FROST: IMPROVING IMAGE FIDELITY AND WATERMARK ROBUSTNESS WITHOUT UPDATING THE BASE MODEL

Anonymous authors
Paper under double-blind review

ABSTRACT

Invisible image watermarking aims to preserve the visual quality of cover images while enabling reliable recovery of embedded messages after image transformations. In autoencoder-based systems such as RoSteALS, output distortion can arise from both message embedding and image reconstruction. We introduce FROST (Frozen-Reader Output-Space Tuning), a framework for improving image fidelity and message recovery robustness without updating the pretrained watermarking model. Its trainable component, the FROST Module, is attached after the image decoder and predicts an additive residual conditioned on the watermarked image and the embedded message. All existing image and message encoders and decoders remain frozen. The complete FROST Adapter combines this module with reconstruction-error correction and high-frequency residual suppression at inference time. Component-wise comparisons show that the module improves bit accuracy under image corruptions, whereas the subsequent residual processing improves fidelity. With 100-bit messages, the complete framework increases PSNR on CLIC from 29.10 to 33.66 dB. Across CLIC and MetFaces, the reported mean bit accuracy over 15 selected ImageNet-C-style corruptions increases from 94.18% to 95.74% at severity 3 and from 89.17% to 91.67% at severity 5. Experiments with a KL-f8 backbone also improve PSNR, SSIM, and average bit accuracy under corruption. However, LPIPS improvements are not consistent across datasets and backbones. These results demonstrate that learned output adjustment and inference-time residual processing can improve evaluated watermarking systems while preserving their pretrained components.

1 INTRODUCTION

Invisible image watermarking embeds a message while preserving the visual quality of a cover image and seeks to recover that message reliably after image transformations. This requires balancing image distortion against message recovery error. Changes that improve message recovery can introduce visible distortion, whereas processing that reduces distortion can weaken the embedded information. This trade-off is therefore central to the design of watermarking architectures and training objectives.

RoSteALS (Bui et al., 2023) embeds messages in the latent representation of a pretrained autoencoder, separating image representation learning from message embedding and recovery. We study how to improve image fidelity and message recovery robustness after such a watermarking system has already been trained, without changing its existing components. Under this constraint, output modifications introduced by an additional module must remain compatible with the pretrained message decoder while limiting their deviation from the original image.

1.1 RESEARCH PROBLEM AND PROPOSED APPROACH

We propose the FROST Module, a lightweight network operating after the image decoder. It takes the existing watermarked image and the embedded message as input and predicts an additional image residual. We freeze the image encoder, image decoder, message encoder, and message decoder,

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

and optimize only the module while considering both image fidelity and message recovery after corruption.

We refer to the complete pipeline comprising the learned module, reconstruction-error correction, and high-frequency residual suppression as the FROST Adapter. Reconstruction-error correction constructs a residual relative to an unwatermarked reconstruction of the same cover image. The subsequent B_0 removal step suppresses high-frequency variation in this residual before it is added to the original image. Only the FROST Module is trainable; the two residual-processing operations are applied at inference time.

The module uses FiLM (Perez et al., 2018) for message conditioning. In our component-wise comparisons, the variant without FiLM exhibits higher LPIPS (Zhang et al., 2018) on both evaluation datasets, with comparatively small differences in mean bit accuracy.

1.2 Main Results and Contributions

For RoSteALS with the VQ-GAN backbone (Esser et al., 2021), the complete FROST Adapter increases PSNR from 29.10 to 33.66 dB on CLIC and from 33.25 to 35.72 dB on MetFaces. The reported mean bit accuracy over the 15 selected corruptions increases from 94.18% to 95.74% at severity 3 and from 89.17% to 91.67% at severity 5. Experiments with a KL-f8 Rombach et al. (2022) backbone also improve PSNR, SSIM (Wang et al., 2004), and average bit accuracy under corruption. Perceptual improvements, however, depend on the dataset and backbone.

Our contributions are threefold:

1. We propose the FROST Module, which learns message-conditioned residual adjustments after the image decoder while keeping the pretrained watermarking components frozen.
2. We construct the FROST Adapter by combining the learned module with reconstruction-error correction and high-frequency residual suppression at inference time.
3. We analyze component-wise and backbone-dependent effects, examining the internal perceptual loss, message conditioning, residual smoothing, and an additional KL-f8 configuration to characterize the benefits and limitations of the approach.

2 Related Work

2.1 Learning-Based Image Watermarking

Learning-based watermarking jointly considers image distortion and message recovery error when training embedding and decoding networks. HiDDeN (Zhu et al., 2018) jointly trains an embedding encoder and a message decoder, incorporating image transformations during training to support recovery after blurring, cropping, and JPEG compression. TrustMark (Bui et al., 2025) balances image quality and recovery accuracy through its architecture and spatial–spectral objectives.

