde{G}_\psi\left(\widetilde{I}_\psi(x^{\mathrm{tgt}}; t_s); t_s\right)$. WMCopier then refines $x^{(0)}$ to increase its agreement with the learned watermarked-image distribution while keeping it close to the target image. We write the corresponding surrogate as $J_{\mathcal{O}_\mathcal{A}}(x, m^\star) = \log \widehat{p}_w(x)$, where $\widehat{p}_w$ is the distribution learned from the watermarked images. The refinement approximately follows

$$\begin{aligned} x^{(i+1)} &= x^{(i)} + \eta_{\mathrm{ref}} \left[\widehat{g}_\psi(x^{(i)}; \zeta_i) - 2\lambda_{\mathrm{copy}}(x^{(i)} - x^{\mathrm{tgt}})\right], \\ x^f &= \mathrm{clip}_{[0,1]}\big(x^{(K_{\mathrm{ref}})}\big), \qquad i = 0, \ldots, K_{\mathrm{ref}} - 1, \end{aligned} \tag{58}$$

where $\eta_{\mathrm{ref}} > 0$ is the update step, $\lambda_{\mathrm{copy}} > 0$ controls preservation of the target image, and K_{ref} is the number of refinement steps. At a fixed low noise level $t_\ell > 0$, the first term is estimated as

$$\widehat{g}_\psi(x; \zeta_i) = -\frac{\epsilon_\psi\left(\sqrt{\bar{\alpha}_{t_\ell}}\, x + \sqrt{1 - \bar{\alpha}_{t_\ell}}\, \zeta_i, t_\ell\right)}{\sqrt{1 - \bar{\alpha}_{t_\ell}}}, \tag{59}$$

with $\zeta_i \sim \mathcal{N}(0, I)$. This term encourages the forged image to follow the learned watermarked-image distribution, while the second term in Eq. 58 keeps it close to x^{tgt}. At low noise, $\widehat{g}_\psi$ approximates the gradient of $\log \widehat{p}_w(x)$ rather than being exactly equal to it.

E Implementation Details

E.1 Details for Distortion and Removal Attacks

We follow the WAVES benchmark (An et al., 2024) for distortion and removal attacks, using the same attack definitions and settings.

Distortion attacks. We consider eight common image transformations: Gaussian blur, brightness adjustment, JPEG compression, contrast adjustment, erasing, Gaussian noise, resized cropping, and rotation. WAVES controls each attack with a strength in $[0, 1]$. The full ranges are Gaussian blur with σ from 0 to 20 pixels, brightness and contrast factors from 1 to 2, JPEG quality from 90 to 10, erased area from 0 to 25%, Gaussian-noise standard deviation from 0 to 0.1 for images normalized to $[0, 1]$, retained crop area from 100% to 50%, and rotation from 0° to 45°. At strength 0.5, these correspond to $\sigma = 10$ pixels, brightness and contrast factors of 1.5, JPEG quality 50, erasing 12.5% of the image area, noise standard deviation 0.05, retaining 75% of the image area before resizing, and a 22.5° rotation.

For resized cropping, a square crop is sampled at a random position and resized to the original image size. Erasing replaces one square region with black pixels. Rotation is applied around the image center without expanding the canvas, and uncovered regions are filled with black. Gaussian blur uses the Gaussian standard deviation rather than a kernel size. JPEG compression uses the default Pillow encoder settings.

Removal attacks. We consider three regeneration-based removal attacks from WAVES. *VAE reconstruction* (`VAE1`) encodes and decodes the watermarked image once using the pretrained `bmshj2018` compression autoencoder at quality level 1. *Diffusion regeneration* (`Diff-40`) adds noise to the image and denoises it with Stable Diffusion v1.4 for 40 diffusion steps. *Rinse* repeats this regeneration process. We use the 2× Rinse setting from WAVES and test 20, 60, and 100 noising/denoising steps per cycle, denoted by `Rinse-20`, `Rinse-60`, and `Rinse-100`, respectively. Larger step counts produce stronger removal attacks.

E.2 TrajBind Implementation Details

Backbone and numerical conventions. Both variants use Stable Diffusion 2.1-base with its frozen text encoder, UNet, and VAE. We generate 512×512 RGB images with 50 deterministic DDIM steps,

classifier-free guidance scale 7.5, and an empty negative prompt. The latent has shape $4 \times 64 \times 64$, so $d = 16{,}384$. Step indices $j = 0, \ldots, 49$ follow denoising order. Generation uses FP16 model inference; carrier projections and the optimization variable use FP32. We encode images with the VAE posterior mode and multiply by the VAE scaling factor 0.18215; decoding divides by the same factor. Images in $[0, 1]$ are mapped to $[-1, 1]$ before VAE encoding or LPIPS evaluation.

