

Article

Cost-Guided Joint Mask-Perturbation Optimization with Attentive Decoding for Image Steganography

Qiuping Li , Xingyu Chen, Haixia Wang, Zemeng Wu and Mingyu Liu

School of Cyberspace Security, Gansu University of Political Science and Law, Lanzhou 730070, China; cxy136098495@gmail.com (X.C.)

* Correspondence: liqiuping@gsupl.edu.cn

Abstract

Digital image steganography aims to imperceptibly embed secret information into a cover image to enable covert communication. This paper focuses on image-level imperceptibility and recovery quality, and proposes a cost-guided joint mask-perturbation optimization with attentive decoding for image steganography method (CMAD). In an end-to-end differentiable framework, CMAD jointly optimizes the embedding mask, perturbation magnitude, and decoding-network parameters, thereby improving recovery accuracy while preserving imperceptibility. During optimization, the proposed AniCost cost-map guidance mechanism computes pixel-level embedding costs through wavelet-based anisotropy analysis, and uses a probability-map guidance loss to directly encourage the mask to activate in complex-texture regions and deactivate in smooth regions. The channel-attention-based decoding network is fine-tuned for each image pair during optimization to adapt to the current pair. Experimental results show that the stego images generated by CMAD achieve PSNR values of 55–59 dB, while the recovered secret images achieve PSNR values of 35–40 dB.

Keywords: image steganography; joint optimization; steganographic cost analysis; channel attention; decoder-adaptive steganography

1. Introduction

Digital image steganography aims to covertly embed secret information into a cover image for applications such as covert communication and copyright protection [1,2]. Its central objective is to achieve an effective balance among high recovery accuracy, the visual and statistical imperceptibility of the secret information, and security against steganalysis.

Early image steganography generally followed the distortion-minimization framework [3], in which handcrafted cost functions and syndrome-trellis codes (STCs) were used to embed information. The central problem was identifying suitable modification locations and magnitudes while avoiding detectable statistical anomalies. Because cost-function design relies heavily on expert prior knowledge, however, further performance improvements gradually reached a bottleneck. Against this background, prediction-error-ordering optimization in conventional steganographic frameworks has sought to exploit interpixel correlations more fully and thereby improve embedding efficiency [4].

In recent years, deep-learning-based steganography has transformed conventional cost design into an end-to-end differentiable optimization problem through data-driven methods, as exemplified by HiDDeN [5] and SteganoGAN [6]. Tang et al. [7] explored tensor subspaces for image-data representation, providing a potential basis for structured



Academic Editor: Danilo Pau

Received: 1 July 2026

Revised: 4 September 2026

Accepted: 6 September 2026

Published: 9 September 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

modeling of cover images in steganography. Frequency–spatial dual-path fusion steganography (FS-Stego) [8], generative-adversarial-network-based adversarial cost adjustment (SE) [9], and diffusion-model-based generative steganography (DPL-COT) [10] have further advanced feature fusion, resistance to detection, and coverless generation. Although these methods have achieved substantial gains in embedding capacity and visual quality, they generally require large quantities of paired training data.

To address the deployment difficulties caused by training and network-transmission overhead in deep-learning-based steganography, Kishore et al. [11] first proposed fixed neural network steganography (FNNS). FNNS exploits a neural network’s sensitivity to small perturbations and uses L-BFGS to optimize a cover image so that the output of a fixed decoding network matches the secret information. Luo et al. [12] subsequently introduced key control to prevent unauthorized extraction. Li et al. [13] proposed Cs-FNNS, which uses artificial-intelligence-generated content (AIGC) to make the cover reproducible and separable from the perturbation, substantially improving imperceptibility. Cheng et al. [14] introduced texture-aware and robust strategies in RFNNS to improve the survival of stego images under channel attacks.

FNNS requires the decoding-network weights to remain fixed while the stego image is optimized, which limits the decoder’s ability to adapt to image content. Existing studies have explored joint optimization of embedding locations and perturbation magnitudes as well as online guidance from cost information, but they remain bounded by the requirement that the decoding network be fixed. In contrast, CMAD incorporates decoder parameters into per-image-pair optimization and updates them jointly with a soft mask and perturbation in an end-to-end framework. It therefore performs joint image–decoder optimization rather than fixed neural network steganography. This paper proposes CMAD (Cost-Guided Joint Mask–Perturbation Optimization with Attentive Decoding for Image Steganography). Its principal contributions are as follows:

1. An adaptive steganographic framework is proposed that jointly optimizes image perturbations and the decoding network, unifying embedding-location selection, modification-magnitude optimization, and decoder-parameter adaptation within an end-to-end differentiable process.
2. An AniCost cost-map guidance mechanism is proposed. It computes pixel-level embedding costs through wavelet-based anisotropy analysis and uses a probability-map guidance loss to encourage the mask to activate in complex-texture regions and deactivate in smooth regions.
3. A channel-attention-based decoding network and a per-image-pair fine-tuning strategy are designed to improve recovery accuracy.

2. Related Work

2.1. Conventional Adaptive Image Steganography

Conventional image steganography generally follows the STC-based distortion-minimization framework [3]. This framework formulates steganography as a coding problem under a prescribed distortion cost and provides a general theoretical basis for subsequent methods. Researchers have proposed numerous adaptive cost functions to quantify more precisely the statistical detection risk introduced by pixel modification. HUGO [15] defines the cost as a weighted sum of differences in a high-dimensional feature space. WOW [16] uses directional high-pass filters to locate regions with complex textures. S-UNIWARD [17] employs a wavelet filter bank to extract directional residuals and guide cost allocation. HILL [18] combines high- and low-pass filters to concentrate modifications along image edges. These cost functions differ in how they quantify distortion, but all depend on analysis of and countermeasures against particular steganalytic features. They

also require manual tuning for the embedding domain and target security level, making it difficult to overcome the performance ceiling imposed by static rules.

2.2. Fixed Neural Network Steganography

FNNS has recently emerged as an important direction for avoiding the model-training and transmission costs of deep-learning-based steganography. Kishore et al. [11] introduced the FNNS paradigm, which exploits a neural network's sensitivity to small perturbations and directly optimizes image pixels with L-BFGS so that the output of a fixed decoding network matches the secret information. They designed three variants: FNNS-R with a random network, FNNS-D with a pretrained decoder, and FNNS-DE with initialization assisted by a pretrained encoder. At embedding rates no greater than 3 bpp, these variants can achieve bit-error rates close to 0%. Their stego-image distortion is nevertheless substantial, and security cannot be guaranteed when the decoding network is public.

Luo et al. [12] proposed a key-controlled scheme titled Securing Fixed Neural Network Steganography. Before a stego image is input to the decoding network, the scheme performs pixel-level encryption and applies Type-I, Type-II, and Type-III losses so that only a user holding the correct key can decode the message. Following the design philosophy of HILL [18], it also estimates an adaptive perturbation cost to improve the visual quality of the stego image.

To improve steganographic imperceptibility, Li et al. [13] proposed cover-separable fixed neural network steganography. The sender and receiver share a key and prompt and use a deep generative model, such as Stable Diffusion, to generate identical AIGC as the cover. The sender encodes the secret information as an extremely weak perturbation and adds it to the cover, constrains the perturbation range through a tanh reparameterization, and solves the optimization problem with Adam. The receiver regenerates the cover and separates the pure perturbation for decoding.

Tang et al. [19] combined FNNS with a secure diffusion model. They generate a cover with a fine-tuned Stable Diffusion model and embed secret information by optimizing a perturbation tensor of the same dimensions as the cover. A composite loss balances imperceptibility and resistance to steganalysis. Critical parameters, including the random seed and prompt, are encrypted end to end with ElGamal before transmission over a public channel. The receiver regenerates the cover and decoding network from the decrypted parameters and recovers the secret image from the perturbation.

Cheng et al. [14] focused on robustness against adversarial attacks and proposed robust fixed neural network steganography. Their method uses a local binary pattern (LBP) operator to encode the gray-level ordering of neighboring pixels and construct a fixed texture-weight map, thereby concentrating perturbations in complex-texture regions. It also employs an RSPG strategy that explicitly simulates channel attacks such as JPEG compression during optimization, making the perturbation more resistant to interference.

In summary, FNNS has developed into a framework that combines technologies such as AIGC cover generation and key control; external modules such as flip watermarking have also been used to improve FNNS-D [20]. Its central mechanism nevertheless remains “train the images, not the network.” A fixed decoding network constrains decoding capacity. CMAD adopts FNNS-style per-image-pair optimization but jointly optimizes the embedding mask and perturbation magnitude. While optimizing the mask and perturbation, it also updates the decoding network at a small learning rate so that the network dynamically adapts to the current cover–secret pair. Because the decoder is no longer fixed, CMAD does not belong to the FNNS paradigm. Instead, it extends FNNS from “optimizing only the image” to “optimizing the image while fine-tuning the decoder”.

3. Method

This section describes the proposed CMAD steganographic method. We first define the problem and present the overall framework, and then introduce the adaptive embedding framework for joint mask-perturbation optimization, the decoding-network design, and the fine-tuning strategy.

3.1. Problem Definition and Overall Framework

Let the cover image be $C \in [0, 1]^{H \times W \times 3}$ and the secret image be $S \in [0, 1]^{h \times w \times 3}$, where H, W and h, w are the heights and widths of the cover and secret images, respectively. The receiver aims to recover from the stego image a secret image $\hat{S}$ that is perceptually consistent with S . Before communication, the sender and receiver must pre-share the cover image C , the preprocessing operations, and the structural definition of the decoding network D . The sender transmits only the stego image C_s and the decoding-network weights θ_D over the public channel. The receiver applies the agreed preprocessing to the cover image C , reads the received C_s and normalizes it to $[0, 1]$, then computes $\delta = C_s - C$, and inputs δ together with θ_D to the decoding network D to recover the secret image $\hat{S}$. The end-to-end procedure is shown in Figure 1.

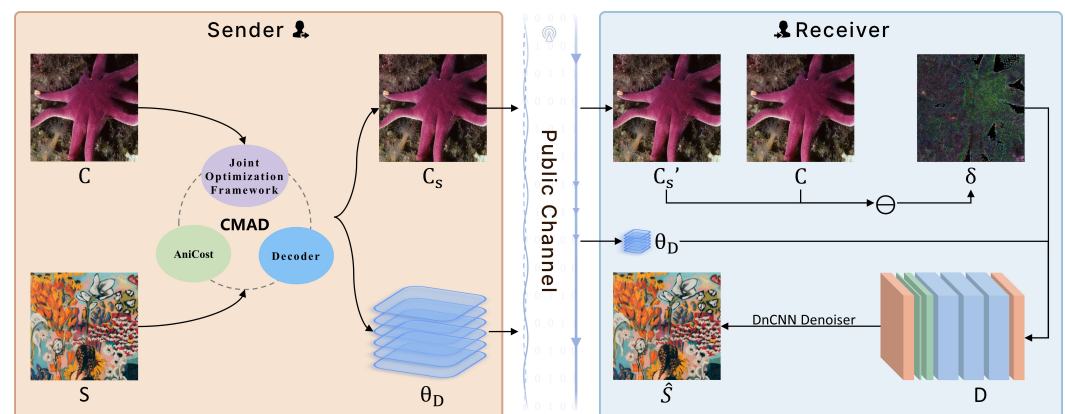


Figure 1. End-to-end system workflow. The sender inputs the cover image C and secret image S to the proposed CMAD method, which outputs the stego image C_s and fine-tuned decoding-network weights θ_D . Both are transmitted to the receiver through a public channel. Using the pre-shared cover C , the receiver extracts the perturbation δ from C_s and inputs δ together with θ_D to the decoding network D to recover the secret image $\hat{S}$.

Under this protocol, the adversary is assumed to have full access to the public channel and may intercept both the stego image C_s and the decoder weights θ_D . The original cover image C is the only secret information that does not enter the public channel and is the system's core security dependency. The receiver must possess the original cover image C to recover the secret image $\hat{S}$. The security evaluation in this paper is restricted to image-level steganographic detectability, namely, the ability of a steganalyzer to distinguish C_s from ordinary images. The additional weight file θ_D in the transmitted data is not included in the security evaluation, and protocol-level covertness is left for future work.

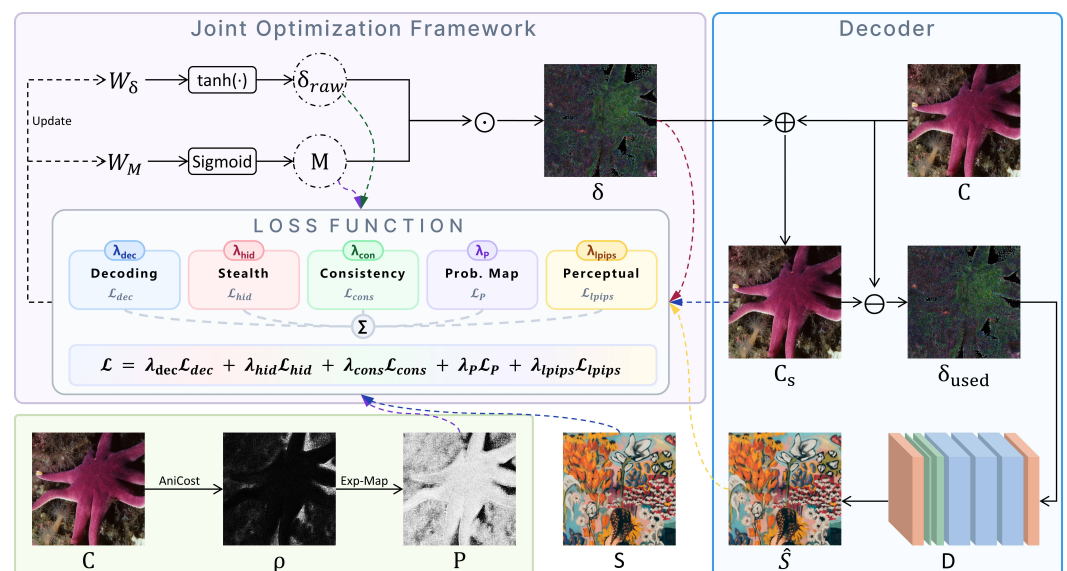
To accommodate different transmission settings, we further design a robust variant, CMAD-R. Standard CMAD targets a lossless channel and prioritizes recovery quality and imperceptibility. CMAD-R introduces differentiable channel-attack simulation during joint optimization so that recovery fidelity can be retained under lossy conditions such as JPEG compression, Gaussian noise, and contrast adjustment. The internal structure of CMAD, corresponding to the expanded CMAD block in Figure 1, is detailed in Section 3.2. The notation used throughout this paper is summarized in Table 1.

Table 1. Notation used in this paper.