Other approaches address high-resolution embedding and imperceptibility. InvisMark (Xu et al., 2025) predicts a residual at a lower resolution and transfers it to the original resolution, together with robustness-oriented training. Pixel Seal (Souček et al., 2025) uses adversarial training for imperceptibility and separates robustness and imperceptibility learning into stages. These methods illustrate the importance of both embedding architectures and optimization objectives in balancing fidelity and recovery performance.

2.2 Watermarking with Pretrained Models

RoSteALS (Bui et al., 2023) freezes a pretrained image autoencoder and trains the message encoder and decoder. A message-dependent latent residual is added to the image representation before decoding, separating image representation learning from watermark embedding and extraction.

WMAdapter (Ci et al., 2025) inserts message-conditioned adapters into the VAE decoder of a latent diffusion model. Its initial training stage freezes both the VAE decoder and the pretrained watermark decoder and trains only the adapters. Our method similarly uses a fixed recovery network, but applies the FROST Module to an image that already contains a watermark, rather than inserting

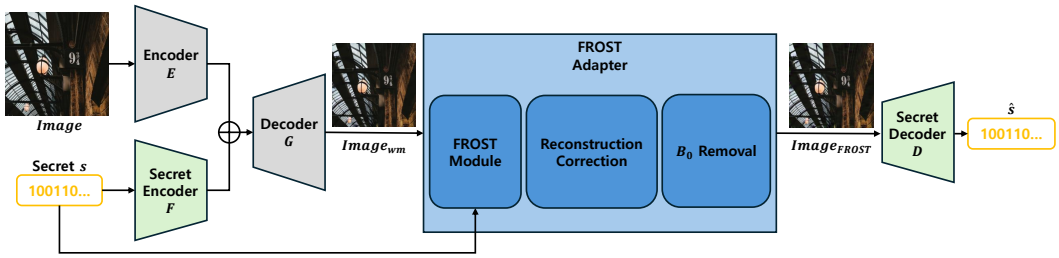


Figure 1: Overview of the FROST Adapter. The FROST Module performs learned output-space adjustment; reconstruction-error correction and B_0 removal are inference-time operations. The existing image and message encoders and decoders remain frozen during module training.

a message into internal decoder features. We retain both the image and message encoders and decoders throughout module training and combine output-space adaptation with reconstruction-error correction and high-frequency residual suppression at inference time.

3 Proposed Method

The FROST Module is a trainable network that predicts message-conditioned image residuals to adjust the output of a pretrained watermarking system. The FROST Adapter denotes the complete configuration consisting of this module, reconstruction-error correction, and B_0 removal. We keep the base watermarking model frozen and optimize only the module parameters.

3.1 Formulation

Let x denote a cover image and $s \in \{0,1\}^L$ a binary message of length L. We denote the image encoder and decoder by E and G, and the message encoder and decoder by F and D, respectively. Let ϵ be the latent residual produced by the message encoder. The base watermarking process is

$$z = E(x), \qquad \epsilon = F(z, s), \qquad y = G(z + \epsilon),$$

where y is the existing watermarked image, denoted by imgwm in the diagrams. The FROST Module produces an adjusted image conditioned on y and s:

$$y_M = H_\theta(y, s).$$

The module output y_M corresponds to imgout, whereas the complete adapter output x_F corresponds to $\mathrm{imgFROST}$. The parameters θ are trainable, while the parameters of E, G, F, and D remain fixed during module training.

Our goal is to preserve the content of x in the final image x_F while enabling reliable message recovery by the frozen decoder after image transformations. The cover image and message are available during embedding; extraction uses only the possibly transformed watermarked image and the existing decoder D.

3.2 Framework Overview

Figure 1 illustrates the complete FROST Adapter. The base watermarking system first produces y. The FROST Module then predicts an additional message-conditioned residual and adds it to y to produce y_M. Training considers the fidelity of y_M to the cover image, the magnitude of the modification relative to y, and message recovery after corruption.

At inference time, an unwatermarked reconstruction $x_0 = G(z)$ is generated from the same latent representation. The difference $y_M - x_0$ is used as a correction residual. High-frequency variation in this residual is suppressed before it is added to the original image. Neither reconstruction-error correction nor B_0 removal has trainable parameters, and neither is included in the loss computation for module training.