DCT components. We use $n = 2{,}048$ unit-norm, channel-specific two-dimensional DCT-II components for the rows of P. For spatial frequencies (k_h, k_w) on the 64×64 grid, the normalized radius is $\rho = \sqrt{(k_h/63)^2 + (k_w/63)^2}/\sqrt{2}$. We partition $[0, 1]$ into 16 equal radial bins, numbered 0 through 15, and sample the watermark components without replacement from bins 2 through 15 using seed 42. The ordered component indices are fixed across images and variants. Content-code extraction uses all 432 channel-specific components in bins 0 and 1. Thus, the content and watermark projections have disjoint DCT support before the subsequent denoising and VAE operations.

Content-code extraction. At $j_r = 44$, we decode the predicted clean latent, quantize the image to 8 bits, and compute a $k_r = 24$-bit content code. We use $N_v = 8$ views: the original image and seven JPEG re-encodings with integer quality levels sampled uniformly from 30 through 70. Each view undergoes luminance histogram equalization, Gaussian blur with standard deviation 2 pixels, and a JPEG compression at quality 25. We encode these views and average their low-frequency coefficients before applying G. The matrix $G \in \mathbb{R}^{24 \times 432}$ has unit-norm Gaussian.

Watermark coding and keyed mapping. The application message contains $k = 512$ bits. We compute a 32-bit CRC over the concatenation of m and the 24-bit content code r, then polar-encode the 512 message bits and 32 CRC bits into a length-2,048 codeword. The CRC uses polynomial `0x104C11DB7`, zero initialization, no bit reflection, and no final XOR. We convert codeword bits to signs using $q = 2b - \mathbf{1}_n$. A 256-bit secret key and the serialized content code drive separate HMAC-SHA256 counter streams, with version tag `dmkc1`, domain labels `perm` and `sign`, and four-byte big-endian counters starting at zero. The permutation uses Fisher–Yates shuffling with rejection sampling; the sign stream supplies one bit per codeword position. Content codes are serialized with a four-byte big-endian bit-length prefix followed by most-significant-bit-first packed bits. The resulting mapping follows the convention $s_{\pi_{\kappa,r}(i)} = \xi_{\kappa,r,i} q_i$. The receiver scores $L_{\mathrm{ver}} = 1024$ candidate content codes for verification and decodes the first $L_{\mathrm{dec}} = 256$ in candidate order, using SCL decoding with list size 32.

F Additional Experiments

F.1 Analysis on White-box Adversary Attacks

We study a key-compromise setting in which the adversary has full access to the deployed watermark receiver. Unlike the five forgery attacks considered in the main paper, this attack constructs watermark evidence directly from the receiver and its keys, rather than transferring evidence from genuine watermarked images.

Threat model. The adversary knows the deployed receiver R, including its code, parameters, and secret keys, and is given an unwatermarked target image x_0, called the *cover image* in the attack implementation. It has no watermarked reference image and no access to the watermark embedder. Its goal is to construct a forged image x^f that the receiver accepts while preserving the appearance of x_0. For multi-bit methods, the attack targets a fixed message $m^\star$.

Detection-level success $\mathrm{ASR}_{\mathrm{det}}$ requires receiver acceptance, whereas message-level success $\mathrm{ASR}_{\mathrm{msg}}$ additionally requires exact recovery of $m^\star$. We use two frozen thresholds, τ_{real} and τ_{unw}, each set at a 0.1% false positive rate using real images and unwatermarked companion images from the same method, respectively. These are two reference distributions for threshold selection, not two subsets of attacked images. The first and second $\mathrm{ASR}_{\mathrm{det}}$ values correspond to these thresholds in that order.

Attack method. All methods are attacked through the same pixel-space variable. Starting from $\delta = 0$, we minimize

$$\min_{\delta} \ \mathcal{L}_{\mathrm{atk}}(x) + \lambda \ \mathrm{LPIPS}_{\mathrm{Alex}}(x, x_0), \quad \text{s.t} \quad x = \mathrm{clip}(x_0 + \delta, 0, 1), \quad \lambda = 1, \tag{60}$$

where $\mathcal{L}_{\text{atk}}$ encourages receiver acceptance, and $\text{LPIPS}(x, x_0)$, computed using the AlexNet model, penalizes perceptual differences from the original unwatermarked image x_0.

Surrogate models. For each method, we build a differentiable version of the receiver that allows gradients to pass from the watermark output back to the image pixels. The compromised key is used to determine the watermark signal that the attack aims to reproduce. During optimization, gradients are computed only through the differentiable parts of the receiver. Operations such as thresholding, voting, candidate-code search, and message decoding are performed by the deployed receiver when the current image is checked. For content-dependent methods, the attacker first computes the content code from the original image x_0 and keeps the corresponding watermark signal fixed during optimization.

StegaStamp. We use a PyTorch implementation of the deployed StegaStamp decoder, which takes a 400×400 image and outputs logits for 100 message bits. The attack minimizes the binary cross-entropy between these logits and the target message. During optimization, gradients pass through the decoder to the image pixels. The deployed receiver is then used to check both watermark detection and exact recovery of the 100-bit message.

Spherical. The receiver first applies 50-step DDIM inversion and then uses the signs of selected latent values to recover the watermark message. We use the same inversion process in the surrogate and optimize these latent values toward the signs corresponding to the target message. The attack therefore changes only the information used by the receiver, rather than matching a complete target latent.