Symbol	Description
C	Cover image in $[0, 1]^{H \times W \times 3}$
C_s	Stego image in $[0, 1]^{H \times W \times 3}$
C'_s	Transmitted stego image
S	Secret image in $[0, 1]^{h \times w \times 3}$
$\hat{S}$	Recovered secret image in $[0, 1]^{h \times w \times 3}$
δ	Final perturbation, $\delta = M \odot \delta_{\text{raw}} \in [-\epsilon, \epsilon]^{H \times W \times 3}$
δ_{raw}	Raw perturbation in $[-\epsilon, \epsilon]^{H \times W \times 3}$
δ_{used}	Extracted perturbation, $\delta_{\text{used}} = C'_s - C \in [-\epsilon - 0.5/255, \epsilon + 0.5/255]^{H \times W \times 3}$
M	Mask in $[0, 1]^{H \times W}$
ω_δ	Learnable perturbation variable
ω_M	Learnable soft-mask variable
θ_D	Decoding-network weights
D	Decoding network

3.2. Joint Perturbation-Embedding Optimization Framework

CMAD unifies embedding-location and modification-magnitude optimization within an end-to-end differentiable framework, thereby addressing their separation in existing methods. Its internal structure is shown in Figure 2. The framework parameterizes the optimization variables ω_δ and ω_M to produce a physically feasible raw perturbation δ_{raw} and mask M , which are fused into the final perturbation δ . The cost-guidance module applies the proposed AniCost algorithm to the cover image and computes a prior probability map P that guides the mask toward complex-texture regions. The decoder module adds the perturbation to the cover to synthesize C_s , separates δ_{used} from C_s using the cover, and recovers the secret image with D . The loss terms are aggregated and their gradients jointly drive updates to ω_δ and ω_M , enabling the mask and perturbation to adapt cooperatively. Sections 3.2.1–3.2.5 present each module in detail.

**Figure 2.** Internal structure and optimization workflow of CMAD.

3.2.1. Image Preprocessing

Before joint optimization begins, the cover and secret images undergo the same preprocessing pipeline, differing only in the target spatial size. During preprocessing, a given RGB image is read as an integer pixel array with values in $[0, 255]$, and all pixel values are divided by 255 to normalize to the floating-point range $[0, 1]$. The scaling

stage uses bilinear interpolation and reflective boundary padding and applies Gaussian low-pass filtering for anti-aliasing when the image size is reduced. After scaling, no quantization or integer rounding is performed. The processed cover image $C \in [0, 1]^{H \times W \times 3}$ and secret image $S \in [0, 1]^{h \times w \times 3}$ have pixel values stored as 32-bit floating-point numbers. This pipeline contains no random operations; under the same software environment and parameter settings, repeated execution on the same image yields identical results. Therefore, as long as both parties use the same preprocessing implementation, they are guaranteed to obtain the same cover image C .

3.2.2. Parameterization of the Perturbation and Mask

Given a cover image $C \in [0, 1]^{H \times W \times 3}$ and an upper perturbation bound $\epsilon \in (0, 1)$, we first compute a feasible interval for each pixel so that the modified pixel remains in $[0, 1]$ and its absolute modification does not exceed ϵ :

$$L_{i,j,c} = \max(-\epsilon, -C_{i,j,c}), \quad U_{i,j,c} = \min(\epsilon, 1 - C_{i,j,c}), \quad (1)$$

where $i \in \{1, \dots, H\}$, $j \in \{1, \dots, W\}$, and $c \in \{1, 2, 3\}$ index the row, column, and color channel, respectively. $L_{i,j,c}$ and $U_{i,j,c}$ are the minimum and maximum permissible perturbations at channel c of location (i, j) .

A learnable variable $\omega_\delta \in \mathbb{R}^{H \times W \times 3}$ is compressed to $[-1, 1]$ by a hyperbolic tangent and linearly mapped to $[L_{i,j,c}, U_{i,j,c}]$, yielding a boundary-constrained raw perturbation δ_{raw} :

$$(\delta_{\text{raw}})_{i,j,c} = L_{i,j,c} + (U_{i,j,c} - L_{i,j,c}) \frac{\tanh(\omega_\delta)_{i,j,c} + 1}{2}. \quad (2)$$

The mask $M \in [0, 1]^{H \times W}$ is a single-channel spatial weight map obtained by applying an elementwise sigmoid to the learnable variable $\omega_M \in \mathbb{R}^{1 \times H \times W}$:

$$M_{i,j} = \sigma(\omega_M)_{1,i,j} = \frac{1}{1 + \exp[-(\omega_M)_{1,i,j}]}. \quad (3)$$

The single-channel mask is expanded along the channel dimension into $M_3 \in [0, 1]^{H \times W \times 3}$, with all channels sharing the same spatial weights:

$$(M_3)_{i,j,c} = M_{i,j}, \quad c = 1, 2, 3. \quad (4)$$

The final perturbation $\delta \in \mathbb{R}^{H \times W \times 3}$ is obtained by channel-wise weighting of the raw perturbation with the mask:

$$\delta_{i,j,c} = (M_3)_{i,j,c} (\delta_{\text{raw}})_{i,j,c} = M_{i,j} (\delta_{\text{raw}})_{i,j,c}. \quad (5)$$

When $M_{i,j} \rightarrow 1$, the perturbation is retained almost completely; when $M_{i,j} \rightarrow 0$, it is suppressed and the pixel remains nearly unmodified. Joint optimization of ω_δ and ω_M enables the mask and perturbation to converge cooperatively under the same gradient feedback, adaptively allocating both embedding locations and modification magnitudes.

To simulate the 8-bit quantization that occurs when a digital image is stored, a straight-through estimator is used to quantize the stego image into discrete pixel values during optimization. The stego image $C_s = C + \delta$ is first scaled from $[0, 1]$ to $[0, 255]$, clipped and rounded, and then scaled back to $[0, 1]$:

$$(C_s^q)_{i,j,c} = \frac{1}{255} \text{round}(\text{clamp}(255(C_s)_{i,j,c}, 0, 255)). \quad (6)$$

Here, $\text{clamp}(x, 0, 255) = \min(\max(x, 0), 255)$, and $\text{round}(\cdot)$ denotes rounding to the nearest integer. The forward pass follows Equation (6), whereas the gradient passes directly through the quantizer during backpropagation, i.e., $\partial(C_s^q)_{i,j,c} / \partial(C_s)_{i,j,c} = 1$. The perturbation actually used by the decoding network is

$$\delta_{\text{used}} = C_s^q - C. \quad (7)$$

This design makes the optimization process more consistent with the physical properties of images after 8-bit storage and avoids an overoptimistic recovery estimate caused by a high-precision floating-point assumption. It should be noted that the cover image is not requantized to the 8-bit grid after resizing, and its pixels therefore remain continuous floating-point values. Consequently, even when the perturbation is zero, quantization may introduce into δ_{used} a small residual of at most $0.5/255 \approx 0.00196$. This residual is extremely small, and the decoding network can compensate for it adaptively during joint optimization; its effect on recovery quality is therefore negligible.

3.2.3. Cost-Map Guidance Mechanism

To guide the embedding perturbation toward complex-texture regions, we propose AniCost (Anisotropic Cost). Starting from the cover C , AniCost quantifies the sensitivity of each pixel to statistical detection through directional wavelet decomposition and anisotropy analysis. It outputs an embedding-cost map $\rho \in \mathbb{R}^{H \times W}$, in which a lower value denotes a location more suitable for embedding. An exponential mapping then converts ρ into a prior embedding-probability map $P \in [0, 1]^{H \times W}$ that serves as a spatial constraint in the loss function.

AniCost uses the Daubechies-8 (Db8) orthogonal wavelet. Its low-pass decomposition filter $\mathbf{l} \in \mathbb{R}^{16}$ and high-pass decomposition filter $\mathbf{h} \in \mathbb{R}^{16}$ form an orthogonal quadrature-mirror pair whose coefficients follow the canonical Db8 construction. Their outer products yield three two-dimensional directional filters:

$$\mathbf{F}_1 = \mathbf{l}\mathbf{h}^\top, \quad \mathbf{F}_2 = \mathbf{h}\mathbf{l}^\top, \quad \mathbf{F}_3 = \mathbf{h}\mathbf{h}^\top. \quad (8)$$

Here, $\mathbf{F}_k \in \mathbb{R}^{16 \times 16}$. $\mathbf{F}_1$ is low-pass horizontally and high-pass vertically, making it sensitive to horizontal edges; $\mathbf{F}_2$ behaves conversely and is sensitive to vertical edges; and $\mathbf{F}_3$ is high-pass in both directions and responds to diagonal textures. Together, the three filters cover the principal texture orientations.

For channel c of the cover, denoted $X^{(c)} \in \mathbb{R}^{H \times W}$, symmetric reflection padding of $P_{\text{pad}} = 18$ pixels is first applied on all sides, producing $\tilde{X}^{(c)} \in \mathbb{R}^{(H+2P_{\text{pad}}) \times (W+2P_{\text{pad}})}$. This suppresses artificial high-frequency components near boundaries caused by incomplete convolutional neighborhoods. $\tilde{X}^{(c)}$ is then convolved linearly in two dimensions with $\mathbf{F}_1$, $\mathbf{F}_2$, and $\mathbf{F}_3$. Because the Db8 filters have even length (16), each convolution output is shifted by half a pixel relative to the original grid. We therefore apply a positive circular shift of one pixel along both rows and columns, which is equivalent to relocating the origin from the filter center toward the higher-index side. Finally, the central $H \times W$ region is cropped to obtain the directional energy-response maps $E_1^{(c)}, E_2^{(c)}, E_3^{(c)} \in \mathbb{R}^{H \times W}$.

At pixel (i, j) , define the total directional energy $T^{(c)}$ and maximum directional energy $A^{(c)}$ as

$$T^{(c)}(i, j) = \sum_{k=1}^3 |E_k^{(c)}(i, j)| + \varepsilon_1, \quad A^{(c)}(i, j) = \max_{k \in \{1, 2, 3\}} |E_k^{(c)}(i, j)|, \quad (9)$$

where $\varepsilon_1 = 10^{-12}$ ensures numerical stability. $T^{(c)}$ describes the overall local texture strength, while $A^{(c)}/T^{(c)} \in [1/3, 1]$ measures directional concentration; values closer to one indicate stronger anisotropy. The single-channel pixel cost is

$$\rho^{(c)}(i, j) = \frac{1}{\frac{T^{(c)}(i, j)}{3} \left(1 + 2 \frac{A^{(c)}(i, j)}{T^{(c)}(i, j)}\right) + \varepsilon_2}, \quad \varepsilon_2 = 10^{-12}. \quad (10)$$

ε_1 prevents division by zero when $T^{(c)} = 0$, while ε_2 prevents reciprocal overflow if the entire denominator is zero. The expression has three progressively stronger effects. In a smooth region, $T^{(c)}$ is extremely small and the cost approaches a large saturation value, strongly suppressing embedding. As texture strength increases, the cost decreases. At equal texture strength, a larger directional concentration produces a larger $1 + 2A^{(c)}/T^{(c)}$ term and a still lower cost because strongly directional textures, such as edges, mask perturbations more effectively than random textures and are less readily detected by steganalysis.

The denominator consists of $T^{(c)}/3$, which increases monotonically with total texture strength, and an anisotropy ratio $A^{(c)}/T^{(c)} \in [1/3, 1]$, which increases monotonically with directional concentration and is amplified by a factor of two. Consequently, the denominator increases with both $T^{(c)}$ and $A^{(c)}/T^{(c)}$, whereas $\rho^{(c)}$ decreases monotonically. As $T^{(c)} \rightarrow 0$, the denominator approaches ε_2 and the cost approaches $1/\varepsilon_2$, strongly suppressing embedding in smooth regions. Increasing texture strength or anisotropy enlarges the denominator and lowers the cost. This monotonicity ensures high costs in smooth regions, lower costs in textured regions, and the lowest costs in strongly directional textured regions.

The costs of the three channels are summed pixelwise:

$$\rho(i, j) = \sum_{c=1}^3 \rho^{(c)}(i, j). \quad (11)$$

The median m is computed over pixels below the truncation limit $\rho < 10^8$. Median normalization is then applied as $\rho(i, j) \leftarrow \rho(i, j)/(m + \varepsilon_1)$, followed by clipping at 10^8 , which is the conventional extreme cost assigned to pixels regarded as completely unsuitable for embedding. Finally, an exponential mapping yields the prior embedding-probability map

$$P(i, j) = \exp[-\alpha \rho(i, j)]. \quad (12)$$

P is subsequently linearly scaled to $[0, 1]$ by dividing it by its maximum so that $\max P = 1$. Low-cost textured regions have P values close to one and are a priori suitable for embedding, whereas smooth regions have values close to zero. As the spatial weighting in the probability-map loss in Equation (16), P imposes an asymmetric constraint: in a high- P region, $1 - P \approx 0$ and the mask can expand freely; in a low- P region, $1 - P \approx 1$ and mask activation is strongly suppressed. The embedding region is thereby driven adaptively toward complex textures.

3.2.4. Loss Function

The joint objective is a weighted sum of five scalar losses, each averaged over its spatial dimensions. Let $M \in [0, 1]^{H \times W}$ denote the single-channel soft mask and let $M_3 = M \otimes \mathbf{1}_{1 \times 1 \times 3}$ be its three-channel broadcast.

The decoding loss measures pixelwise mean-squared error between the recovered secret image $\hat{S} \in [0, 1]^{h \times w \times 3}$ and the ground-truth secret image $S \in [0, 1]^{h \times w \times 3}$:

$$\mathcal{L}_{\text{dec}} = \frac{1}{3hw} \|\hat{S} - S\|_F^2, \quad (13)$$

where $\|\cdot\|_F$ is the Frobenius norm.

The concealment loss constrains the energy of the final perturbation $\delta = M_3 \odot \delta_{\text{raw}}$ and promotes imperceptibility:

$$\mathcal{L}_{\text{hid}} = \frac{1}{3HW} \|\delta\|_F^2. \quad (14)$$

The consistency loss forces the raw perturbation toward zero wherever the mask is closed, ensuring that modifications occur only where the mask permits them:

$$\mathcal{L}_{\text{cons}} = \frac{1}{3HW} \|(1 - M_3) \odot |\delta_{\text{raw}}|\|_1, \quad (15)$$

where $\|\cdot\|_1$ is the sum of the absolute values of all elements and $|\cdot|$ denotes elementwise absolute value.