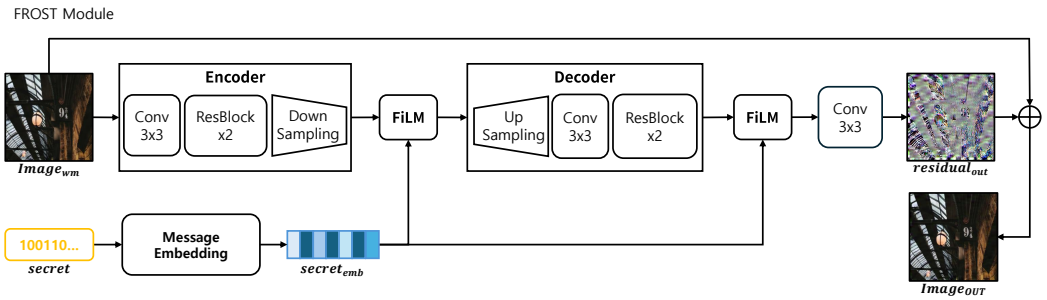


Figure 2: Architecture of the FROST Module. Message conditioning is applied after downsampling and after the decoder residual blocks. The final convolution predicts an image residual that is added to the input watermarked image. Residual smoothing, learned scaling, and output clipping are omitted from the diagram for clarity.

For KL-f8, we first use a watermarking system trained for that backbone as the baseline. We then freeze this system and train the FROST Module. Backbone-specific preparation and subsequent module adaptation are therefore separate training procedures.

3.3 FROST Module

3.3.1 Encoder-Decoder with Residual Blocks

The default FROST Module is a lightweight encoder–decoder with residual blocks and 315,652 trainable parameters(approximately 316K). The encoder first maps the RGB input to 32 channels using a 3×3 convolution, applies two residual blocks, and then uses a stride-2 convolution to halve the spatial resolution and increase the channel count to 64. For a 256×256 input, the bottleneck representation has dimensions $64 \times 128 \times 128$. The decoder performs bilinear upsampling, reduces the channel count to 32 through a convolution, and applies two residual blocks. Downsampling and upsampling are separate from the residual blocks.

Each residual block combines a shortcut with a learned feature transformation:

$$h_{\text{out}} = h + \mathcal{R}_\phi(h).$$

Here, h is the input feature and $\mathcal{R}_\phi$ is the residual branch, which uses convolutional and nonlinear transformations. These feature-level residuals differ from the image residual predicted by the full module. The within-block shortcuts are also distinct from long-range connections between encoder and decoder features.

We map the 100-bit message to a 128-dimensional conditioning representation and apply FiLM to the bottleneck and decoder features:

$$\text{FiLM}(h, s) = \gamma(s) \odot h + \beta(s),$$

where $\gamma(s)$ and $\beta(s)$ are message-dependent scale and shift parameters. Conditioning is applied at these two locations, not repeatedly within every residual block. It allows feature responses and image residuals to depend on the embedded message. We use this conditioning to control message-dependent residuals and examine the resulting outputs through component-wise comparisons.

The final 3×3 convolution predicts a three-channel residual $\Delta_\theta(y, s)$, rather than a complete image. After Gaussian smoothing and learned scaling, the residual is added to the input watermarked image:

$$r_\theta(y, s) = \alpha \mathcal{B}_{\sigma_A}\big(\Delta_\theta(y, s)\big), \qquad y_M = \Pi_{[-1,1]}\big[y + r_\theta(y, s)\big],$$

where $\mathcal{B}_{\sigma_A}$ denotes Gaussian smoothing and $\Pi_{[-1,1]}$ clips values to the valid image range. Thus, the module outputs an image, while the residual is an intermediate quantity. The residual output layer is zero-initialized, and the positive scale is parameterized as $\alpha = \exp(a)$ with $a = -4$ initially. The default smoothing parameter is $\sigma_A = 1.0$. Smoothing here acts only on the newly predicted residual and is distinct from the inference-time filtering of the reconstruction-relative residual.

3.3.2 TRAINING OBJECTIVE

We train the module to improve message recovery after corruption while limiting the difference between its output and both the cover image and the existing watermarked image. We augment the image–message objective of RoSteALS with an additional perceptual term and an output-preservation term:

$$\mathcal{L}_{\text{FROST}} = \mathcal{L}_{\text{IS}} + \lambda_{\text{perc}}\mathcal{L}_{\text{perc}}^{\text{out}} + \lambda_{\text{pres}}\mathcal{L}_{\text{pres}}.$$

The inherited objective comprises message recovery, reconstruction, and perceptual terms:

$$\mathcal{L}_{\text{IS}} = \lambda_{\text{bit}}\mathcal{L}_{\text{bit}} + \lambda_{\text{rec}}\mathcal{L}_{\text{rec}} + \lambda_{\text{perc}}^{\text{in}}\mathcal{L}_{\text{perc}}^{\text{in}}.$$

The internal coefficients denote the effective weights under the inherited weighting implementation, including its image-loss schedule; they need not be constant throughout training. The additional perceptual and preservation terms use fixed coefficients.