SemBind-GS. The attacker first computes the content code of x_0 using the same content encoder as the receiver. This code determines the watermark sign pattern associated with the target message. The content code and its watermark pattern are kept fixed during optimization. Gradients pass through the 50-step inversion, while the deployed receiver recomputes the content code and performs the final message recovery when the image is checked.

For Spherical and SemBind-GS, let $v(x)$ denote the latent values whose signs are used by the receiver, and let $s^\star \in -1, +1^{d_v}$ denote the desired signs. We optimize

$$\mathcal{L}_{\text{atk}}(x) = \frac{1}{d_v} \sum_{i=1}^{d_v} \text{softplus}\left(\gamma - s_i^\star v_i(x)\right), \tag{61}$$

where γ controls how far each latent value is pushed toward the desired sign. This loss directly changes the signs used by the receiver.

TRAJBIND. The attacker first obtains the content code $r_0 = \widehat{r}(x_0)$ from the original image and constructs the target codeword $q(m^\star, r_0)$ using the compromised key. During optimization, gradients pass through the VAE encoder, the selected DCT components, and the soft watermark scores $\ell(x)$. The key-dependent mapping for r_0 then places these scores in codeword order as $\ell_{r_0}(x) = R_{\kappa,r_0}^\top \ell(x)$. The attack pushes the signs of $\ell_{r_0}(x)$ toward those of the target codeword $q(m^\star, r_0)$.

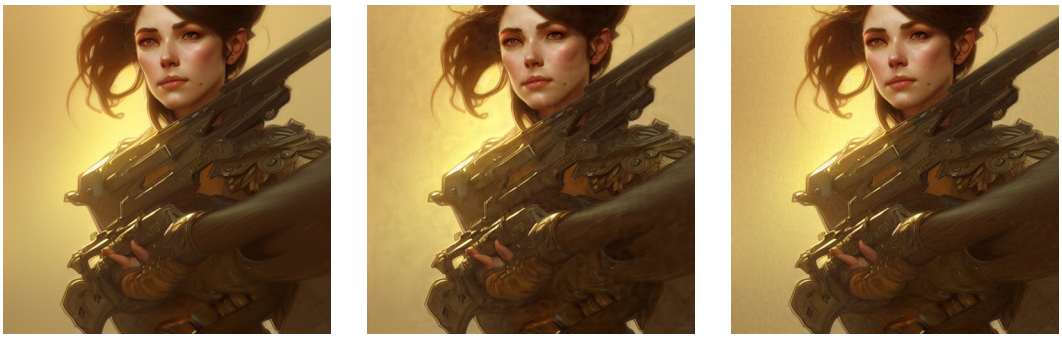
The deployed receiver is used for the final decision. In particular, it recomputes the content code from the current image, constructs the candidate-code set, computes the verification score, and performs message decoding. Therefore, making the surrogate scores agree with the target codeword does not by itself imply a successful forgery. Detection and exact-message recovery are checked separately.

Results. Tab. 6 shows three main findings. **First**, the white-box setting is challenging for all evaluated methods: every method reaches $\text{ASR}_{\text{det}} = 1.000/1.000$. The two values are computed using real images and unwatermarked companion images from MS-COCO dataset. This result shows that once the receiver and its secret key are available to the adversary, the detection criterion of every evaluated method can be bypassed. **Second**,even with full access to the receiver and secret key, the white-box attack does not reliably recover the complete target message for all images. Exact-message recovery succeeds for 61.0% of StegaStamp images, 34.1% of Spherical images, and 72.7% of SemBind-GS images. In contrast, the success rates are only 4.3% for TRAJBIND-MIN and 3.3% for TRAJBIND-OPT. Thus, although the white-box attack can bypass the detection criterion of all evaluated methods, recovering the complete target message remains much harder for TRAJBIND. **Third**, PSNR is computed between the forged image x^f and the original unwatermarked image x_0. A

Table 6: **White-box forgery results on DiffusionDB.** ASR_{det} is reported as A/B, where A uses the threshold set from real images and B uses the threshold set from unwatermarked companion images of the same method. ASR_{msg} is the exact-message success rate. PSNR is measured between the first accepted forged image and the original unwatermarked cover image x_0. Higher PSNR means smaller image change.

Methods	**ASR_{det}**	**ASR_{msg}**	**PSNR**
StegaStamp	1.000 / 1.000	0.610	36.7 dB
Spherical	1.000 / 1.000	0.341	38.0 dB
SemBind-GS	1.000 / 1.000	0.727	38.3 dB
TRAJBIND-MIN	1.000 / 1.000	0.043	43.4 dB
TRAJBIND-OPT	1.000 / 1.000	0.033	43.4 dB

higher PSNR indicates that the forged image is closer to the original image. Both TRAJBIND variants achieve a PSNR of 43.4 dB, compared with 36.7–38.3 dB for the other methods. Fig. 3 illustrates forged images from white-box attacks.