The probability-map guidance loss uses the AniCost prior $P \in [0, 1]^{H \times W}$ to impose a spatially selective constraint on M . A bidirectional guidance loss both suppresses activation in low-probability regions and encourages activation in high-probability regions:

$$\mathcal{L}_P = \frac{1}{HW} [\|(1 - P) \odot M\|_1 + \|P \odot (1 - M)\|_1]. \quad (16)$$

The first term is suppressive. When P is close to one in a textured region, its weight $1 - P$ is almost zero and the mask is barely penalized; when P is close to zero in a smooth region, the weight approaches one and the mask is strongly suppressed. The second term is activating: when P is close to one but M has not opened sufficiently, the large value of $1 - M$ incurs a penalty that expands the mask over high-probability regions. Thus, the mask avoids ineffective modifications in smooth regions while remaining able to activate textured regions as required by the decoding loss, improving the trade-off between imperceptibility and secret-recovery accuracy.

The perceptual loss measures differences in a deep feature space to improve the subjective visual quality of the recovered image:

$$\mathcal{L}_{\text{lpips}} = \text{LPIPS}(S, \hat{S}), \quad (17)$$

where LPIPS is computed with an AlexNet backbone.

The total objective is

$$\mathcal{L} = \lambda_{\text{dec}} \mathcal{L}_{\text{dec}} + \lambda_{\text{hid}} \mathcal{L}_{\text{hid}} + \lambda_{\text{cons}} \mathcal{L}_{\text{cons}} + \lambda_P \mathcal{L}_P + \lambda_{\text{lpips}} \mathcal{L}_{\text{lpips}}. \quad (18)$$

The weights λ_{dec} and λ_{hid} are adjusted dynamically according to the current losses, as described in Section 3.2.5.

3.2.5. Coordinated Dynamic Weight Adjustment

To strengthen cooperative learning between the mask and perturbation, CMAD adaptively adjusts the decoding-loss weight λ_{dec} and concealment-loss weight λ_{hid} during optimization. Early in optimization, the perturbation is not fully formed and $\mathcal{L}_{\text{dec}}$ is high. λ_{dec} must therefore increase progressively to prioritize secret recovery, encourage the mask to open in textured regions, and concentrate perturbation energy in recoverable locations. As optimization proceeds and perturbation energy decreases, λ_{hid} increases concurrently, strengthening the imperceptibility constraint and suppressing perturbation magnitudes in flat regions. This coordinated adjustment allows the mask and perturbation to constrain

and adapt to one another rather than being optimized independently. Specifically, λ_{dec} grows according to the current decoding loss:

$$\lambda_{\text{dec}} \leftarrow \begin{cases} \min(1.012\lambda_{\text{dec}}, 50), & \mathcal{L}_{\text{dec}} > 3 \times 10^{-4}, \\ \min(1.010\lambda_{\text{dec}}, 50), & \mathcal{L}_{\text{dec}} > 2 \times 10^{-4}, \\ \min(1.005\lambda_{\text{dec}}, 50), & \mathcal{L}_{\text{dec}} > 1 \times 10^{-4}, \\ \lambda_{\text{dec}}, & \text{otherwise,} \end{cases} \quad (19)$$

and λ_{hid} grows according to the current perturbation energy:

$$\lambda_{\text{hid}} \leftarrow \begin{cases} \min(1.002\lambda_{\text{hid}}, 50), & \mathcal{L}_{\text{hid}} > 3 \times 10^{-6}, \\ \min(1.001\lambda_{\text{hid}}, 50), & \mathcal{L}_{\text{hid}} > 2 \times 10^{-6}, \\ \lambda_{\text{hid}}, & \text{otherwise.} \end{cases} \quad (20)$$

The $\min(\cdot)$ operation clips both weights to an upper bound of 50, ensuring numerical stability. To prevent instability caused by imbalanced gradient magnitudes among jointly optimized variables, a global gradient-norm clip of 1.0 is also applied to the perturbation variable, mask variable, and decoder parameters before every update. This keeps the perturbation mapping numerically stable and preserves coordinated convergence between spatial mask selection and perturbation-magnitude adjustment.

3.3. Decoding-Network Design and Fine-Tuning

After the joint-optimization framework determines the embedding region and perturbation magnitude, the decoding network must accurately recover the secret image from the perturbation. CMAD incorporates channel attention so that the network can adaptively focus on important feature channels. It also permits small updates to the decoding-network parameters during joint optimization, enabling dynamic adaptation to the current cover-secret pair. To further suppress high-frequency noise in the recovered image and improve its visual quality at the receiver, a pretrained DnCNN [21] is used for post-processing.

3.3.1. Network Architecture

The input to D is the perturbation $\delta_{\text{used}} \in \mathbb{R}^{3 \times H \times W}$ separated from the stego image, and the output is $\hat{S} \in \mathbb{R}^{3 \times h \times w}$. As shown in Figure 3, PixelUnshuffle first reduces the spatial resolution and increases the number of channels to 3psf^2 , where $\text{psf} = 2$:

$$\mathbf{X}_0 = \text{PixelUnshuffle}(\delta_{\text{used}}) \in \mathbb{R}^{12 \times H/2 \times W/2}. \quad (21)$$

$\mathbf{X}_0$ then passes through three convolutional layers. The first uses a 3×3 kernel with stride 1 and produces 96 channels, followed by LeakyReLU and instance normalization. The strides d_2 and d_3 of the second and third layers depend on the ratio of cover to secret image size: $(d_2, d_3) = (1, 2)$ for a ratio of two, $(2, 2)$ for a ratio of four, and $(1, 1)$ for equal sizes. Both kernels are 3×3 . The first two layers maintain 96 output channels, while the third produces $3\text{psf}^2 = 12$ channels and has no activation function. Its output is passed directly to the channel-attention module, which adaptively emphasizes channels that are more important to recovery.

The channel-attention layer (CALayer) [22] globally average-pools an input feature map, generates channel weights through two 1×1 convolutions, applies a sigmoid, and multiplies the weights channelwise with the original feature map:

$$\mathbf{X}_{\text{att}} = \mathbf{X}_{\text{in}} \odot \sigma(\text{Conv}_{1 \times 1}(\text{ReLU}(\text{Conv}_{1 \times 1}(\text{GAP}(\mathbf{X}_{\text{in}}))))). \quad (22)$$

Finally, the feature map passes through a sigmoid and PixelShuffle [23]. PixelShuffle reduces the channel count to three and restores the target spatial resolution $h \times w$, producing $\hat{S}$.

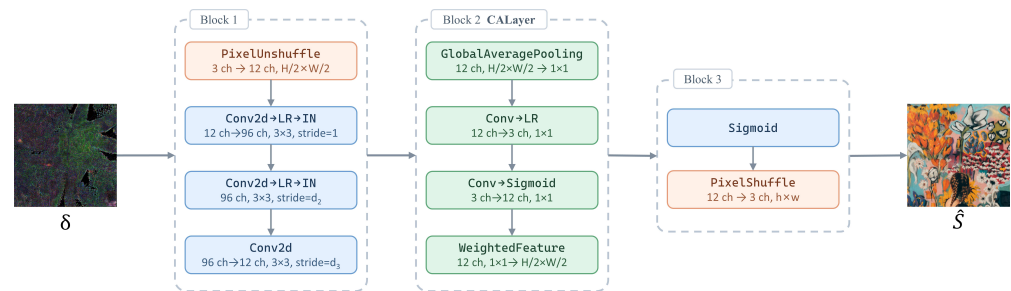


Figure 3. Architecture of the decoding network D .

3.3.2. Weight Initialization and Fine-Tuning

The weights θ_D of D are randomly initialized by default. To accelerate convergence and improve recovery quality, joint optimization may first be performed over multiple cover–secret pairs. The resulting decoder weights can then be loaded as pretrained weights $\theta_D^{(0)}$ to initialize subsequent per-image-pair fine-tuning.

During joint optimization, Adam jointly updates the perturbation variable ω_δ , soft-mask variable ω_M , and decoder parameters θ_D . The variables are divided into two parameter groups. The first contains ω_δ and ω_M and uses the global learning rate; the second contains θ_D and uses half the global learning rate. Adam adaptively adjusts the step size of every variable from its gradient history.

The learning rate is scheduled with ReduceLROnPlateau. The mode is set to minimization, the decay factor is 0.8, the patience is 100, and the minimum learning rate is 10^{-5} . After every optimizer update, the scheduler adjusts the learning rate according to changes in $\mathcal{L}_{\text{dec+hid}} = \mathcal{L}_{\text{dec}} + \mathcal{L}_{\text{hid}}$. This strategy retains the decoder’s existing feature-extraction ability while adapting it precisely to the current image pair.

3.3.3. Quantization Strategy

To generate a stego image, the soft mask modulates the continuous perturbation, which is then added to the cover. The resulting continuous stego image is mapped to the 8-bit integer domain: it is multiplied by 255, clipped to $[0, 255]$, quantized, and divided by 255 to recover a floating-point representation. The actual perturbation is the difference between the quantized stego image and the original cover and is input to the decoding network. Notably, the concealment loss is computed from the continuous perturbation, whereas the decoding loss is computed from the quantized, physically realized perturbation. Optimization therefore accounts for both a continuous-domain constraint and the actual written image.

Because quantization is nondifferentiable, a straight-through estimator (STE) approximates the rounding operation. The forward pass performs standard rounding, while the backward pass approximates the gradient with the identity map, i.e., the quantizer gradient is treated as one. Gradients can therefore traverse the nondifferentiable layer and enable end-to-end optimization. This mechanism allows effective updates to the perturbation and mask while accounting for actual 8-bit quantization.

4. Experiments

4.1. Experimental Setup

4.1.1. Datasets

Three public image datasets are used: COCO, CelebA, and DIV2K. In every experiment, the cover and secret images are resized to 512×512 and 256×256 , respectively.

The raw secret-image source rate is 6 bpp, defined as the original size of the secret image, $256 \times 256 \times 3 \times 8$ bits, divided by the number of cover-image pixels, 512×512 .

4.1.2. Evaluation Metrics

The methods are evaluated from the following perspectives.

Visual quality. Let X and Y denote the original and evaluated images, respectively, and N the total number of pixels. Before computing the metrics, both images are rescaled from $[0, 1]$ to the 8-bit intensity domain $[0, 255]$, so the pixel dynamic range is taken as $L = 255$, and the constants in PSNR, MAE, and SSIM are computed on this scale. The following four metrics are used:

- Peak signal-to-noise ratio (PSNR), which measures pixelwise mean-squared error,

$$\text{PSNR} = 10 \log_{10} \left(\frac{255^2}{\text{MSE}} \right), \quad \text{MSE} = \frac{1}{N} \sum_{i=1}^N (X_i - Y_i)^2.$$

A higher value indicates less distortion.

- Structural similarity index measure (SSIM), which jointly evaluates luminance, contrast, and structural consistency,

$$\text{SSIM} = \frac{(2\mu_X\mu_Y + C_1)(2\sigma_{XY} + C_2)}{(\mu_X^2 + \mu_Y^2 + C_1)(\sigma_X^2 + \sigma_Y^2 + C_2)},$$

where μ and σ are the mean and standard deviation, $C_1 = (0.01L)^2$, and $C_2 = (0.03L)^2$. SSIM lies in $[0, 1]$, and a value closer to one indicates greater structural similarity.

- Learned perceptual image patch similarity (LPIPS) [24], computed with a linearly calibrated AlexNet backbone in a deep feature space. A lower value indicates greater perceptual similarity.
- Mean absolute error (MAE),

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^N |X_i - Y_i|,$$

where a lower value indicates more accurate reconstruction.

Resistance to steganalysis. Four steganalysis tools evaluate the detectability of the stego images: (a) StegExpose [25], a conventional statistical steganalysis tool; (b) YeNet [26], a deep-learning-based steganalysis network; (c) SRNet [27], an end-to-end model based on a deep residual network; and (d) SiaStegNet [28], a steganalyzer based on a Siamese convolutional neural network.

4.1.3. Hyperparameters

The core hyperparameters of CMAD are set as follows. The total number of optimization iterations is 2000, the initial learning rate is 8×10^{-3} , and a ReduceLROnPlateau scheduler is used for dynamic adjustment. The upper bound on the perturbation magnitude is $\epsilon = 0.25$. The weights of the total-loss components are $\lambda_{\text{dec}} = 1.5$, $\lambda_{\text{hid}} = 0.4$, $\lambda_{\text{cons}} = 0.15$, $\lambda_P = 1.0$, and $\lambda_{\text{lpips}} = 0.01$. AniCost converts the cost map using the exponential mapping $P = \exp(-\alpha\rho)$ with a scaling factor of $\alpha = 0.2$. The decoder is randomly initialized, and its fine-tuning learning rate is half the global learning rate. All experiments are conducted on a single NVIDIA RTX 5060 GPU. For the experiments in Sections 4.2, 4.4, and 4.5, each group is run independently three times with random seeds 42, 23, and 54, and the statistical results are recorded; the observations for all cover–secret image pairs from the three runs are pooled for descriptive statistics, i.e., the mean and standard deviation are computed over all image-pair observations across the three runs.

4.2. Visual-Quality Analysis

To evaluate the performance of CMAD, we compare it with two recent FNNS-based methods, Cs-FNNS [13] and RFNNS [14]. All three methods were rerun under a unified experimental protocol using the same dataset splits, image-pairing rule, input resolutions, quantization procedure, payload setting, evaluation metrics, and DnCNN post-processing. For CelebA and COCO-val2017, the first 2000 images in the predefined dataset order were selected. The first 1000 images were used as covers and the remaining 1000 as secrets, forming 1000 cover–secret pairs according to their corresponding indices. DIV2K contains 900 images; the first 450 were used as covers and the remaining 450 as secrets, forming 450 image pairs. The recovered secret images produced by all three methods were processed using the same DnCNN model. The quantitative results are reported in Table 2, and Figure 4 presents representative CMAD results on the different datasets.

From Table 2, under the same image size and source-rate settings, CMAD outperforms Cs-FNNS and RFNNS in image quality, but at the cost of transmitting an additional message-specific decoder weight file of about 411 KB, which the comparison methods do not require. Specifically, the stego-image PSNR improves over Cs-FNNS by 13.06 dB, 17.83 dB, and 12.43 dB on COCO, CelebA, and DIV2K, respectively, and over RFNNS by 12.10 dB, 15.89 dB, and 13.13 dB; SSIM, LPIPS, and MAE follow the same trend. The recovered-secret PSNR improves over Cs-FNNS by 4.01 dB, 5.12 dB, and 3.90 dB, and over RFNNS by 5.93 dB, 7.19 dB, and 5.82 dB on the three datasets, with SSIM, LPIPS, and MAE of the recovered images also superior. However, the standard deviations of CMAD metrics are generally larger than those of the comparison methods, which are all below 1 dB, indicating that the influence of different cover images on optimization results is more pronounced for CMAD.

Table 2. Image-quality comparison of CMAD and state-of-the-art methods on different datasets (mean $\pm$ standard deviation). $\uparrow$ indicates larger is better, $\downarrow$ indicates smaller is better.