The message recovery term $\mathcal{L}_{\text{bit}}$ is the binary cross-entropy between the embedded message and the predictions of the frozen decoder from a corrupted module output. The reconstruction term $\mathcal{L}_{\text{rec}}$ measures error in YUV space, whereas the internal perceptual term $\mathcal{L}_{\text{perc}}^{\text{in}}$ uses LPIPS. The additional term $\mathcal{L}_{\text{perc}}^{\text{out}}$ also compares y_M with x using LPIPS but is applied outside the internal image-loss schedule. The two perceptual terms therefore compare the same image pair while differing in their weighting schedules.

We define the output-preservation loss as the mean absolute difference between the module input and output:

$$\mathcal{L}_{\text{pres}} = \frac{1}{N}\|y_M - y\|_1,$$

where N is the number of pixel–channel elements. Unlike reconstruction and perceptual losses, which compare the output with x, this term limits the change introduced relative to the existing watermarked image y. All pretrained components remain frozen, and only the FROST Module is updated. Reconstruction-error correction and B_0 removal are used only at inference time.

3.4 OUTPUT CONSTRUCTION OF THE FROST ADAPTER

3.4.1 RECONSTRUCTION-ERROR CORRECTION

Using the same latent representation z, we generate an unwatermarked reconstruction and compute the correction residual:

$$x_0 = G(z), \qquad r_{\text{corr}} = y_M - x_0, \qquad x_{\text{corr}} = \Pi_{[-1,1]}\big[x + r_{\text{corr}}\big].$$

The same construction is applied to both VQ-GAN and KL-f8, using matched latent representations and image resolutions. Before clipping, $x + r_{\text{corr}} = y_M - (x_0 - x)$, so the operation compensates for the unwatermarked reconstruction error in the module output. The correction residual is a difference between two outputs; it is not assumed to contain only message information.

3.4.2 HIGH-FREQUENCY RESIDUAL SUPPRESSION

We apply a Gaussian low-pass filter to attenuate rapid spatial variation in the correction residual. We define the B_0 component as the difference between the residual and its smoothed version:

$$B_0(r) = r - \mathcal{B}_{\sigma_B}(r).$$

Removing this component replaces the correction residual with its smoothed version. The final image is

$$x_F = \Pi_{[-1,1]}\left[x + \mathcal{B}_{\sigma_B}(y_M - x_0)\right].$$

We use $\sigma_B = 1.0$ and a 5×5 kernel. Filtering is applied to the correction residual rather than to the original image. Unlike the smoothing inside the module, this operation acts on a residual that includes changes from both the base watermarking pathway and the module. We do not assume that the suppressed component is free of message information; its effect is assessed in terms of both fidelity and message recovery.

4 EXPERIMENTS

We evaluate the module and inference-time residual processing on VQ-GAN-based RoSteALS, then examine an additional watermarking system trained with a KL-f8 backbone. Component-wise comparisons investigate the internal LPIPS term, module-level residual smoothing, and FiLM conditioning.

4.1 EXPERIMENTAL SETUP

4.1.1 DATASETS AND EVALUATION PROTOCOL

We use a 100,000-image subset of MIR-Flickr (Huiskes et al., 2010), with 99,000 images for training and 1,000 for validation. Evaluation uses 530 CLIC (Toderici et al., 2020) images and 1,336 MetFaces (Karras et al., 2020) images at 256×256 resolution. A single 100-bit message generated with seed 42 is shared across all evaluation images. We report raw bit accuracy without error correction.

Robustness is evaluated using a selected set of 15 ImageNet-C-style corruptions (Hendrycks & Dietterich, 2019) at severities 3 and 5. The evaluated corruption types are listed in Table **??**. Following the RoSteALS corruption implementation, images are converted from RGB to BGR before corruption and converted back afterward. For each corruption and severity, we average the CLIC and MetFaces bit accuracies with equal weight. IC-s3 and IC-s5 denote the averages over all 15 selected corruptions at the respective severities. Each table cell reports severity 3 / severity 5 accuracy in percent.

We measure image fidelity with PSNR and SSIM Wang et al. (2004), and perceptual discrepancy with LPIPS (Zhang et al., 2018). SSIM uses Gaussian weighting with $\sigma = 1.5$. Higher PSNR and SSIM and lower LPIPS indicate better performance. Clean-image bit accuracy is distinguished from accuracy after corruption.