(a) Cover (Original) (b) StegaStamp (c) Spherical

(d) SemBind-GS (e) TRAJBIND-MIN (f) TRAJBIND-OPT

Figure 3: **Examples of white-box forged images.** All forged images are generated from the same original unwatermarked image. PSNR is computed between each forged image and the original image.

F.2 ABLATION STUDY

We study the contribution of three components in TRAJBIND-MIN: content binding, content correction, and multi-view aggregation. We remove one component at a time while keeping the remaining settings unchanged. Each entry in Tab. 7 reports ASR_{det} using the threshold set from real images / ASR_{det} using the threshold set from unwatermarked companion images / ASR_{msg}. Lower is better.

Table 7: **Ablation study of TRAJBIND-MIN under forgery attacks.** Each entry reports $\mathrm{ASR}_{\mathrm{det}}$ from real images / $\mathrm{ASR}_{\mathrm{det}}$ from unwatermarked companion images / $\mathrm{ASR}_{\mathrm{msg}}$. Lower is better.

Configuration	Averaging	Latent-Matching	Imprinting	Reprompting	WMCopier
TRAJBIND-MIN	0.000/0.017/0.000	0.000/0.000/0.000	0.000/0.020/0.000	0.100/0.100/0.030	0.000/0.000/0.000
w/o content binding	1.000/1.000/0.483	1.000/1.000/0.830	1.000/1.000/0.240	1.000/1.000/1.000	0.155/0.115/0.000
w/o content correction	0.067/0.000/0.000	0.000/0.000/0.000	0.020/0.000/0.000	0.820/0.700/0.030	0.000/0.000/0.000
w/o multi-view aggregation	0.033/0.000/0.000	0.000/0.000/0.000	0.040/0.020/0.000	0.640/0.500/0.050	0.000/0.000/0.000

Results. Removing content binding causes the largest increase in forgery success. For Averaging, Latent-Matching, Imprinting, and Reprompting, both detection-level success rates increase to 1.000, while $\mathrm{ASR}_{\mathrm{msg}}$ reaches 0.483, 0.830, 0.240, and 1.000, respectively. This shows that making the watermark pattern depend on the content code is the main component that prevents watermark evidence from being reused across different content. Content correction and multi-view aggregation are also important for Reprompting: removing content correction increases the two $\mathrm{ASR}_{\mathrm{det}}$ values from 0.100/0.100 to 0.820/0.700, while removing multi-view aggregation increases them to 0.640/0.500. Their effect is smaller for the other attacks. These results show that content binding provides the main forgery defense, while content correction and multi-view aggregation further improve robustness when the attack causes larger content changes.

F.3 ADDITIONAL RESULTS FOR FORGERY ATTACKS

Table 8: **Watermark forgery attack success rates on DiffusionDB.** $\mathrm{ASR}_{\mathrm{det}}$ reports detection-level forgery success, $\mathrm{ASR}_{\mathrm{msg}}$ message-level success. Lower is better. $\mathrm{ASR}_{\mathrm{det}}$ is computed using unmarked companion images as the reference set. "–" indicates that $\mathrm{ASR}_{\mathrm{msg}}$ is not applicable to zero-bit watermarking methods.

Method	Averaging		Latent-Matching		Imprinting		Reprompting		WMCopier	
	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$
TreeRing	1.000	–	0.763	–	1.000	–	1.000	–	0.030	–
RingID	1.000	–	1.000	–	1.000	–	1.000	–	0.343	–
ShallowDiffuse	0.917	–	0.997	–	1.000	–	1.000	–	0.004	–
SERUM	0.977	–	0.767	–	0.030	–	0.990	–	0.000	–
SemBind-TR	0.537	–	0.343	–	0.580	–	0.050	–	0.000	–
StegaStamp	0.933	0.003	0.000	0.000	0.570	0.000	0.970	0.000	0.785	0.000
Gaussian-Shading	1.000	0.663	1.000	0.577	1.000	0.970	1.000	1.000	0.732	0.000
StableSignature	0.008	0.000	0.003	0.000	0.000	0.000	0.000	0.000	0.999	0.311
DiffMark	0.865	0.065	0.983	0.000	1.000	0.000	1.000	1.000	0.000	0.000
Spherical	0.730	0.000	1.000	0.263	1.000	0.670	1.000	1.000	0.001	0.000
PRC	0.000	0.000	0.080	0.000	1.000	1.000	1.000	0.990	0.000	0.000
SemBind-GS	0.943	0.020	0.863	0.010	0.860	0.030	0.230	0.000	0.000	0.000
TRAJBIND-MIN	0.013	0.000	0.010	0.000	0.010	0.000	0.160	0.090	0.006	0.000
TRAJBIND-OPT	0.004	0.000	0.000	0.000	0.000	0.000	0.051	0.051	0.000	0.000