Dataset	Method	Stego Image				Recovered Secret			
		PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	MAE $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	MAE $\downarrow$
COCO	CMAD	55.31 $\pm$ 2.99	0.9990 $\pm$ 0.0032	0.0001 $\pm$ 0.0003	0.22 $\pm$ 0.16	35.98 $\pm$ 2.91	0.9653 $\pm$ 0.0218	0.0061 $\pm$ 0.0051	2.81 $\pm$ 0.95
	Cs-FNNS [13]	42.25 $\pm$ 0.99	0.9717 $\pm$ 0.0233	0.0017 $\pm$ 0.0013	1.47 $\pm$ 0.22	31.97 $\pm$ 2.38	0.9150 $\pm$ 0.0454	0.0241 $\pm$ 0.0131	4.91 $\pm$ 1.52
	RFNNS [14]	43.21 $\pm$ 0.97	0.9752 $\pm$ 0.0219	0.0015 $\pm$ 0.0012	1.36 $\pm$ 0.21	30.05 $\pm$ 2.40	0.9090 $\pm$ 0.0460	0.0272 $\pm$ 0.0138	5.58 $\pm$ 1.55
CelebA	CMAD	59.72 $\pm$ 1.96	0.9992 $\pm$ 0.0057	0.0001 $\pm$ 0.0014	0.08 $\pm$ 0.21	40.35 $\pm$ 0.60	0.9743 $\pm$ 0.0080	0.0069 $\pm$ 0.0028	1.80 $\pm$ 0.13
	Cs-FNNS [13]	41.89 $\pm$ 1.60	0.9496 $\pm$ 0.0501	0.0078 $\pm$ 0.0053	1.56 $\pm$ 0.39	35.23 $\pm$ 2.36	0.9292 $\pm$ 0.0742	0.0255 $\pm$ 0.0106	3.71 $\pm$ 1.50
	RFNNS [14]	43.83 $\pm$ 1.55	0.9531 $\pm$ 0.0488	0.0069 $\pm$ 0.0050	1.43 $\pm$ 0.37	33.16 $\pm$ 2.33	0.9226 $\pm$ 0.0730	0.0286 $\pm$ 0.0110	4.24 $\pm$ 1.48
DIV2K	CMAD	55.09 $\pm$ 3.14	0.9980 $\pm$ 0.0042	0.0001 $\pm$ 0.0003	0.22 $\pm$ 0.18	36.75 $\pm$ 2.72	0.9731 $\pm$ 0.0176	0.0063 $\pm$ 0.0051	2.62 $\pm$ 0.84
	Cs-FNNS [13]	42.66 $\pm$ 0.91	0.9766 $\pm$ 0.0252	0.0020 $\pm$ 0.0047	1.42 $\pm$ 0.28	32.85 $\pm$ 2.18	0.9397 $\pm$ 0.0451	0.0229 $\pm$ 0.0123	4.40 $\pm$ 1.18
	RFNNS [14]	41.96 $\pm$ 0.90	0.9500 $\pm$ 0.0239	0.0064 $\pm$ 0.0045	1.50 $\pm$ 0.26	30.93 $\pm$ 2.16	0.9332 $\pm$ 0.0445	0.0258 $\pm$ 0.0126	5.01 $\pm$ 1.20

Figure 5 compares the RGB-channel histograms of the cover and stego images. Their distributions almost completely overlap, indicating that the CMAD perturbation causes no appreciable shift in the first-order pixel distribution.

Figure 6 visually compares CMAD and Cs-FNNS [13] on COCO and DIV2K. It shows the residual between each stego and cover image, magnified tenfold, and the residual between each recovered and original secret image, also magnified tenfold. CMAD introduces an extremely weak stego residual that remains almost entirely black even after magnification, while the Cs-FNNS residual is similarly weak. The recovered-secret residual of CMAD is clearly smaller than that of Cs-FNNS.

4.3. Resistance to Steganalysis

Consistent with the threat model defined in Section 3.1, the evaluation in this subsection considers only the stego image C_s and does not consider the fine-tuned decoder

weights θ_D transmitted as part of the complete CMAD protocol. Accordingly, the reported detection accuracy quantifies image-level steganographic detectability rather than the detectability of the complete protocol.

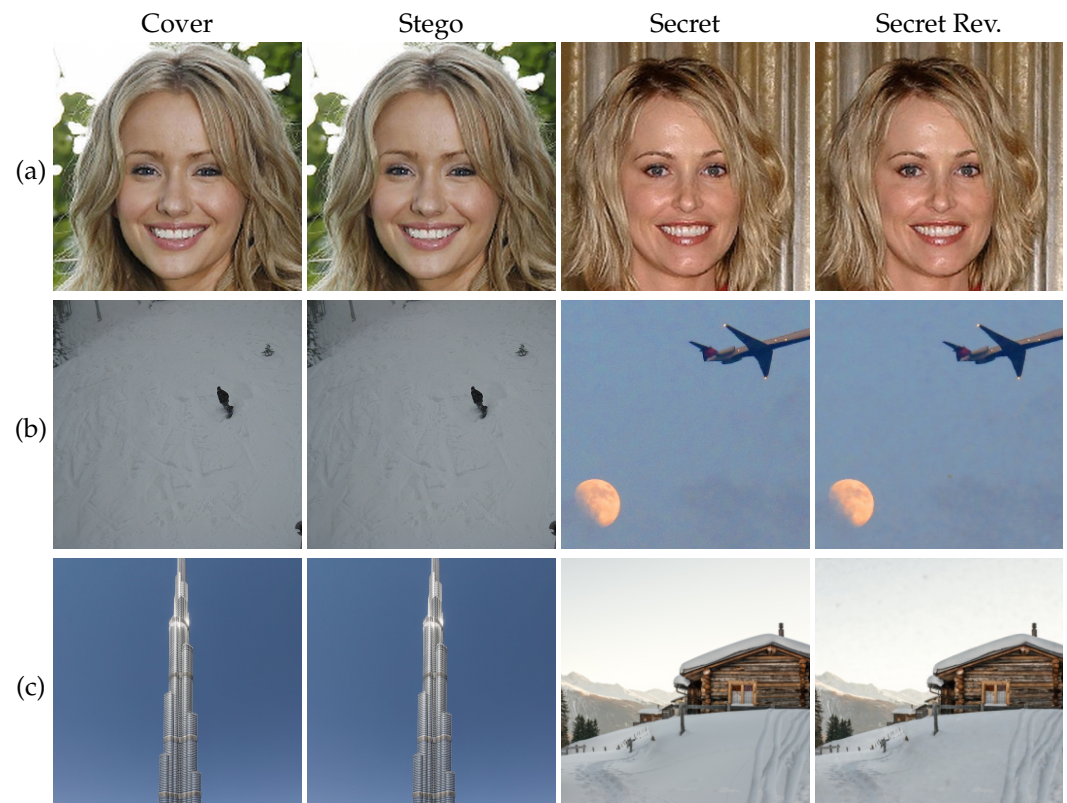


Figure 4. Representative results generated by the proposed method: (a) CelebA; (b) COCO; and (c) DIV2K. The columns show the cover, stego image, secret image, and recovered secret, respectively.



Figure 5. Comparison of RGB-channel histograms for representative experimental images. The first row shows the original images, and the second row shows the corresponding RGB-channel histograms (red, green, and blue curves represent the respective color channels).

For the deep-learning detectors, we consider two training assumptions: a universal detector and a CMAD-specific detector. The training and validation data are drawn from the COCO and CelebA datasets. We first exclude the first 2000 images from each dataset because these images were already used for the visual-quality evaluation in Section 4.2. From the remaining images, 1500 cover images are randomly selected from each dataset,

yielding 3000 covers in total. Both detector settings use these 3000 covers to generate stego images. For the universal detector, 1000 stego images are generated by Cs-FNNS, 1000 by StegGAN, 500 by S-UNIWARD, and 500 by WOW. For the CMAD-specific detector, CMAD generates one corresponding stego image for each cover. The resulting 3000 cover–stego pairs are then randomly divided in an 8:2 ratio into 2400 training pairs and 600 validation pairs. The independent test set consists of the 2450 image pairs already generated in Section 4.2, including 1000 COCO pairs, 1000 CelebA pairs, and 450 DIV2K pairs. All detection accuracies reported in Table 3 are obtained on this independent test set.

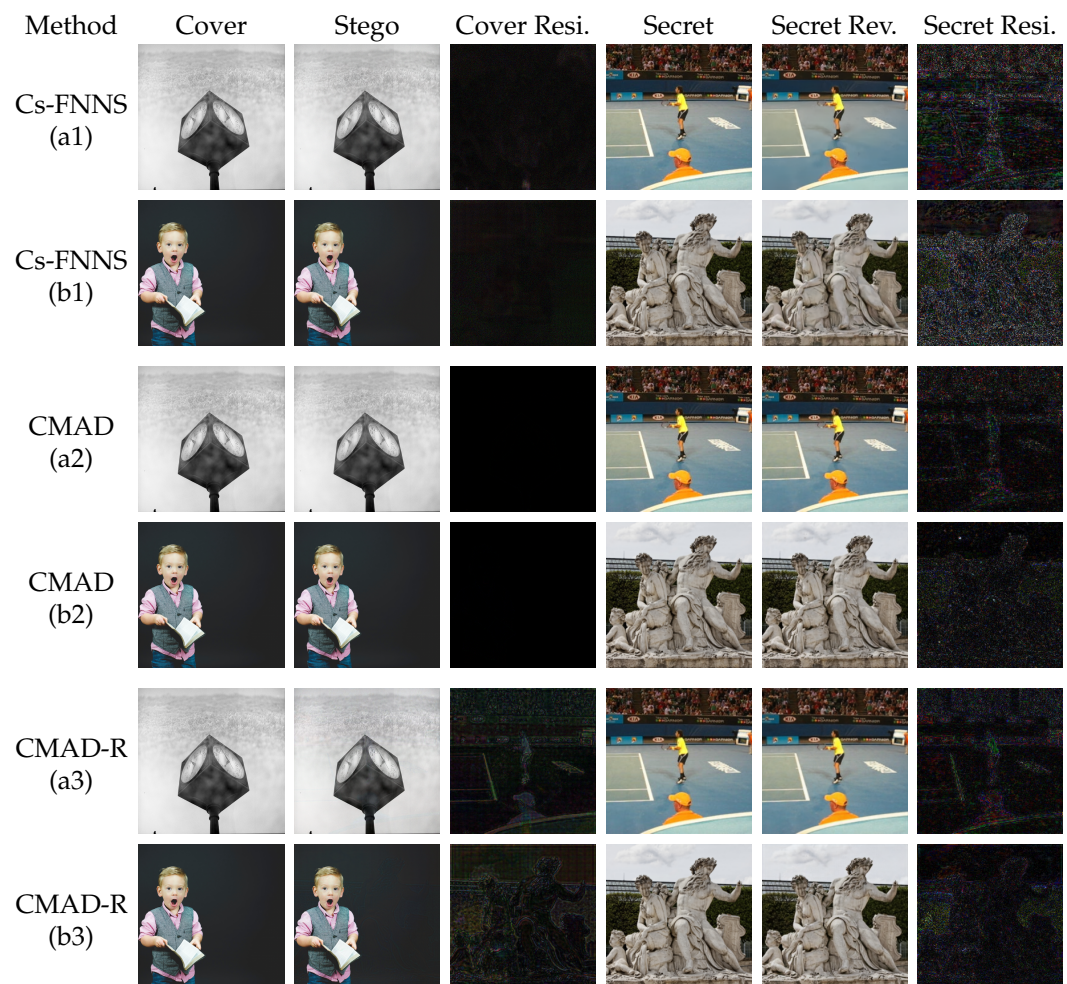


Figure 6. Visual comparison among CMAD, Cs-FNNS, and CMAD-R using samples from (a1–a3) COCO and (b1–b3) DIV2K. Residuals are magnified by a factor of ten.

All steganalysis networks are trained using the Adam optimizer with a weight decay of 1×10^{-5} , an initial learning rate of 2×10^{-4} , and a batch size of 4. The learning rate is halved every 30 epochs, and each network is trained for 100 epochs. For each detector, three instances are trained using random seeds 42, 23, and 54, respectively. During training, the checkpoint with the best validation-set performance is retained as the final model for that seed, and this model is then evaluated once on the independent test set to obtain the detection accuracy for that seed. Finally, the detection accuracies from the three seeds are averaged to produce the reported result for that detector on the corresponding dataset, along with the standard deviation. The results are shown in Table 3.

CMAD's detectability varies across datasets. In the universal setting, CelebA produces accuracies closest to random guessing, whereas COCO and DIV2K are more detectable. On DIV2K, YeNet reaches 63.91%, while SiaStegNet and SRNet reach 60.38% and 58.97%,

respectively. In the specific setting, accuracy rises substantially on every dataset; SRNet reaches 74.15% on DIV2K, and YeNet reaches 75.36% on CelebA. These results describe detection difficulty only under the present system configuration and threat model. CMAD shows limited resistance to detectors not trained specifically on CMAD, but further improvement is needed on COCO and DIV2K, and its overall resistance to a detector trained specifically for CMAD remains limited.

Table 3. Steganalysis detection accuracy for CMAD on different datasets (mean $\pm$ standard deviation). Accuracy closer to the random-guessing level of 50% indicates lower detectability.

Detector	Dataset	YeNet	SRNet	SiaStegNet
Universal	CelebA	53.27 $\pm$ 0.83%	55.93 $\pm$ 0.41%	51.44 $\pm$ 0.80%
	COCO	60.76 $\pm$ 0.58%	57.25 $\pm$ 1.08%	55.31 $\pm$ 0.32%
	DIV2K	63.91 $\pm$ 0.94%	58.97 $\pm$ 0.73%	60.38 $\pm$ 1.09%
Specific	CelebA	75.36 $\pm$ 0.53%	71.87 $\pm$ 1.01%	65.97 $\pm$ 0.93%
	COCO	64.96 $\pm$ 0.73%	63.63 $\pm$ 1.41%	61.63 $\pm$ 0.52%
	DIV2K	72.00 $\pm$ 0.54%	74.15 $\pm$ 0.82%	70.00 $\pm$ 0.36%

Figure 7 shows the results of StegExpose on CMAD stego images. The cover and stego score distributions overlap substantially, so no threshold effectively separates them. The receiver-operating-characteristic (ROC) curve closely follows the diagonal random-guessing reference, with an area under the curve (AUC) of only 0.4791. StegExpose’s conventional statistical features are therefore almost entirely unable to identify CMAD stego images.