4.1.2 TRAINING SETTING AND COMPARED CONFIGURATIONS

The default configuration, EXP-A, uses FiLM and module-level residual smoothing with an internal LPIPS weight of 0.1. The weights for output preservation, external LPIPS, message recovery, and reconstruction are 3.0, 1.0, 5.0, and 3.0, respectively. ImageNet-C augmentation is applied with probability 0.9.

All configurations are evaluated after 20,000 optimization steps. EXP-A uses two RTX 3090 GPUs with four images per GPU, giving a global batch size of eight. EXP-B, EXP-D, and EXP-E each use one RTX 3090 GPU with a batch size of four. These settings correspond to 160,000 and 80,000 cumulative image presentations, respectively. The results are therefore interpreted as component-wise comparisons rather than isolated estimates of each component's contribution.

EXP-B removes the internal LPIPS term from EXP-A, EXP-D removes residual smoothing from EXP-A, and EXP-E removes FiLM conditioning from EXP-A.

We distinguish four output configurations: the corresponding RoSteALS baseline, the FROST Module output, the module output with reconstruction-error correction, and the complete FROST Adapter. REC denotes reconstruction-error correction. Unless otherwise specified, comparisons are against the baseline using the same backbone.

4.2 RESULTS ON VQ-GAN-BASED ROSTEALS

Table 1 reports image quality for the default 20k-step configuration. The module alone yields lower PSNR and SSIM and higher LPIPS than the baseline. The complete FROST Adapter improves PSNR by 4.56 dB on CLIC and 2.47 dB on MetFaces, with higher SSIM on both datasets. LPIPS decreases from 0.0406 to 0.0267 on CLIC but increases slightly from 0.0381 to 0.0403 on MetFaces.

Figure 3 qualitatively compares the original image, RoSteALS baseline, FROST Module output, FROST Module with REC, and the complete FROST Adapter. The examples show the visual changes introduced at each stage of the output construction pipeline.

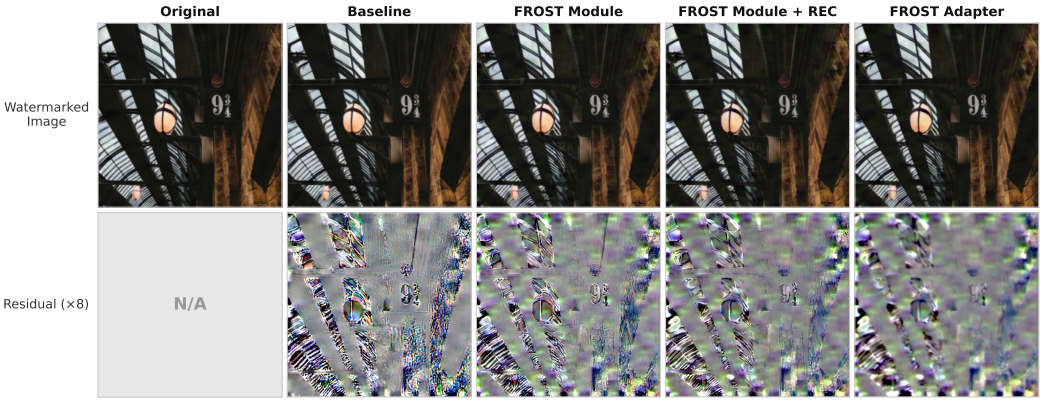


Figure 3: Qualitative comparison on VQ-GAN-based RoSteALS. From left to right: the original image, RoSteALS baseline, FROST Module output, FROST Module with reconstruction-error correction (REC), and the complete FROST Adapter. The examples illustrate the progressive changes introduced by the learned module and the subsequent inference-time residual processing.

Table 1: Image quality with the VQ-GAN backbone. REC denotes reconstruction-error correction. Best values within each dataset are bold.

Dataset	Configuration	PSNR ↑	SSIM ↑	LPIPS ↓
CLIC	RoSteALS	29.10	0.8827	0.0406
CLIC	+ FROST Module	28.67	0.8759	0.0570
CLIC	+ Module + REC	31.81	0.9571	0.0373
CLIC	FROST Adapter	**33.66**	**0.9781**	**0.0267**
MetFaces	RoSteALS	33.25	0.8940	**0.0381**
MetFaces	+ FROST Module	32.37	0.8864	0.0665
MetFaces	+ Module + REC	34.48	0.9617	0.0504
MetFaces	FROST Adapter	**35.72**	**0.9819**	0.0403

Table 2 presents corruption-wise results averaged across CLIC and MetFaces. The FROST Module improves IC-s3 and IC-s5 by 1.99 and 3.10 percentage points, respectively. Although residual processing reduces bit accuracy relative to the module-only output, the complete FROST Adapter remains 1.56 and 2.50 percentage points above the RoSteALS baseline. These results indicate a trade-off between image fidelity and message recovery rather than uniform improvements across all metrics.