Results. Tabs. 8 and 9 show that both TRAJBIND variants keep forgery success low across the five attacks and two datasets. For TRAJBIND-MIN, $\mathrm{ASR}_{\mathrm{det}}$ is at most 0.020 for Averaging, Latent-Matching, Imprinting, and WMCopier; Reprompting is the hardest attack, with $\mathrm{ASR}_{\mathrm{det}}$ of 0.160 on DiffusionDB and 0.090 on DALL-E3. TRAJBIND-OPT further reduces Reprompting to 0.051 and 0.031, while all other $\mathrm{ASR}_{\mathrm{det}}$ values are at most 0.004. At the message level, both variants obtain zero $\mathrm{ASR}_{\mathrm{msg}}$ for all attacks except Reprompting; TRAJBIND-OPT reduces this value to 0.051 on DiffusionDB and 0.031 on DALL-E3. In contrast, several baselines obtain low success for some attacks but remain highly vulnerable to others, such as StableSignature under WMCopier and PRC under Imprinting and Reprompting. These results show that TRAJBIND provides more consistent forgery resistance across different attack types.

Table 9: **Watermark forgery attack success rates on DALL-E3.** $\mathrm{ASR}_{\mathrm{det}}$ reports detection-level forgery success, $\mathrm{ASR}_{\mathrm{msg}}$ message-level success. Lower is better. $\mathrm{ASR}_{\mathrm{det}}$ is computed using unmarked companion images as the reference set. "–" indicates that $\mathrm{ASR}_{\mathrm{msg}}$ is not applicable to zero-bit watermarking methods.

Method	Averaging		Latent-Matching		Imprinting		Reprompting		WMCopier	
	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$	$\mathrm{ASR}_{\mathrm{det}}$	$\mathrm{ASR}_{\mathrm{msg}}$
TreeRing	1.000	–	0.750	–	1.000	–	1.000	–	0.094	–
RingID	1.000	–	0.990	–	1.000	–	1.000	–	0.030	–
ShallowDiffuse	0.950	–	0.963	–	1.000	–	1.000	–	0.001	–
SERUM	0.963	–	0.477	–	0.030	–	1.000	–	0.001	–
SemBind-TR	0.260	–	0.070	–	0.170	–	0.030	–	0.000	–
StegaStamp	0.812	0.001	0.003	0.000	0.150	0.000	0.770	0.000	0.325	0.000
Gaussian-Shading	1.000	0.757	1.000	0.363	1.000	0.940	1.000	0.990	0.848	0.000
StableSignature	0.013	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.989	0.214
DiffMark	0.935	0.134	0.920	0.000	1.000	0.000	1.000	0.170	0.001	0.000
Spherical	0.443	0.000	0.967	0.067	1.000	0.640	1.000	1.000	0.001	0.000
PRC	0.003	0.000	0.060	0.000	1.000	0.940	1.000	0.990	0.000	0.000
SemBind-GS	0.773	0.000	0.430	0.000	0.520	0.000	0.190	0.000	0.000	0.000
TRAJBIND-MIN	0.020	0.000	0.007	0.000	0.000	0.000	0.090	0.070	0.003	0.000
TRAJBIND-OPT	0.000	0.000	0.000	0.000	0.004	0.000	0.031	0.031	0.000	0.000

F.4 CONTENT-CODE COLLISIONS IN FORGERY ATTACKS

Theorem 1 separates detection-level forgery into term (A), in which a candidate code of the forged image was used by an observed watermarked image, and term (B), in which the score arises from chance correlation with the pattern of an unused code. We measure this decomposition directly. For each forged image x^f, we record whether the committed content code of the watermarked image against which x^f is scored lies in its candidate list $\mathcal{C}_L(x^f)$ with $L = 1024$, and we split detection-level acceptance and exact-message recovery by this event.

Table 10: **Forgery outcomes split by content-code collision.** Counts pool MS-COCO, DiffusionDB, and DALL-E3 and both TRAJBIND variants. A collision means that the committed content code of the watermarked image against which the forgery is scored lies in the candidate list of the forged image ($L = 1024$). Acceptance uses the normalized score at the threshold set on real images; exact message means that the decoder returned the target payload. Averaging counts all five donor budgets (1, 50, 100, 200, 400) applied to the same covers. For Averaging with more than one donor and for WMCopier, a forgery draws on many observed images and the collision column checks one of their codes, so it is a lower bound on term (A) for these two attacks.

			Accepted		Exact message	
Attack	**Forged**	**Collisions**	with	without	with	without
Reprompting	600	71 (11.8%)	71/71	0/529	37/71	0/529
Latent-Matching	1,800	6 (0.3%)	6/6	1/1,794	0/6	0/1,794
Averaging	9,000	9 (0.1%)	0/9	15/8,991	0/9	0/8,991
Imprinting	600	0 (0.0%)	–	0/600	–	0/600
WMCopier	6,000	5 (0.1%)	0/5	5/5,995	0/5	0/5,995
Total	18,000	91 (0.5%)	77/91	21/17,909	37/91	0/17,909