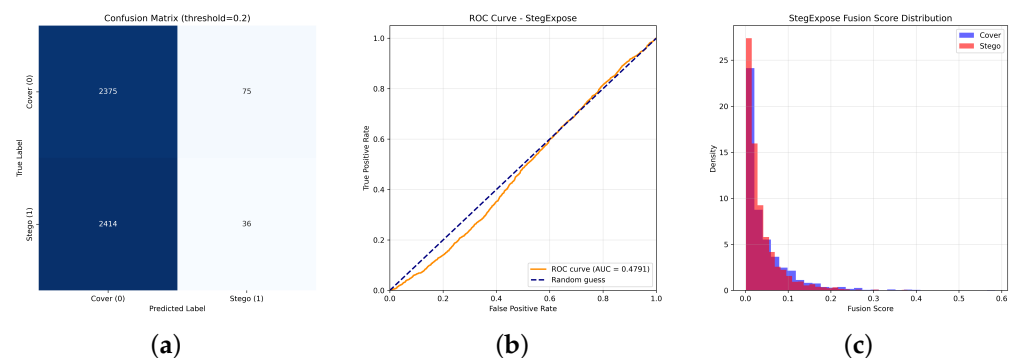


Figure 7. StegExpose detection results for CMAD-generated stego images. (a) Confusion matrix. At the default threshold of 0.2, the detection rate is only 1.47%. (b) ROC curve constructed by varying the threshold over continuous scores; the dashed line is random guessing. (c) Score-distribution histogram. The cover and stego fusion-score distributions and their density curves substantially overlap.

4.4. Robustness Evaluation

CMAD achieves high imperceptibility and recovery quality under lossless transmission, but its recovery performance deteriorates substantially under channel attacks. To improve robustness, we construct CMAD-R. At each optimization iteration, an attack is triggered with a probability of 0.2. When an attack is triggered, one of three training attacks—JPEG compression, Gaussian noise, or contrast adjustment—is selected uniformly at random and applied to the current stego image $C_s = C + \delta$. The decoder then reconstructs the secret image $\hat{S}'$ from the attacked perturbation $\delta' = C'_s - C$. For each attack, the attack-induced distortion residual is detached from the computational graph and added to the original perturbation, allowing gradients to propagate only through the δ path. An attack-recovery loss, $\mathcal{L}_{\text{attack}} = \text{MSE}(\hat{S}', S)$, with a weight of 0.5 is added to the original objective. Attack simulation is used only during optimization. At test time, each attack is applied directly to an unattacked stego image.

The robustness experiments are conducted on DIV2K, using the first 450 images as covers and the remaining 450 as secrets. The cover and secret images are resized to 512×512 and 256×256 , respectively, and all other hyperparameters are identical to those of standard CMAD. Six attacks are evaluated at test time: JPEG compression with a quality factor of $QF = 80$; additive white Gaussian noise with a standard deviation of $\sigma = 0.07$; contrast adjustment with a factor of $\gamma = 0.7$, where $\gamma = 1$ leaves the image unchanged and $\gamma < 1$ reduces the contrast according to $C'_s = \mu + \gamma(C_s - \mu)$, with μ denoting the mean of C_s ; scaling with a factor of $\alpha = 0.95$, implemented by downsampling the image to 95% of its original size and restoring it using bilinear interpolation; Gaussian blur with a standard deviation of $\sigma_b = 0.1$; and random cropping with a retention ratio of $r = 0.9$, followed by bilinear resizing to the original spatial resolution. During training, only JPEG compression, Gaussian noise, and contrast adjustment are used, with weaker settings of $QF = 90$, $\sigma = 0.05$, and $\gamma = 0.8$, respectively. Scaling, Gaussian blur, and cropping are not included during optimization and are used to evaluate generalization to unseen attack types. Each experiment is repeated using multiple random seeds, and the averaged results are reported.

As shown in Table 4, standard CMAD achieves a stego-image PSNR of 55.09 dB in the absence of attacks, but its recovered-secret PSNR decreases substantially under channel attacks, demonstrating its dependence on an accurate residual. CMAD-R reduces the unattacked stego-image PSNR to 40.38 dB but improves the recovered-secret PSNR by 6.02–15.58 dB across the six evaluated attacks. Compared with RFNNS, CMAD-R achieves comparable recovery quality under JPEG compression, scaling, and Gaussian blur, but remains weaker under Gaussian noise, contrast adjustment, and cropping. These results indicate that attack-aware optimization substantially improves robustness, although further improvement is required for stronger noise, contrast distortion, and geometric content loss.

Table 4. Quality of stego and recovered-secret images under different attacks. $\uparrow$ indicates larger is better, $\downarrow$ indicates smaller is better.

Method	Metric	Stego Image							Recovered Secret						
		None	JPEG	Gaussian	Contrast	Scaling	Blur	Cropping	None	JPEG	Gaussian	Contrast	Scaling	Blur	Cropping
Cs-FNNS [13]	PSNR $\uparrow$	42.66	21.35	16.14	26.77	27.58	25.62	26.53	32.85	19.07	18.76	15.64	17.52	25.51	20.08
	SSIM $\uparrow$	0.9766	0.6038	0.4057	0.6824	0.7445	0.6308	0.6912	0.9397	0.6825	0.6723	0.3747	0.5506	0.8298	0.7012
	LPIPS $\downarrow$	0.0020	0.3379	0.3421	0.4482	0.1977	0.1903	0.2103	0.0229	0.3052	0.3635	0.5813	0.4010	0.1387	0.2987
RFNNS [14]	PSNR $\uparrow$	41.96	21.53	22.55	31.67	28.54	27.39	28.04	30.93	20.97	24.09	29.96	17.43	25.86	23.46
	SSIM $\uparrow$	0.9500	0.6032	0.7018	0.8075	0.7924	0.7051	0.7489	0.9332	0.6841	0.6722	0.8931	0.5365	0.8398	0.7488
	LPIPS $\downarrow$	0.0064	0.2005	0.1386	0.0778	0.1713	0.1513	0.1512	0.0258	0.2794	0.2617	0.0686	0.4163	0.1303	0.2521
CMAD	PSNR $\uparrow$	55.09	32.91	23.32	23.39	32.16	40.30	32.48	36.75	10.85	10.89	11.23	12.00	17.26	11.52
	SSIM $\uparrow$	0.9980	0.9349	0.5135	0.9189	0.9467	0.9865	0.9452	0.9731	0.0965	0.0927	0.2110	0.2137	0.5216	0.1523
	LPIPS $\downarrow$	0.0001	0.0248	0.3306	0.0369	0.0791	0.0111	0.0612	0.0063	0.7015	0.7267	0.6878	0.6414	0.4115	0.6487
CMAD-R	PSNR $\uparrow$	40.38	32.37	23.24	23.31	31.89	36.63	31.96	37.43	21.42	18.19	26.81	18.02	26.95	21.92
	SSIM $\uparrow$	0.9671	0.9182	0.5101	0.9037	0.9369	0.9644	0.9315	0.9749	0.6287	0.4147	0.8431	0.5495	0.8392	0.7021
	LPIPS $\downarrow$	0.0200	0.0490	0.3356	0.0474	0.0865	0.0276	0.0712	0.0041	0.2883	0.4783	0.0921	0.4043	0.1221	0.3518

Attack parameters: JPEG $QF=80$, Gaussian $\sigma = 0.07$, contrast $\gamma = 0.7$, scaling $\alpha = 0.95$, blur $\sigma = 0.1$, cropping (random).

4.5. Ablation Studies

4.5.1. Effectiveness of Decoder Fine-Tuning and Channel Attention

Three variants evaluate the contribution of decoder fine-tuning and channel attention: a static decoder (Static Dec.), no channel attention (No CA), and both mechanisms disabled (Static Dec. & No CA). Table 5 reports the results.

Disabling decoder fine-tuning reduces stego-image PSNR from 55.46 to 35.82 dB and recovered-secret PSNR from 37.04 to 27.45 dB; the corresponding SSIM values fall to 0.7736 and 0.8074. A static decoder therefore cannot adapt to the current image pair and forces the optimizer to increase the perturbation to compensate for insufficient decoding capacity. Removing channel attention reduces stego-image PSNR further to 30.75 dB, while recovered-secret PSNR is 30.12 dB, confirming the importance of channel attention for

feature selection. When both mechanisms are disabled, recovered-secret PSNR falls to 12.20 dB and SSIM to 0.2287; decoding almost entirely fails. The degradation is greater than when either mechanism is removed alone, showing that decoder adaptation and channel attention are mutually dependent.

Table 5. Image-quality comparison among decoder variants. $\uparrow$ indicates larger is better, $\downarrow$ indicates smaller is better.

Variant	Stego Image				Recovered Secret			
	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	MAE $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	MAE $\downarrow$
Baseline	55.46 $\pm$ 2.69	0.9987 $\pm$ 0.0018	0.0001 $\pm$ 0.0001	0.20 $\pm$ 0.10	37.04 $\pm$ 3.06	0.9701 $\pm$ 0.0073	0.0073 $\pm$ 0.0050	2.51 $\pm$ 0.93
Static Dec.	35.82 $\pm$ 3.07	0.7736 $\pm$ 0.0483	0.1863 $\pm$ 0.0361	4.52 $\pm$ 0.79	27.45 $\pm$ 2.31	0.8074 $\pm$ 0.0712	0.1214 $\pm$ 0.0277	13.80 $\pm$ 1.72
No CA	30.75 $\pm$ 2.23	0.8389 $\pm$ 0.0417	0.1378 $\pm$ 0.0289	5.38 $\pm$ 1.12	30.12 $\pm$ 2.53	0.9067 $\pm$ 0.0563	0.1096 $\pm$ 0.0239	4.95 $\pm$ 0.94
Static Dec. & No CA	24.35 $\pm$ 3.41	0.5423 $\pm$ 0.0819	0.4052 $\pm$ 0.0436	12.85 $\pm$ 2.33	12.20 $\pm$ 1.92	0.2287 $\pm$ 0.0793	0.6125 $\pm$ 0.0684	55.80 $\pm$ 8.67

4.5.2. Effectiveness of AniCost Guidance

AniCost is evaluated on the first 200 DIV2K images, with the first 100 serving as covers and the remaining 100 as secrets. Every experiment is repeated with multiple random seeds, and the mean and standard deviation are reported. The comparison priors are No Prior ($\lambda_P = 0$), Uniform Prior ($P \equiv 1$), Random Prior ($P \sim \mathcal{U}(0, 1)$), Inverse AniCost ($P = 1 - P_{\text{AniCost}}$), the WOW cost map, the S-UNIWARD cost map, and AniCost. Table 6 reports stego-image and recovered-secret quality for each prior.

Table 6. Image-quality comparison under different cost-guidance priors. $\uparrow$ indicates larger is better, $\downarrow$ indicates smaller is better.

Prior	Stego Image				Recovered Secret			
	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	MAE $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	MAE $\downarrow$
AniCost (baseline)	55.46 $\pm$ 2.69	0.9987 $\pm$ 0.0018	0.0001 $\pm$ 0.0001	0.20 $\pm$ 0.10	37.04 $\pm$ 3.06	0.9701 $\pm$ 0.0073	0.0073 $\pm$ 0.0050	2.51 $\pm$ 0.93
WOW	52.31 $\pm$ 2.41	0.9985 $\pm$ 0.0018	0.0001 $\pm$ 0.0001	0.22 $\pm$ 0.12	35.29 $\pm$ 3.01	0.9547 $\pm$ 0.0311	0.0065 $\pm$ 0.0043	3.20 $\pm$ 1.27
S-UNIWARD	53.19 $\pm$ 2.60	0.9986 $\pm$ 0.0015	0.0001 $\pm$ 0.0001	0.22 $\pm$ 0.10	35.70 $\pm$ 3.16	0.9632 $\pm$ 0.0254	0.0063 $\pm$ 0.0052	2.95 $\pm$ 1.06
No Prior	37.27 $\pm$ 1.33	0.9702 $\pm$ 0.0151	0.0036 $\pm$ 0.0021	2.68 $\pm$ 0.51	39.24 $\pm$ 5.05	0.9796 $\pm$ 0.0266	0.0041 $\pm$ 0.0055	1.83 $\pm$ 1.15
Uniform Prior	38.09 $\pm$ 1.34	0.9716 $\pm$ 0.0149	0.0032 $\pm$ 0.0019	2.53 $\pm$ 0.51	38.85 $\pm$ 5.03	0.9763 $\pm$ 0.0266	0.0054 $\pm$ 0.0056	1.76 $\pm$ 1.14
Random Prior	51.34 $\pm$ 0.82	0.9974 $\pm$ 0.0020	0.0004 $\pm$ 0.0010	0.17 $\pm$ 0.03	35.45 $\pm$ 3.10	0.9613 $\pm$ 0.0254	0.0064 $\pm$ 0.0049	3.05 $\pm$ 1.09
Inverse AniCost	57.41 $\pm$ 2.82	0.9987 $\pm$ 0.0011	0.0002 $\pm$ 0.0002	0.08 $\pm$ 0.03	33.70 $\pm$ 2.48	0.9477 $\pm$ 0.0246	0.0091 $\pm$ 0.0053	3.74 $\pm$ 1.12

No Prior and Uniform Prior produce similar results. Without content-adaptive cost guidance, the optimizer tends to reduce the decoding loss by increasing the perturbation globally, sharply lowering stego-image PSNR and seriously compromising imperceptibility. Although recovered-secret PSNR is relatively high, it is obtained at the cost of concealment. AniCost concentrates perturbation energy in textured regions, achieving a stego-image PSNR of 55.46 dB while retaining a recovered-secret PSNR of 37.04 dB and thus providing the best trade-off between imperceptibility and recovery. Compared with AniCost, WOW and S-UNIWARD have stego-image PSNR values that are 3.15 and 2.27 dB lower and recovered-secret PSNR values that are 1.75 and 1.34 dB lower, respectively. AniCost provides a more favorable empirical trade-off under the evaluated configuration. Random Prior lacks content adaptivity and reaches only 35.45 dB for recovered-secret PSNR. Inverse AniCost directs perturbations toward flat regions and reduces recovered-secret PSNR further to 33.70 dB, confirming that the direction of the cost prior influences embedding locations.

4.5.3. Information-Carrier Validation and Security Analysis

Controlled experiments are conducted on 100 DIV2K pairs by reusing generated cover–stego pairs and their corresponding fine-tuned weights. The tested conditions include zero perturbation, random perturbation, swapped decoders, fixed inputs, averaged

weights, randomly initialized decoders, cover estimation and mismatch, and training driven by fixed noise. DnCNN post-processing and 8-bit quantization are used throughout. The combination of the correct perturbation and correct fine-tuned weights serves as the baseline. Table 7 summarizes the configurations and recovery quality.

Table 7. Configurations and recovery quality in the information-carrier validation experiment.