The gains are particularly apparent for noise, JPEG compression, and frost. At severity 5, Gaussian-noise accuracy increases from 74.98% to 80.19% with the FROST Module and remains at 78.94% with the complete FROST Adapter. JPEG-compression accuracy follows a similar trend, increasing from 81.56% to 86.93% with the module and reaching 85.92% with the complete adapter. Elastic transformation remains challenging: at severity 3, accuracy decreases from 82.66% for the baseline to 81.80% for the complete adapter. Thus, improvements in mean robustness do not imply improvements under every corruption.

4.3 Component-Wise Comparisons

Table 3 compares direct outputs of the FROST Module variants. Because global batch size and cumulative image presentations differ between EXP-A and the other variants, the observed differences cannot be attributed solely to the removed components.

4.3.1 Internal LPIPS Term

EXP-B, which excludes the internal LPIPS term, shows higher LPIPS than EXP-A: 0.0907 vs. 0.0570 on CLIC and 0.1072 vs. 0.0665 on MetFaces. In contrast, the differences in IC-s3 and IC-s5

Table 2: Bit accuracy (%) under the selected corruptions for VQ-GAN-based RoSteALS and its FROST variants, averaged equally over CLIC and MetFaces. Each cell reports severity 3 / severity 5. The FROST Module denotes the direct module output; the complete FROST Adapter includes both residual-processing operations.

Corruption	RoSteALS	FROST Module	FROST Adapter
Gaussian Noise	85.94 / 74.98	91.85 / 80.19	90.71 / 78.94
Shot Noise	88.24 / 77.70	93.10 / 83.66	92.39 / 82.29
Impulse Noise	91.00 / 79.46	95.49 / 85.64	94.63 / 84.56
Defocus Blur	98.15 / 90.13	98.49 / 92.73	98.42 / 92.47
Frost	91.52 / 90.07	96.58 / 96.27	96.06 / 95.74
Fog	98.41 / 95.23	99.18 / 98.22	98.93 / 97.65
Brightness	99.45 / 99.04	99.37 / 99.06	99.22 / 98.85
Contrast	99.22 / 94.21	99.50 / 97.79	99.33 / 97.26
Elastic Transform	82.66 / 80.24	82.19 / 80.89	81.80 / 80.66
Pixelate	99.52 / 99.10	99.50 / 99.27	99.40 / 99.15
JPEG Compression	90.62 / 81.56	95.03 / 86.93	94.16 / 85.92
Speckle Noise	91.55 / 85.84	94.65 / 90.11	94.18 / 89.21
Gaussian Blur	99.05 / 94.95	99.22 / 96.85	99.05 / 96.64
Spatter	97.77 / 96.92	98.90 / 98.35	98.50 / 97.99
Saturate	99.56 / 98.12	99.47 / 98.16	99.35 / 97.80
Average	**94.18 / 89.17**	**96.17 / 92.27**	**95.74 / 91.67**

are relatively small. Under the evaluated training settings, EXP-B therefore shows higher LPIPS despite comparable mean bit accuracy.

4.3.2 Module-Level Residual Smoothing

EXP-D, which excludes module-level residual smoothing, achieves 95.62% IC-s3 and 91.26% IC-s5, compared with 96.17% and 92.27% for EXP-A. CLIC LPIPS is higher for EXP-D (0.0640 vs. 0.0570), whereas MetFaces LPIPS is unchanged at the reported precision (0.0665).

4.3.3 FiLM Conditioning

EXP-E, which excludes FiLM conditioning, exhibits higher LPIPS than EXP-A on CLIC (0.0754 vs. 0.0570) and MetFaces (0.0822 vs. 0.0665). Its IC-s3 and IC-s5 scores are lower by 0.18 and 0.33 percentage points, respectively.

Table 3: Component-wise comparison of FROST Module variants after 20,000 optimization steps. EXP-A uses a global batch size of 8 and 160K image presentations; EXP-B, EXP-D, and EXP-E use a global batch size of 4 and 80K image presentations. LPIPS is reported per dataset, and bit accuracy (%) is averaged equally over CLIC and MetFaces.