Results. Tab. 10 gives three observations. First, every exact-message forgery (37 of 37) and 77 of the 98 detection-level acceptances occur with a collision. No forgery without a collision recovers the target message, and none of the 18,000 forgeries is decoded to a wrong message. Second, without a collision, acceptance is 21 of 17,909 forgeries (0.12%). This is comparable to the 0.1% false-positive operating point and below the term (B) bound $L\exp(-\tau^2/2)$, which is 0.7%–1.2% at the normalized thresholds used here. Third, collisions are rare for four of the five attacks (at most 0.3%) and occur mainly under Reprompting (11.8%), which regenerates an image from the donor's inverted initial latent and can therefore reproduce its coarse layout. Imprinting also targets the donor's initial latent

but keeps the cover image, and it produces no collision. The two variants behave alike, with 44 and 47 collisions and 19 and 18 exact-message forgeries for TRAJBIND-MIN and TRAJBIND-OPT, respectively. These results agree with Theorem 1: forgery success is determined by content-code collision, and the normalized score keeps acceptance without a collision at the false-positive level.

F.5 ADDITIONAL RESULTS FOR DISTORTION ATTACKS

Table 11: **Watermark robustness on MS-COCO under common image transformations**. Each entry reports TPR@0.1%FPR using real images / unmarked companion images as the respective reference sets. Higher is better.

Method	Clean	Blur	Brightness	Compression	Contrast	Noise	Rotation	Average
TreeRing	1.000/1.000	1.000/0.999	1.000/1.000	1.000/1.000	1.000/1.000	1.000/0.999	0.278/0.050	0.897/0.864
RingID	1.000/1.000	0.999/0.999	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	0.009/0.004	0.858/0.858
ShallowDiffuse	1.000/1.000	0.008/0.003	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	0.005/0.002	0.716/0.715
SERUM	1.000/1.000	0.988/0.960	1.000/1.000	1.000/1.000	1.000/1.000	1.000/0.999	0.069/0.018	0.865/0.854
SemBind-TR	0.994/0.992	0.140/0.077	0.993/0.990	0.980/0.970	0.987/0.987	0.953/0.940	0.000/0.000	0.721/0.708
StegaStamp	1.000/1.000	0.315/0.390	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	0.000/0.000	0.759/0.770
Gaussian-Shading	1.000/1.000	0.987/0.990	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	0.000/0.000	0.855/0.856
StableSignature	1.000/1.000	0.000/0.000	1.000/1.000	0.998/0.998	1.000/1.000	0.998/0.998	0.805/0.703	0.829/0.814
DiffMark	1.000/1.000	0.001/0.001	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	0.000/0.002	0.714/0.715
Spherical	1.000/1.000	0.024/0.040	1.000/1.000	0.997/0.997	1.000/1.000	0.994/0.994	0.000/0.001	0.716/0.719
PRC	1.000/1.000	0.033/0.093	0.993/0.997	0.990/0.997	1.000/1.000	0.977/0.980	0.000/0.000	0.713/0.724
SemBind-GS	0.998/0.998	0.875/0.875	1.000/1.000	1.000/1.000	1.000/1.000	1.000/1.000	0.000/0.000	0.839/0.839
TRAJBIND-MIN	0.986/0.986	0.879/0.924	0.873/0.857	0.987/0.987	0.967/0.953	0.983/0.983	0.004/0.000	0.811/0.813
TRAJBIND-OPT	1.000/1.000	0.949/0.971	0.960/0.961	1.000/1.000	0.982/0.988	0.995/0.995	0.004/0.000	0.841/0.845

Table 12: **Watermark robustness on DiffusionDB under common image transformations**. Each entry reports TPR@0.1%FPR. Higher is better. Average is computed over all columns, including Clean.

Method	Clean	Blur	Brightness	Compression	Contrast	Noise	Rotation	Average
TreeRing	1.000	0.999	1.000	1.000	1.000	1.000	0.013	0.859
RingID	1.000	1.000	1.000	1.000	1.000	1.000	0.030	0.861
ShallowDiffuse	1.000	0.002	1.000	1.000	1.000	0.999	0.000	0.714
SERUM	1.000	0.989	1.000	0.999	1.000	0.999	0.020	0.858
SemBind-TR	0.986	0.133	0.990	0.960	0.993	0.930	0.000	0.713
StegaStamp	1.000	0.496	1.000	1.000	1.000	1.000	0.000	0.785
Gaussian-Shading	1.000	0.996	1.000	1.000	1.000	1.000	0.000	0.857
StableSignature	0.999	0.000	0.999	0.987	0.999	0.993	0.826	0.829
DiffMark	1.000	0.000	1.000	1.000	1.000	0.999	0.000	0.714
Spherical	1.000	0.067	1.000	1.000	1.000	0.999	0.002	0.724
PRC	1.000	0.007	0.997	0.983	1.000	0.957	0.000	0.706
SemBind-GS	1.000	1.000	1.000	1.000	1.000	1.000	0.000	0.857
TRAJBIND-MIN	0.996	0.983	0.955	0.997	0.990	0.994	0.002	0.845
TRAJBIND-OPT	1.000	0.971	0.961	1.000	0.988	0.995	0.002	0.845

Results. Tabs. 11–13 show that TRAJBIND remains robust to most common image transformations. On MS-COCO, TRAJBIND-OPT reaches an average TPR of 0.841/0.845 and improves over TRAJBIND-MIN under blur and brightness, while keeping near-perfect recovery under compression and noise. On DiffusionDB, both variants obtain an average TPR of 0.845 and keep TPR above 0.95 under brightness, compression, contrast, and noise. On DALL-E3, the corresponding averages are 0.821 and 0.826, with strong robustness under the same transformations. Rotation is the main failure case for both variants, where TPR drops close to zero. Overall, the results show that the content-bound watermark remains stable under common pixel-level changes, while stronger geometric changes such as rotation remain challenging.