Case	Perturbation/Input	Decoder Weights	Cover Condition	PSNR (dB)
Baseline	Correct perturbation	Correct weights	Correct cover	37.04 ± 3.06
Zero perturbation	All zeros	Correct weights	Correct cover	11.50 ± 1.40
Random perturbation	Equal-variance Gaussian noise	Correct weights	Correct cover	10.03 ± 1.32
Swapped decoder	Correct perturbation	Weights from another image	Correct cover	17.69 ± 4.95
Fixed input (zero)	All zeros	Correct weights	Correct cover	11.50 ± 1.40
Fixed input (0.5)	Constant 0.5	Correct weights	Correct cover	11.16 ± 1.39
Fixed input (random)	Random tensor	Correct weights	Correct cover	10.30 ± 1.33
Averaged weights	Correct perturbation	Averaged weights	Correct cover	20.28 ± 3.19
Random decoder	Correct perturbation	Random initialization	Correct cover	11.30 ± 1.37
Cover estimation ($\sigma = 1.0$)	Extracted from estimated cover	Correct weights	Estimated cover	10.76 ± 1.48
Cover mismatch	Extracted from mismatched cover	Correct weights	Mismatched cover	10.73 ± 1.48
Fixed-noise training	Fixed random noise	Noise-specific weights	Correct cover	10.78 ± 1.32

The experimental results show that the recovered-secret PSNR remains below 12 dB under zero perturbation, random perturbation, and all fixed-input conditions. Decoder swapping and weight averaging also yield recovered-secret PSNR values of only 17.69 and 20.28 dB, respectively, far below the baseline of 37.04 dB. These findings indicate that a valid perturbation and the matching weights are both indispensable for secret recovery. Cover-estimation and cover-mismatch attacks both reduce the recovered-secret PSNR to approximately 10.7 dB, further demonstrating that decoding is highly dependent on the exact cover. The cover-reuse experiment in Appendix E shows that even when an attacker averages multiple stego images to obtain an estimated cover with a PSNR as high as 65.5 dB, the recovered-secret PSNR remains only approximately 18 dB, indicating that a high-precision estimate cannot substitute for the exact cover. Once the exact cover is compromised, however, an attacker can directly extract the exact perturbation from the stego image and combine it with the intercepted fine-tuned weights to raise the recovered-secret PSNR to the baseline level of approximately 37 dB, thereby fully exposing the secret. Therefore, under the evaluated averaging attack, for the tested cover, effective recovery was not achieved with up to 100 reused messages. The protocol permits cover reuse, but its core security assumption is that the exact cover must remain strictly confidential.

In the control experiment with fixed-noise-driven fine-tuning, the decoder receives the same fixed random noise during training and testing, and the perturbation does not participate in decoding. The recovered-secret PSNR is 10.78 dB. Although the weights exhibit some memorization capability, they do not enable effective recovery. This result shows that successful recovery depends jointly on the image perturbation and the corresponding message-specific decoder weights.

4.5.4. Effect of the Dynamic Weight-Scheduling Strategy

Five schedules evaluate the adaptive dynamic strategy in Equations (19) and (20): (1) Adaptive, the proposed threshold-based multiplicative-growth strategy with upper-bound clipping; (2) Const-Small, which fixes the initial weights $\lambda_{\text{dec}} = 1.5$ and $\lambda_{\text{hid}} = 0.4$; (3) Const-Large, which fixes manually tuned values $\lambda_{\text{dec}} = 150$ and $\lambda_{\text{hid}} = 40$; (4) Stepwise, which applies $\lambda_{\text{dec}} \leftarrow 2.0\lambda_{\text{dec}}$ and $\lambda_{\text{hid}} \leftarrow 1.5\lambda_{\text{hid}}$ every 200 iterations with an upper bound of 10^4 ; and (5) Reciprocal, which sets $\lambda_{\text{dec}} = 1/(\mathcal{L}_{\text{dec}} + \varepsilon)$ and $\lambda_{\text{hid}} = 1/(\mathcal{L}_{\text{hid}} + \varepsilon)$ at every iteration, where $\varepsilon = 10^{-8}$. Experiments use the first 200 DIV2K images, divided equally

into covers and secrets, and are repeated with multiple random seeds. Table 8 reports the mean and standard deviation.

The schedules exhibit a clear trade-off between stego imperceptibility and secret recovery. Adaptive gives the best overall stego-image quality. Its recovered-secret PSNR is 37.04 dB, less than 3 dB below Reciprocal, Stepwise, and Const-Large. Const-Small retains reasonable stego quality but produces the lowest recovered-secret PSNR because its small static weights provide insufficient decoding pressure. Const-Large and Stepwise improve recovery but reduce stego-image PSNR by more than 14 dB relative to Adaptive. Reciprocal achieves the highest recovered-secret PSNR, but its stego-image PSNR is only 36.71 dB; moreover, its stego and recovered-secret PSNR standard deviations are 2.95 and 5.26 dB, respectively. The inverse-loss weights therefore fluctuate strongly and destabilize optimization. Overall, Adaptive provides the best balance and demonstrates the importance of adaptive multiplicative growth to joint mask-perturbation optimization.

Table 8. Image quality under different weight-scheduling strategies on DIV2K and COCO (mean $\pm$ standard deviation). $\uparrow$ indicates larger is better, $\downarrow$ indicates smaller is better.

Dataset	Schedule	Stego Image				Recovered Secret			
		PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	MAE $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	MAE $\downarrow$
DIV2K	Adaptive	55.46 $\pm$ 2.69	0.9987 $\pm$ 0.0018	0.0001 $\pm$ 0.0001	0.20 $\pm$ 0.10	37.04 $\pm$ 3.06	0.9701 $\pm$ 0.0073	0.0073 $\pm$ 0.0050	2.51 $\pm$ 0.93
	Const-Small	49.89 $\pm$ 2.83	0.9958 $\pm$ 0.0050	0.0004 $\pm$ 0.0009	0.32 $\pm$ 0.15	35.63 $\pm$ 2.90	0.9597 $\pm$ 0.0081	0.0081 $\pm$ 0.0050	3.04 $\pm$ 1.02
	Const-Large	39.50 $\pm$ 2.06	0.9747 $\pm$ 0.0240	0.0024 $\pm$ 0.0019	1.84 $\pm$ 0.61	39.11 $\pm$ 4.42	0.9792 $\pm$ 0.0043	0.0043 $\pm$ 0.0050	1.96 $\pm$ 1.08
	Stepwise	41.91 $\pm$ 1.66	0.9810 $\pm$ 0.0171	0.0025 $\pm$ 0.0042	1.16 $\pm$ 0.26	39.89 $\pm$ 4.97	0.9816 $\pm$ 0.0037	0.0037 $\pm$ 0.0051	1.76 $\pm$ 1.12
	Reciprocal	36.71 $\pm$ 2.95	0.9498 $\pm$ 0.0411	0.0069 $\pm$ 0.0071	2.97 $\pm$ 1.16	40.33 $\pm$ 5.26	0.9827 $\pm$ 0.0035	0.0035 $\pm$ 0.0051	1.64 $\pm$ 1.15
COCO	Adaptive	55.57 $\pm$ 3.73	0.9976 $\pm$ 0.0001	0.0001 $\pm$ 0.0001	0.21 $\pm$ 0.16	34.93 $\pm$ 2.62	0.9520 $\pm$ 0.0150	0.0150 $\pm$ 0.0062	3.20 $\pm$ 0.90
	Const-Small	51.23 $\pm$ 2.63	0.9961 $\pm$ 0.0003	0.0003 $\pm$ 0.0004	0.25 $\pm$ 0.11	34.24 $\pm$ 2.65	0.9409 $\pm$ 0.0148	0.0148 $\pm$ 0.0063	3.55 $\pm$ 1.00
	Const-Large	39.63 $\pm$ 2.72	0.9654 $\pm$ 0.0024	0.0024 $\pm$ 0.0021	1.85 $\pm$ 0.75	36.38 $\pm$ 3.73	0.9668 $\pm$ 0.0113	0.0113 $\pm$ 0.0067	2.68 $\pm$ 1.06
	Stepwise	42.58 $\pm$ 1.92	0.9766 $\pm$ 0.0023	0.0023 $\pm$ 0.0026	1.03 $\pm$ 0.29	36.67 $\pm$ 3.95	0.9690 $\pm$ 0.0110	0.0110 $\pm$ 0.0069	2.57 $\pm$ 1.08
	Reciprocal	37.68 $\pm$ 5.05	0.9397 $\pm$ 0.0057	0.0057 $\pm$ 0.0063	2.91 $\pm$ 1.59	36.82 $\pm$ 4.07	0.9702 $\pm$ 0.0108	0.0108 $\pm$ 0.0069	2.52 $\pm$ 1.09

4.6. Communication Overhead and Computational Efficiency

The overhead of the complete CMAD protocol comprises pre-shared information and per-message transmission. Before communication, the sender and receiver pre-share the structural definition of D and the original cover C . The network description is approximately 1.86 KB, whereas the cover size depends on the image. For each message, the sender transmits C_s and θ_D through the public channel. The stego-image size is approximately the same as that of the cover, and the weight file is 411 KB.

For every cover-secret pair, the sender performs 2000 joint-optimization iterations, whereas the receiver performs only decoding, which takes approximately 2 s. The sender requires approximately 27 s per pair for CMAD, 12 s for C_s -FNNS, and 24 s for RFNNS. CMAD is slower because joint mask-perturbation optimization and decoder fine-tuning require additional backpropagation. Peak memory use is approximately 1809.3 MB for CMAD, 1154.3 MB for C_s -FNNS, and 1906.1 MB for RFNNS. CMAD therefore lies between the two comparison methods and fits within the memory capacity of the experimental GPU. These measurements use one NVIDIA RTX 5060 GPU with 8 GB of memory, 512×512 cover images, 256×256 secret images, and raw secret-image source rate of 6 bpp.

5. Conclusions

This paper proposed CMAD, an adaptive steganographic method based on cost-guided joint mask-perturbation optimization. CMAD jointly optimizes the embedding mask and perturbation magnitude in an end-to-end differentiable framework and combines AniCost guidance with a channel-attention decoding network. On COCO, CelebA, and DIV2K, its stego images achieve PSNR values of 55–60 dB, and its recovered secret images achieve PSNR values of 35–40 dB. CMAD approaches random-guessing performance against conventional statistical steganalysis, but its resistance to deep-learning-based

steganalyzers requires further improvement. CMAD-R retains some recovery capability under JPEG compression, contrast adjustment, and other attacks, providing a lossy-channel variant of CMAD. Ablation results identify AniCost guidance as essential to balancing imperceptibility and recovery quality. Information-carrier validation further shows that successful recovery depends jointly on the image perturbation and the corresponding message-specific decoder weights. The mask, cost guidance, and attention-based fine-tuning are all necessary within the joint-optimization framework.

When only the stego images are considered, CMAD approaches random-guessing performance under conventional statistical steganalysis, but its resistance to deep-learning-based steganalyzers still requires further improvement. In addition, the complete CMAD protocol also transmits fine-tuned decoder weights, which may reveal the use of CMAD. The security claims in this paper are therefore restricted to image-level steganographic detectability, while protocol-level covertness remains an open challenge for future work. The robustness of CMAD-R against attacks such as Gaussian noise also remains to be improved.

Author Contributions: Conceptualization, Q.L. and X.C.; Methodology, Q.L.; Software, X.C.; Validation, Q.L., X.C., H.W., Z.W. and M.L.; Formal Analysis, Q.L. and H.W.; Investigation, Q.L., Z.W. and M.L.; Resources, Q.L.; Data Curation, X.C., H.W. and M.L.; Writing—Original Draft Preparation, X.C.; Writing—Review and Editing, Q.L., X.C., H.W., Z.W. and M.L.; Visualization, Z.W.; Project Administration, Q.L.; Funding Acquisition, Q.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Key Research Project of Gansu University of Political Science and Law under Grant No.GZF2025XZD19.

Data Availability Statement: The data used in this study are publicly available datasets. COCO is available at <https://cocodataset.org>; CelebA is available at <https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html>; DIV2K is available at <https://data.vision.ee.ethz.ch/cv1/DIV2K/>. All datasets were accessed on 13 December 2025.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Communication Overhead and Capacity Analysis

Appendix A.1. Communication Overhead

The primary communication-overhead difference among CMAD, Cs-FNNS, and RFNNS lies in decoder-weight transmission. Their pre-sharing overheads are of comparable magnitude: the Cs-FNNS decoder architecture requires approximately 4.73 KB, the RFNNS architecture 6.05 KB, and the CMAD architecture 1.86 KB. These are one-time costs that are much smaller than the transmission required for each message. Cs-FNNS and RFNNS generate their covers from prompts and therefore require the sender and receiver to share a prompt in addition to the decoding network. CMAD can use any image as its cover. However, the cover must be treated as secret. The sender and receiver must pre-share the cover through a confidential channel. Merely downloading the same image from a public source is not secure, because an adversary could also access that source and reconstruct the cover.

Per-message transmission constitutes the main difference. Assuming that a 512×512 RGB image occupies approximately 768 KB, Cs-FNNS and RFNNS initialize their decoder weights from a shared seed and do not transmit a weight file; each message therefore contains only an approximately 768 KB stego image. CMAD fine-tunes its decoder for each image pair and must transmit an additional 411 KB of weights, for a total of approximately 1179 KB per message. Under an equal total budget of 1179 KB, Cs-FNNS and RFNNS can allocate the full budget to the stego image, whereas CMAD allocates 411 KB to the weights

and approximately 768 KB to the stego image. All three methods remain within the same order of magnitude, from several hundred kilobytes to approximately 1 MB per message. CMAD exchanges an additional 411 KB for the image-quality improvements reported in Table 2.

Appendix A.2. Estimated Information in Residual and Weight Increment

We separately estimate the effective information carried by the image residual and the transmitted fine-tuned weight increment. The original secret contains 192 KB (1.57×10^6 bits). Nonzero elements of $\delta = C_s - C$ are discretized into 256 equal-width bins, yielding a total residual entropy of 2.08×10^5 bits, or approximately 25 KB of effective information. Applying the same procedure to the weight increment $\Delta\theta$ yields 3.18×10^5 bits, or approximately 40 KB.

Appendix A.3. Comparison with Cryptography

For a 512×512 cover and a 256×256 secret, the original secret occupies approximately 192 KB. CMAD transmits an approximately 768 KB stego image and 411 KB of decoder weights, for a total of approximately 1179 KB and a transmission ratio of about 6.1:1; approximately 6.1 transmitted bits are required per useful recovered bit. Cs-FNNS and RFNNS transmit only an approximately 768 KB stego image, producing a ratio of approximately 4:1. In bit efficiency alone, CMAD is therefore less efficient because it transmits decoder weights. In multi-message settings, the 1–6 KB decoder-architecture descriptions are one-time pre-sharing costs and can be amortized, as can a pre-shared cover if it is reused by a sender. CMAD's 411 KB weight file, however, is specific to each message and cannot be amortized. Its overhead accumulates linearly over a long sequence of messages, which is an inherent cost of per-image decoder fine-tuning.