ID	Modification	CLIC LPIPS ↓	MetFaces LPIPS ↓	IC-s3 ↑	IC-s5 ↑
EXP-A	Default	**0.0570**	**0.0665**	96.17	92.27
EXP-B	w/o internal LPIPS	0.0907	0.1072	**96.29**	**92.40**
EXP-D	w/o residual smoothing	0.0640	**0.0665**	95.62	91.26
EXP-E	w/o FiLM	0.0754	0.0822	95.99	91.94

4.4 Application to a KL-f8 Backbone

To examine an additional backbone, we replace the VQ-GAN component of RoSteALS with the pretrained KL-f8 autoencoder released with latent diffusion models (Rombach et al., 2022), train the corresponding watermarking system, and then attach and train a FROST Module while freezing that system. This experiment evaluates adaptation to a second backbone, not zero-shot transfer of a module trained on VQ-GAN.

Table 4: Image quality with the KL-f8 backbone. REC denotes reconstruction-error correction. Best values within each dataset are bold.

Dataset	Configuration	PSNR ↑	SSIM ↑	LPIPS ↓
CLIC	KL-f8 baseline	29.81	0.8921	**0.0387**
CLIC	+ FROST Module	28.54	0.8485	0.0825
CLIC	+ Module + REC	31.74	0.9204	0.0606
CLIC	**FROST Adapter**	**33.62**	**0.9455**	0.0487
MetFaces	KL-f8 baseline	33.38	0.8805	**0.0410**
MetFaces	+ FROST Module	31.31	0.8329	0.1151
MetFaces	+ Module + REC	33.35	0.9123	0.0962
MetFaces	**FROST Adapter**	**34.99**	**0.9440**	0.0762

As shown in Table 4, the complete adapter improves PSNR by 3.81 dB on CLIC and 1.61 dB on MetFaces and also improves SSIM. However, LPIPS increases from 0.0387 to 0.0487 on CLIC and from 0.0410 to 0.0762 on MetFaces. Thus, the PSNR and SSIM gains do not imply a consistent improvement in perceptual quality.

Complete corruption-wise results are provided in Appendix Table 5. The complete adapter increases the mean accuracy from 93.47% to 95.38% at severity 3 and from 86.24% to 90.18% at severity 5. At severity 5, Gaussian-noise accuracy increases from 75.76% to 83.73%, and JPEG accuracy increases from 81.84% to 87.97%. However, saturation accuracy decreases from 96.82% to 92.66%, and elastic-transformation accuracy remains low at 74.64%.

4.5 Limitations

Our evaluation covers VQ-GAN-based RoSteALS and a RoSteALS system retrained with KL-f8. The results support the usefulness of frozen-base adaptation in these two configurations but do not establish generalization to arbitrary watermarking models or autoencoders.

Evaluation uses one fixed 100-bit message across all images, so performance variation across messages remains outside the reported evaluation. Perceptual improvements are also not uniform: LPIPS increases on MetFaces with the VQ-GAN backbone and on both datasets with KL-f8. Residual color artifacts in low-texture regions and weaknesses under saturation and elastic transformation remain. Higher PSNR and average bit accuracy therefore do not eliminate all perceptual or corruption-specific failures.

5 Conclusion

We introduce the FROST Module for adjusting the output of a trained watermarking system while retaining all pretrained parameters. The module uses a lightweight encoder–decoder with residual blocks and message conditioning to predict an image residual, which is added to the existing watermarked image. Training jointly considers message recovery, fidelity to the cover image, and preservation of the original watermarked output. The complete FROST Adapter combines the learned module with reconstruction-error correction and B_0 removal.

For VQ-GAN-based RoSteALS, the complete adapter improves PSNR and SSIM on both datasets and increases average bit accuracy at corruption severities 3 and 5. LPIPS decreases on CLIC but increases slightly on MetFaces.Under their respective training settings, the variants without the internal perceptual term or FiLM exhibit higher LPIPS, whereas the variant without residual smoothing exhibits lower mean bit accuracy than the default configuration. With KL-f8, PSNR, SSIM, and average bit accuracy also improve, although LPIPS deteriorates on both datasets.

These results demonstrate that an additional learned module and residual processing can improve the evaluated pretrained watermarking systems without updating their base components. Future work includes evaluation across more watermarking models and messages, as well as reducing low-texture color artifacts and corruption-specific vulnerabilities.

AI USE STATEMENT

Generative AI tools, including ChatGPT (OpenAI) and Claude (Anthropic), were used to assist with code development, verification of experimental results, retrieval and discovery of relevant literature, drafting portions of the manuscript, language polishing, and verification of mathematical expressions and claims. All AI-assisted code, experimental interpretations, references, and manuscript content were reviewed and verified by the authors against actual experimental results, source code, and original literature. The authors take full responsibility for the final content, results, and claims presented in this paper.