Table 13: **Watermark robustness on DALL-E3 under common image transformations**. Each entry reports TPR@0.1%FPR. Higher is better. Average is computed over all columns, including Clean.

Method	Clean	Blur	Brightness	Compression	Contrast	Noise	Rotation	Average
TreeRing	1.000	0.990	1.000	0.998	1.000	0.999	0.052	0.863
RingID	1.000	0.996	1.000	1.000	1.000	1.000	1.000	0.999
ShallowDiffuse	1.000	0.003	1.000	0.999	1.000	0.997	0.000	0.714
SERUM	1.000	0.980	1.000	1.000	1.000	0.999	0.048	0.861
SemBind-TR	0.923	0.100	0.923	0.840	0.927	0.817	0.003	0.648
StegaStamp	1.000	0.151	0.996	1.000	1.000	1.000	0.000	0.735
Gaussian-Shading	1.000	0.984	1.000	1.000	1.000	1.000	0.002	0.855
StableSignature	1.000	0.000	1.000	0.984	0.999	0.991	0.585	0.794
DiffMark	1.000	0.001	1.000	0.997	1.000	0.997	0.001	0.714
Spherical	1.000	0.047	0.999	0.999	1.000	0.993	0.000	0.720
PRC	1.000	0.017	1.000	0.957	1.000	0.953	0.007	0.705
SemBind-GS	0.981	0.656	0.969	0.969	0.938	0.938	0.010	0.780
TRAJBIND-MIN	0.977	0.920	0.940	0.976	0.955	0.973	0.004	0.821
TRAJBIND-OPT	1.000	0.898	0.931	1.000	0.957	0.976	0.020	0.826

Table 14: **Watermark robustness on DiffusionDB under removal attacks.** Each entry reports TPR@0.1%FPR. Higher is better. Average is computed over all columns, including Clean.

Method	Clean	VAE1	Diff-40	Rinse-20	Rinse-60	Rinse-100	Average
TreeRing	1.000	0.972	0.999	0.995	0.964	0.861	0.965
RingID	1.000	1.000	1.000	1.000	1.000	0.998	1.000
ShallowDiffuse	1.000	0.835	1.000	1.000	1.000	0.991	0.971
SERUM	1.000	0.996	1.000	1.000	1.000	0.999	0.999
SemBind-TR	0.986	0.583	0.943	0.953	0.900	0.803	0.861
StegaStamp	1.000	1.000	0.997	0.991	0.869	0.457	0.886
Gaussian-Shading	1.000	1.000	1.000	1.000	1.000	1.000	1.000
StableSignature	0.999	0.000	0.002	0.001	0.001	0.001	0.167
DiffMark	1.000	0.945	1.000	1.000	1.000	1.000	0.991
Spherical	1.000	0.913	1.000	1.000	0.999	0.998	0.985
PRC	1.000	0.313	1.000	0.983	0.913	0.697	0.818
SemBind-GS	1.000	1.000	1.000	1.000	1.000	1.000	1.000
TRAJBIND-MIN	0.996	0.997	0.984	0.983	0.971	0.958	0.982
TRAJBIND-OPT	1.000	1.000	0.996	0.996	0.988	0.980	0.993

F.6 RESULTS FOR REMOVAL ATTACKS

Results. Tabs. 14 and 15 show strong robustness to regeneration-based removal attacks. TRAJBIND-MIN obtains average TPRs of 0.982 on DiffusionDB and 0.955 on DALL-E3. TRAJBIND-OPT further improves these values to 0.993 and 0.997, respectively. In particular, TRAJBIND-OPT keeps TPR at 0.980 on DiffusionDB and 0.990 on DALL-E3 even under the strongest Rinse-100 setting. The gain over TRAJBIND-MIN becomes larger as Rinse becomes stronger, showing that the optimization step improves resistance to stronger regeneration. Both variants also remain substantially more robust than SemBind-TR under VAE reconstruction and the stronger Rinse settings.

F.7 QUALITATIVE VISUALIZATION

Fig. 4 provides a qualitative comparison between unwatermarked outputs and the two variants of TRAJBIND. The first three columns show the unwatermarked image, the image generated by TRAJBIND-MIN, and the image generated by TRAJBIND-OPT. The last two columns visualize the mean absolute RGB difference between each watermarked image and its unwatermarked counterpart. Across all three examples, both variants preserve the main content and overall appearance of the

Table 15: **Watermark robustness on DALL-E3 under removal attacks.** Each entry reports TPR@0.1%FPR. Higher is better. Average is computed over all columns, including Clean.