If bit efficiency is the only criterion, encrypting the secret with a standard cipher such as AES-256 produces ciphertext approximately equal in size to the plaintext and a transmission ratio of about 1:1, substantially better than all three steganographic methods. This comparison highlights the fundamental distinction between steganography and cryptography. Encryption protects message content but does not conceal the existence of communication. The ciphertext generated by direct encryption appears as a random bitstream; its detectability depends on the specific protocol and traffic context, and in some cases it may be identified as encrypted communication and raise suspicion. Steganography hides secret information in an apparently ordinary image and seeks to make the stego image visually and statistically indistinguishable from a normal image, concealing the act of communication itself. The two technologies therefore solve different problems and serve different use cases.

Appendix B. CMAD Performance Statistics and Robustness

Unless otherwise stated, the appendix experiments are conducted on the first 200 images of the DIV2K dataset, with the first 100 serving as covers and the remaining 100 as secrets. Cover and secret image sizes are set to 512×512 and 256×256 , respectively. Each experiment is repeated with random seeds 42, 23, and 54, and the mean and standard deviation are computed over all image-pair observations across the three runs.

Table A1. Mean, standard deviation, and variance of CMAD quality metrics on different datasets. The dataset partitioning is the same as in 4.2. ↑ indicates larger is better, ↓ indicates smaller is better.

Dataset	Statistic	Stego Image				Recovered Secret			
		PSNR ↑	SSIM ↑	LPIPS ↓	MAE ↓	PSNR ↑	SSIM ↑	LPIPS ↓	MAE ↓
COCO	Mean	55.31	0.9990	0.0001	0.22	35.98	0.9653	0.0061	2.81
	SD	2.99	0.0032	0.0003	0.16	2.91	0.0218	0.0051	0.95
	Variance	8.9627	0.000010	0	0.0246	8.4904	0.000476	0.000026	0.9116
CelebA	Mean	59.72	0.9992	0.0001	0.08	40.35	0.9743	0.0069	1.80
	SD	1.96	0.0057	0.0014	0.21	0.60	0.0080	0.0028	0.13
	Variance	3.8237	0.000033	0.000002	0.0435	0.3565	0.000064	0.000008	0.0166
DIV2K	Mean	55.09	0.9980	0.0001	0.22	36.75	0.9731	0.0063	2.62
	SD	3.14	0.0042	0.0003	0.18	2.72	0.0176	0.0051	0.84
	Variance	9.8737	0.000018	0	0.0341	7.4039	0.000310	0.000027	0.7103

Table A2. CMAD-R image quality under compound attacks. ↑ indicates larger is better, ↓ indicates smaller is better.

Attack	Stego Image				Recovered Secret			
	PSNR ↑	SSIM ↑	LPIPS ↓	MAE ↓	PSNR ↑	SSIM ↑	LPIPS ↓	MAE ↓
None	41.63	0.9968	0.0006	0.27	37.32	0.9758	0.0073	2.47
JPEG + Gaussian	22.82	0.4872	0.3415	14.61	10.30	0.0572	0.7185	65.07
JPEG + Contrast	22.97	0.8716	0.0585	15.25	10.81	0.1679	0.7082	61.68
Gaussian + Contrast	21.45	0.5598	0.3155	17.72	11.00	0.1376	0.7240	60.07
JPEG + Gaussian + Contrast	21.17	0.5328	0.3267	18.26	10.74	0.1146	0.7285	61.98

Table A3. Paired comparisons between CMAD and the comparison methods on identical cover–secret image pairs. For each cover–secret image pair, the metric for each method (CMAD, Cs-FNNS, and RFNNS) is first averaged over the three random-seed runs to obtain a representative value for that image pair. Paired differences are then computed as CMAD minus the corresponding comparison method using these averaged values. SD denotes the standard deviation of the paired differences across independent image pairs. The 95% confidence intervals are computed using the t-distribution, and the reported p-values are obtained from two-sided paired-sample t-tests with the image pair as the statistical unit. Sample sizes: COCO n = 1000, CelebA n = 1000, DIV2K n = 450.

Comparison	Metric	Dataset	Mean Difference (dB)	SD (dB)	95% CI Lower	95% CI Upper	p-Value
CMAD vs. Cs-FNNS	Stego PSNR	COCO	13.07	3.21	12.87	13.27	<0.001
		CelebA	17.83	2.50	17.68	17.99	<0.001
		DIV2K	12.44	3.41	12.12	12.75	<0.001
	Recovered-Secret PSNR	COCO	4.01	1.78	3.90	4.12	<0.001
		CelebA	5.12	2.41	4.97	5.27	<0.001
		DIV2K	3.90	1.44	3.76	4.03	<0.001
CMAD vs. RFNNS	Stego PSNR	COCO	12.10	3.14	11.91	12.30	<0.001
		CelebA	15.89	2.50	15.74	16.05	<0.001
		DIV2K	13.13	3.27	12.83	13.43	<0.001
	Recovered-Secret PSNR	COCO	5.93	3.77	5.70	6.16	<0.001
		CelebA	7.19	2.41	7.04	7.34	<0.001
		DIV2K	5.82	3.47	5.50	6.14	<0.001

Note: Mean differences are computed as CMAD minus the corresponding comparison method. SD denotes the standard deviation of the paired differences. The confidence intervals are 95% confidence intervals for the mean paired differences, and the reported p-values are obtained from two-sided paired-sample t-tests.

Table A4. Mean, standard deviation, and variance of recovered-secret quality before and after DnCNN denoising. The dataset partitioning is the same as in 4.2. $\uparrow$ indicates larger is better, $\downarrow$ indicates smaller is better.

Dataset	Statistic	After Denoising				Before Denoising			
		PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	MAE $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	MAE $\downarrow$
COCO	Mean	35.98	0.9653	0.0061	2.81	33.68	0.9606	0.0095	3.35
	SD	2.91	0.0218	0.0051	0.95	3.20	0.0254	0.0065	1.10
	Variance	8.4904	0.000476	0.000026	0.9116	10.24	0.000625	0.000042	1.21
CelebA	Mean	40.35	0.9743	0.0069	1.80	38.85	0.9712	0.0090	2.05
	SD	0.60	0.0080	0.0028	0.13	0.65	0.0093	0.0035	0.15
	Variance	0.3565	0.000064	0.000008	0.0166	0.4225	0.000081	0.000012	0.0225
DIV2K	Mean	36.75	0.9731	0.0063	2.62	34.98	0.9690	0.0088	3.10
	SD	2.72	0.0176	0.0051	0.84	2.95	0.0192	0.0060	0.95
	Variance	7.4039	0.000310	0.000027	0.7103	8.7025	0.000361	0.000036	0.9025

Appendix C. Embedding Distribution and Optimization Convergence

Table A5 shows that 84.24% of pixels are activated, while the mean mask value is only 0.0298 and mask entropy is only 0.115. Perturbation energy in textured regions is approximately 4.2 times that in smooth regions. Most pixels therefore receive only extremely small modifications, and the spatial distribution is biased toward complex textures. The final gradient norms are on the order of 10^{-4} , indicating that optimization is close to convergence. Figure A1 shows the dynamic trajectories of λ_{dec} and λ_{hid} together with the gradient trajectories. Figure A2 shows the cost, probability, mask, and perturbation maps.

Table A5. Embedding spatial-distribution and optimization-convergence metrics (mean $\pm$ SD) over 100 DIV2K cover–secret pairs. Smooth and textured regions are separated at the median AniCost ρ . Gradient norms are the ℓ_2 norms of the final-iteration gradients with respect to ω_δ and ω_M .

Metric	Mean $\pm$ SD	Metric	Mean $\pm$ SD
Activated-pixel ratio	0.8424 ± 0.0563	Mask entropy	0.1148 ± 0.0236
Secret-image differential entropy (bits)	3.9022 ± 0.8449	Final λ_{dec}	7.62 ± 4.85
Residual total entropy (bits)	$(2.08 \pm 1.89) \times 10^5$	Final λ_{hid}	21.75 ± 0.00
Perturbation energy in smooth regions	$(1.20 \pm 1.27) \times 10^{-5}$	Final decoding loss $\mathcal{L}_{\text{dec}}$	$(8.02 \pm 0.99) \times 10^{-5}$
Perturbation energy in textured regions	$(4.99 \pm 3.54) \times 10^{-5}$	Final concealment loss $\mathcal{L}_{\text{hid}}$	$(1.02 \pm 0.80) \times 10^{-5}$
Spearman correlation: P vs. effective modifications	0.5708 ± 0.1080	Gradient norm (δ)	$(2.40 \pm 1.14) \times 10^{-4}$
Spearman correlation: M vs. effective modifications	0.6216 ± 0.1338	Gradient norm (M)	$(6.77 \pm 4.27) \times 10^{-5}$

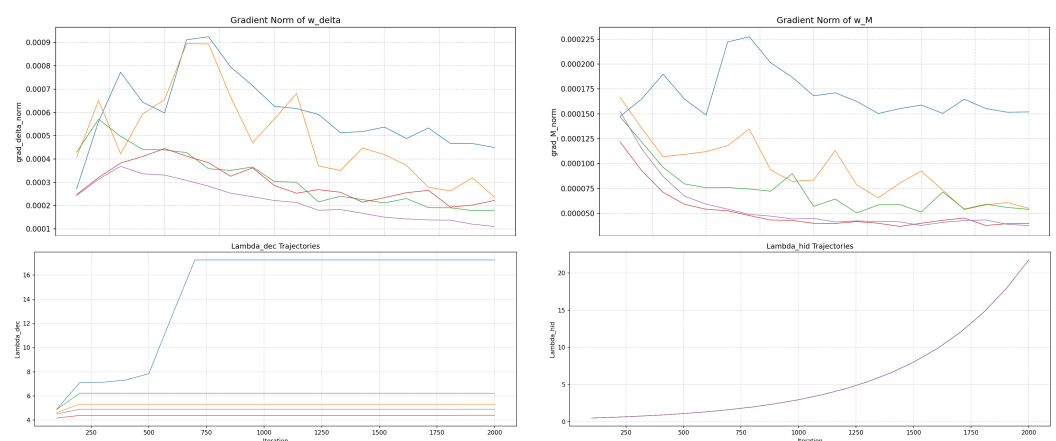


Figure A1. Dynamic trajectories of λ_{hid} and λ_{dec} and the corresponding gradient trajectories. Different colors represent different groups of images.

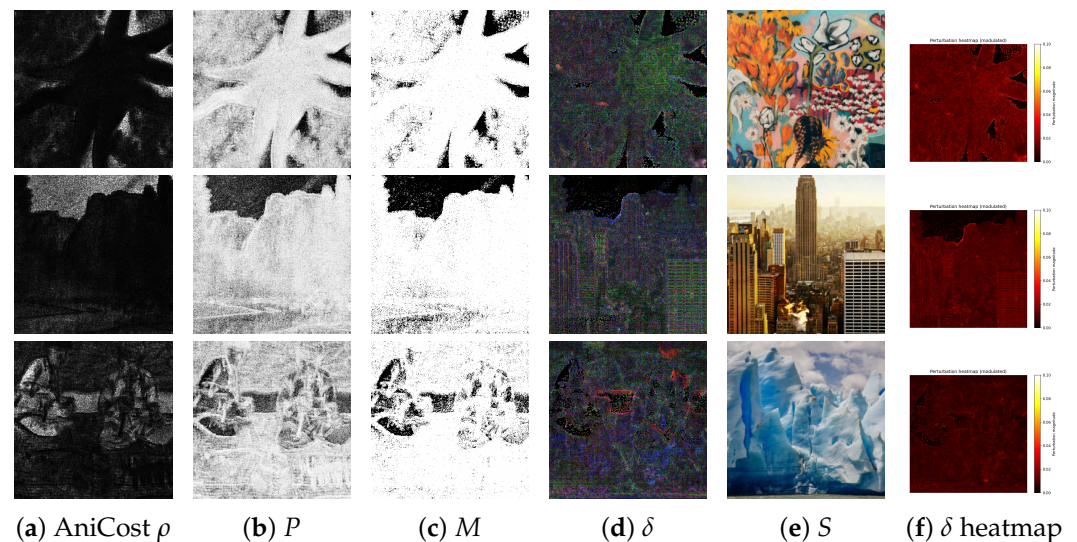


Figure A2. Spatial maps produced during CMAD optimization.

Appendix D. Controlled Synthetic Examples

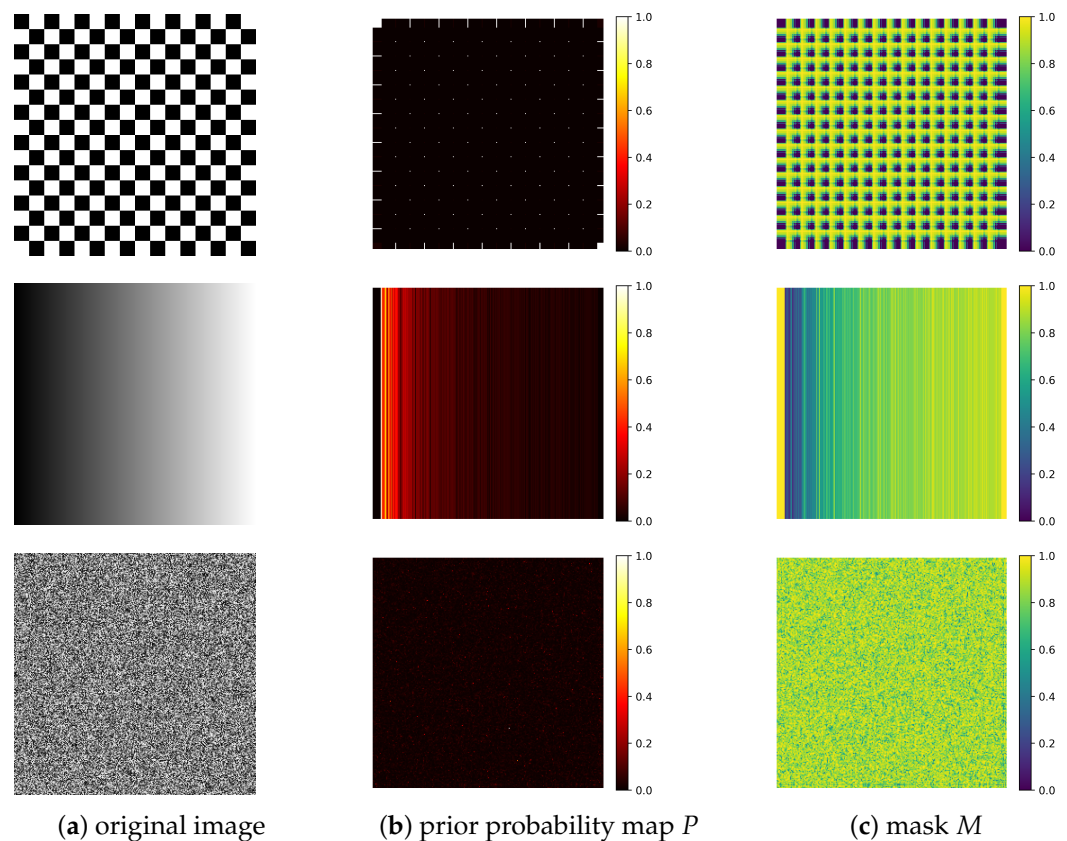


Figure A3. Controlled synthetic examples. Within each row, the images from left to right are the original image, prior probability map P , and mask M .