Method	Clean	VAE1	Diff-40	Rinse-20	Rinse-60	Rinse-100	Average
TreeRing	1.000	0.969	0.989	0.978	0.937	0.843	0.953
RingID	1.000	0.998	1.000	0.999	0.997	0.991	0.998
ShallowDiffuse	1.000	0.900	0.999	0.995	0.987	0.946	0.971
SERUM	1.000	0.995	1.000	0.999	0.999	0.999	0.999
SemBind-TR	0.923	0.567	0.823	0.823	0.730	0.680	0.758
StegaStamp	1.000	1.000	0.990	0.966	0.765	0.460	0.864
Gaussian-Shading	1.000	1.000	1.000	1.000	1.000	1.000	1.000
StableSignature	1.000	0.002	0.002	0.001	0.000	0.000	0.168
DiffMark	1.000	0.942	1.000	0.998	0.996	0.990	0.988
Spherical	1.000	0.908	0.998	0.996	0.990	0.990	0.980
PRC	1.000	0.530	0.963	0.940	0.833	0.643	0.818
SemBind-GS	0.981	1.000	1.000	1.000	1.000	1.000	0.997
TRAJBIND-MIN	0.977	0.980	0.963	0.959	0.939	0.911	0.955
TRAJBIND-OPT	1.000	1.000	1.000	1.000	0.990	0.990	0.997

original generation. The residual maps show that the watermark-induced changes remain visually subtle and are spread over the image rather than appearing as strong localized artifacts. Overall, the visualization supports the quantitative image-quality results and shows that the optimization step in TRAJBIND-OPT maintains high visual fidelity while embedding a robust content-bound watermark.

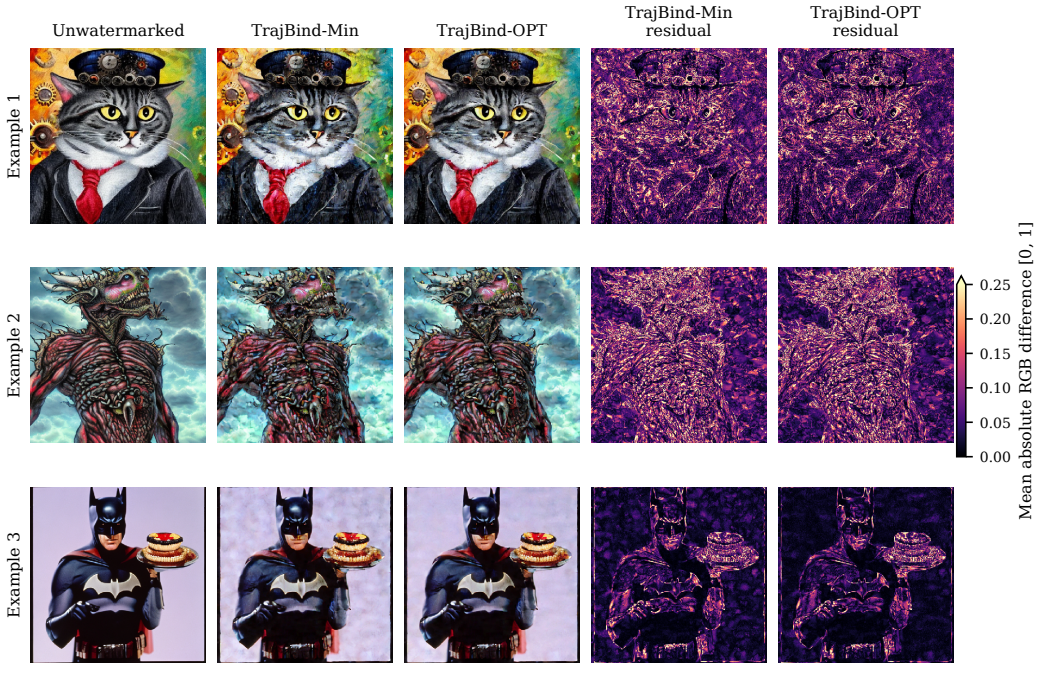


Figure 4: **Qualitative visualization of the full system.** From left to right, each row shows the unwatermarked image, the outputs of TRAJBIND-MIN and TRAJBIND-OPT, and the corresponding residual maps with respect to the unwatermarked image. The residual maps visualize the mean absolute RGB difference in $[0, 1]$. Both variants preserve the main visual content, while the differences remain small and visually subtle.

G LIMITATIONS

First, it is designed for diffusion pipelines that expose the denoising trajectory and a VAE latent space. Applying the same idea to generators with different internal representations may require a different choice of content-code features and watermark embedding space. Second, the current verifier is robust to common pixel-level changes and regeneration attacks, but it is not designed to be invariant to large geometric changes. In particular, the results in Sec. F.5 show weak recovery under rotation; adding geometric alignment or more invariant content features is a natural direction for future work. Third, as with other keyed watermarking methods, our threat model also assumes that the secret key remains private; secure key management is therefore required in deployment.