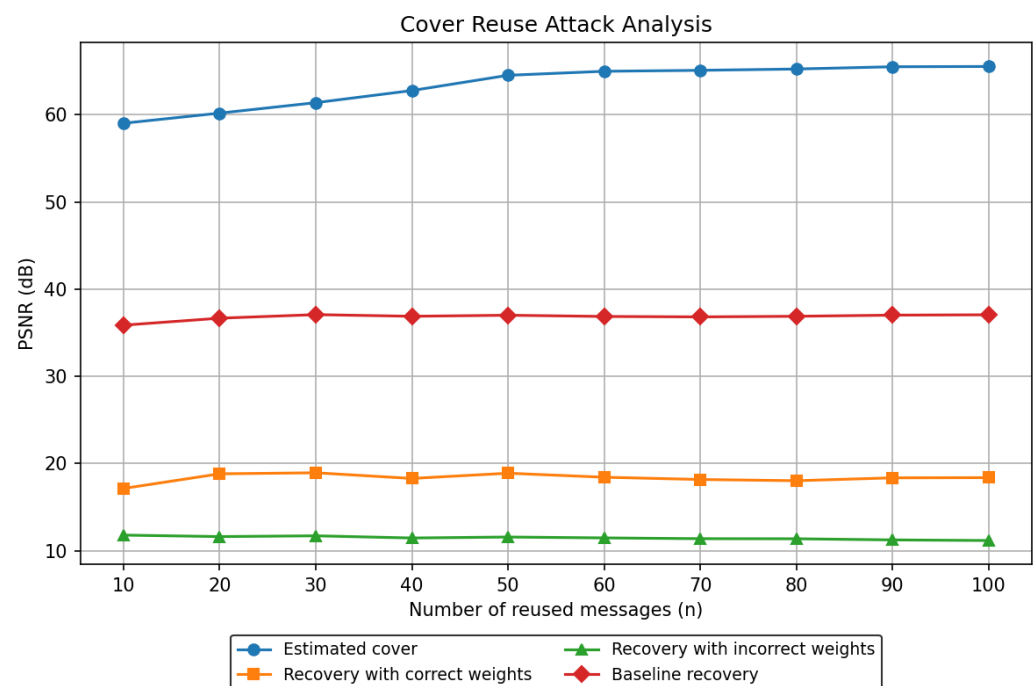
Appendix E. Cover Reuse

One DIV2K cover is fixed, and the attacker is assumed to intercept all stego images and their corresponding fine-tuned weights while not knowing the original cover. The attacker calculates the pixel-wise average of the first n stego images as the estimated cover, extracts the perturbations using this estimated cover, and recovers the secrets using either the correct fine-tuned weights or decoder weights obtained by weight averaging. Table A6 reports the results at representative values of n .

Table A6. Results of the cover-reuse attack.

n	Estimated-Cover PSNR (dB)	Recovery PSNR with Correct Weights (dB)	Recovery PSNR with Averaged Weights (dB)	Baseline Recovery PSNR (dB)
20	60.18	18.81 ± 2.27	11.61 ± 1.92	36.66 ± 3.18
40	62.77	18.28 ± 3.23	11.45 ± 1.75	36.88 ± 3.15
60	64.98	18.42 ± 3.13	11.46 ± 1.64	36.86 ± 2.82
80	65.24	18.02 ± 3.32	11.36 ± 1.71	36.88 ± 2.84
100	65.53	18.37 ± 3.29	11.16 ± 1.67	37.05 ± 2.69

Figure A4 presents the complete trends. The horizontal axis denotes the number of reused messages n , and the vertical axis denotes PSNR (dB). The estimated-cover PSNR rises from approximately 60 dB to 65.5 dB as n increases and then tends to saturate. The recovery PSNR obtained using the estimated cover and the correct weights remains at 18–19 dB, whereas the baseline recovery PSNR obtained using the true cover remains stable at approximately 37 dB. Recovery with averaged weights yields a PSNR of approximately 11 dB, which is indistinguishable from random guessing. These results show that even if an attacker intercepts all stego images and weights, averaging alone cannot produce a cover sufficiently accurate for effective secret recovery, which remains far below the baseline.

**Figure A4.** Trends in estimated-cover accuracy and secret-recovery quality as the number of reused messages increases in the cover-reuse attack.

Appendix F. Sensitivity to α and Wavelet Selection

The α sweep fixes the wavelet to Db8, while the wavelet sweep fixes $\alpha = 0.2$. Table A7 shows that increasing α concentrates P more strongly in low-cost regions, reduces the activated-pixel ratio, and increases stego-image PSNR, but decreases recovered-secret PSNR. An overly strong prior therefore restricts the available modification locations and harms recovery. Table A8 shows that directional frequency resolution improves from Db1 to Db8, allowing the cost map to characterize textured edges more precisely. The activated-pixel ratio falls while both stego-image and recovered-secret PSNR increase, indicating that Db8 provides better texture discrimination and embedding guidance.

Table A7. Sensitivity to the exponential-mapping factor α .

α	Stego PSNR (dB)	Secret PSNR (dB)	Activation Ratio
0.05	48.93 $\pm$ 2.15	38.21 $\pm$ 3.42	0.8874 $\pm$ 0.061
0.10	49.87 $\pm$ 2.31	37.84 $\pm$ 3.51	0.8645 $\pm$ 0.058
0.20	50.80 $\pm$ 2.59	37.22 $\pm$ 3.77	0.8424 $\pm$ 0.056
0.40	51.62 $\pm$ 2.74	36.41 $\pm$ 3.86	0.8123 $\pm$ 0.055
0.80	52.18 $\pm$ 2.88	35.28 $\pm$ 3.95	0.7746 $\pm$ 0.053

Table A8. Sensitivity to wavelet selection.

Wavelet	Stego PSNR (dB)	Secret PSNR (dB)	Activation Ratio
Db1	48.25 $\pm$ 2.48	36.38 $\pm$ 3.62	0.8734 $\pm$ 0.060
Db2	49.12 $\pm$ 2.53	36.79 $\pm$ 3.68	0.8589 $\pm$ 0.058
Db4	49.95 $\pm$ 2.56	37.01 $\pm$ 3.72	0.8491 $\pm$ 0.057
Db8	50.80 $\pm$ 2.59	37.22 $\pm$ 3.77	0.8424 $\pm$ 0.056

Appendix G. Deployability Analysis on Edge Processors

The deployment analysis in this work is based on experimental results obtained on a single NVIDIA RTX 5060 GPU with 8 GB of memory. On the sender side, CMAD requires 2000 joint optimization iterations per image pair, with an average runtime of approximately 27 s and a peak memory usage of about 1809 MB. This computational cost is suitable for high-performance personal computers equipped with discrete GPUs or edge servers, but remains impractical for real-time processing on low-power edge nodes. On the receiver side, only a single forward decoding pass is required, taking approximately 2 s, which is computationally lightweight and amenable to deployment on resource-constrained edge devices. With additional techniques such as network pruning and 8-bit quantization, the decoder can be further compressed and accelerated using edge NPUs.

To alleviate the sender-side computational bottleneck, adaptive early stopping can be introduced to reduce the number of optimization iterations, while mixed-precision training and partial parameter freezing can be employed to lower memory consumption and computational overhead. Overall, CMAD exhibits an asymmetric deployment profile: the sender is computationally intensive, whereas the receiver is lightweight and can be migrated to edge terminals. With the aforementioned strategies, the sender side has the potential to approach near-real-time operation on edge nodes with moderate computational capacity.

References

1. Song, B.; Wei, P.; Wu, S.; Lin, Y.; Zhou, W. A survey on deep-learning-based image steganography. *Expert Syst. Appl.* **2024**, *254*, 124390. [\[CrossRef\]](#)
2. Luo, W.; Wei, K.; Li, Q.; Ye, M.; Tan, S.; Tang, W.; Huang, J. A comprehensive survey of digital image steganography and steganalysis. *APSIPA Trans. Signal Inf. Process.* **2024**, *13*, 2024. [\[CrossRef\]](#)
3. Filler, T.; Judas, J.; Fridrich, J.J. Minimizing additive distortion in steganography using syndrome-trellis codes. *IEEE Trans. Inf. Forensics Secur.* **2011**, *6*, 920–935. [\[CrossRef\]](#)
4. Chen, Y.; Fan, W. Prediction error ordering reversible data hiding via self-similarity and bicubic interpolation. *Electronics* **2026**, *15*, 115.
5. Zhu, J.; Kaplan, R.; Johnson, J.; Fei-Fei, L. HiDDeN: Hiding data with deep networks. In *Computer Vision—ECCV 2018*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 682–697. [\[CrossRef\]](#)
6. Zhang, K.A.; Cuesta-Infante, A.; Xu, L.; Veeramachaneni, K. SteganoGAN: High-capacity image steganography with GANs. *arXiv* **2019**, arXiv:1901.03892.
7. Tang, Y.; Xie, Y.; Zhang, W. Affine subspace robust low-rank self-representation: From matrix to tensor. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 9357–9373. [\[CrossRef\]](#) [\[PubMed\]](#)
8. Meng, X.; Li, Y.; Li, W.; Guo, Y.; Dong, Y.; Sun, H. Deep learning image steganography based on dual-path fusion in frequency and spatial domains. *Electronics* **2026**, *15*, 2777. [\[CrossRef\]](#)

9. Tan, L.; Li, Y.; Chen, P.; Zhang, Y. A hybrid adversarial embedding scheme with gradient calculator for JPEG steganography. *Electronics* **2024**, *14*, 4046.
10. Woźniak, K.; Ogiela, M.R.; Ogiela, L. Distribution-preserving latent steganography via conditional optimal transport. *Electronics* **2026**, *15*, 1321. [[CrossRef](#)]
11. Kishore, V.; Chen, X.; Wang, Y.; Li, B.; Weinberger, K.Q. Fixed neural network steganography: Train the images, not the network. In Proceedings of the International Conference on Learning Representations (ICLR), Virtual, 25–29 April 2022.
12. Luo, Z.; Li, S.; Li, G.; Qian, Z.; Zhang, X. Securing fixed neural network steganography. In *Proceedings of the 31st ACM International Conference on Multimedia*; ACM: New York, NY, USA, 2023; pp. 7943–7951. [[CrossRef](#)]
13. Li, G.; Li, S.; Qian, Z.; Zhang, X. Cover-separable fixed neural network steganography via deep generative models. In *Proceedings of the 32nd ACM International Conference on Multimedia*; ACM: New York, NY, USA, 2024; pp. 10238–10247. [[CrossRef](#)]
14. Cheng, Y.; Zhou, J.; Chen, J.; Yin, Z.; Zhang, X. RFNNS: Robust fixed neural network steganography with universal text-to-image models. *Proc. AAAI Conf. Artif. Intell.* **2026**, *40*, 3318–3326. [[CrossRef](#)]
15. Pevný, T.; Filler, T.; Bas, P. Using high-dimensional image models to perform highly undetectable steganography. In *Information Hiding*; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6387, pp. 161–177. [[CrossRef](#)]
16. Holub, V.; Fridrich, J. Designing steganographic distortion using directional filters. In Proceedings of the 2012 IEEE International Workshop on Information Forensics and Security (WIFS), Virtual, 12–16 December 2012; pp. 234–239. [[CrossRef](#)]
17. Holub, V.; Fridrich, J.; Denemark, T. Universal distortion function for steganography in an arbitrary domain. *EURASIP J. Inf. Secur.* **2014**, *2014*, 1–13. [[CrossRef](#)]
18. Li, B.; Wang, M.; Huang, J.; Li, X. A new cost function for spatial image steganography. In Proceedings of the 2014 IEEE International Conference on Image Processing (ICIP), Bordeaux, France, 16–19 October 2014; pp. 4206–4210. [[CrossRef](#)]
19. Tang, Y.; Zhang, M.; Lai, P.; Yue, Y.; Di, F. Fixed neural network image steganography based on secure diffusion models. *Comput. Mater. Contin.* **2025**, *84*, 5733–5750. [[CrossRef](#)]
20. Qiu, R.; Luo, Z.; Fan, R.; Cao, N.; Wang, Y.; Yin, C. Generative image steganography based on flip watermarking. *Electronics* **2026**, *15*, 939. [[CrossRef](#)]
21. Zhang, K.; Zuo, W.; Chen, Y.; Meng, D.; Zhang, L. Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Trans. Image Process.* **2017**, *26*, 3142–3155. [[CrossRef](#)] [[PubMed](#)]
22. Tan, J.; Liao, X.; Liu, J.; Cao, Y.; Jiang, H. Channel attention image steganography with generative adversarial networks. *IEEE Trans. Netw. Sci. Eng.* **2022**, *9*, 888–903. [[CrossRef](#)]
23. Shi, W.; Caballero, J.; Huszár, F.; Totz, J.; Aitken, A.P.; Bishop, R.; Rueckert, D.; Wang, Z. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 19–24 June 2016; pp. 1874–1883. [[CrossRef](#)]
24. Zhang, R.; Isola, P.; Efros, A.A.; Shechtman, E.; Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–24 June 2018; pp. 586–595.
25. Boehm, B. StegExpose—A tool for detecting LSB steganography. *arXiv* **2014**, arXiv:1410.6656.
26. Ye, J.; Ni, J.; Yi, Y. Deep learning hierarchical representations for image steganalysis. *IEEE Trans. Inf. Forensics Secur.* **2017**, *12*, 2545–2557. [[CrossRef](#)]
27. Boroumand, M.; Chen, M.; Fridrich, J. Deep residual network for steganalysis of digital images. *IEEE Trans. Inf. Forensics Secur.* **2019**, *14*, 1181–1193. [[CrossRef](#)]
28. You, W.; Zhang, H.; Zhao, X. A Siamese CNN for image steganalysis. *IEEE Trans. Inf. Forensics Secur.* **2020**, *16*, 1–14. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.