ETHICS STATEMENT

This work uses publicly available image datasets and pretrained watermarking models for research purposes. Invisible image watermarking can support applications such as content provenance tracking and copyright protection by enabling information to be embedded without substantially altering the visible content. However, such techniques may also be misused for covert tracking or other forms of undisclosed information embedding. In addition, watermark detection or message recovery should not be interpreted as conclusive evidence of content provenance or ownership, particularly in the presence of false positives, false negatives, or distribution shifts. Practical deployment should therefore consider transparency, appropriate disclosure, detection reliability, and the limitations of watermark-based attribution.

REPRODUCIBILITY STATEMENT

The experimental setup, datasets, evaluation protocol, training settings, and compared configurations are described in Section 4.1. The FROST Module architecture, training objective, and inference-time residual processing are specified in Section 3. Additional implementation details, corruption-wise results, training-cost information, and hyperparameters are provided in the appendix. We explicitly report differences in GPU count and effective batch size between the default configuration and the component-wise comparison variants. All reported results are obtained from the configurations and evaluation procedures described in the paper. Code for reproducing the core method and experiments is provided in anonymized form as supplementary material.

REFERENCES

Tu Bui, Shruti Agarwal, Ning Yu, and John Collomosse. RoSteALS: Robust steganography using autoencoder latent space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 933–942, 2023. doi: 10.1109/CVPRW59228.2023.00100.

Tu Bui, Shruti Agarwal, and John Collomosse. TrustMark: Robust watermarking and watermark removal for arbitrary resolution images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 18629–18639, 2025. doi: 10.1109/ICCV51701.2025.01731.

Hai Ci, Yiren Song, Pei Yang, Jinheng Xie, and Mike Zheng Shou. WMAdapter: Adding WaterMark control to latent diffusion models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 10901–10919. PMLR, 2025. URL https://proceedings.mlr.press/v267/ci25a.html.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12873–12883, 2021.

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

Pierre Fernandez, Hady Elsahar, I. Zeki Yalniz, and Alexandre Mourachko. Video seal: Open and efficient video watermarking. *arXiv preprint arXiv:2412.09492*, 2024.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations (ICLR)*, 2019. URL https://openreview.net/forum?id=HJz6tiCqYm.

Mark J. Huiskes, Bart Thomee, and Michael S. Lew. New trends and ideas in visual concept detection: The mir flickr retrieval evaluation initiative. In *Proceedings of the International Conference on Multimedia Information Retrieval*, pp. 527–536. ACM, 2010. doi: 10.1145/1743384.1743475.

Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In *Advances in Neural Information Processing Systems*, volume 33, pp. 12104–12114, 2020.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. doi: 10.1609/aaai.v32i1.11671.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Tom Sander, Pierre Fernandez, Alain Durmus, Teddy Furon, and Matthijs Douze. Watermark anything with localized messages. In *International Conference on Learning Representations (ICLR)*, 2025.

Tomáš Souček, Pierre Fernandez, Hady Elsahar, Sylvestre-Alvise Rebuffi, Valeriu Lacatusu, Tuan Tran, Tom Sander, and Alexandre Mourachko. Pixel Seal: Adversarial-only training for invisible image and video watermarking. arXiv preprint arXiv:2512.16874, 2025. URL https://arxiv.org/abs/2512.16874.

George Toderici, Wenzhe Shi, Radu Timofte, Lucas Theis, Johannes Ballé, Eirikur Agustsson, Nick Johnston, and Fabian Mentzer. Workshop and challenge on learned image compression (CLIC2020). CVPR Workshop and Challenge, 2020. URL https://www.compression.cc/.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. doi: 10.1109/TIP.2003.819861.

Rui Xu, Mengya Hu, Deren Lei, Yaxi Li, David Lowe, Alex Gorevski, Mingyu Wang, Emily Ching, and Alex Deng. InvisMark: Invisible and robust watermarking for AI-generated image provenance. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, pp. 909–918, 2025.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 586–595, 2018. doi: 10.1109/CVPR.2018.00068.

Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. HiDDeN: Hiding data with deep networks. In *Computer Vision – ECCV 2018*, volume 11219 of *Lecture Notes in Computer Science*, pp. 682–697. Springer, 2018. doi: 10.1007/978-3-030-01267-0_40.

Table 5: Bit accuracy (%) under the selected corruptions for KL-f8-based RoSteALS and its FROST variants, averaged equally over CLIC and MetFaces. Each cell reports severity 3 / severity 5. REC denotes reconstruction-error correction.

Corruption	RoSteALS	FROST Module	+ REC	FROST Adapter
Gaussian Noise	88.51 / 75.76	95.92 / 86.19	95.78 / 86.03	94.33 / 83.73
Shot Noise	90.32 / 78.81	96.08 / 88.52	96.03 / 88.41	94.69 / 86.35
Impulse Noise	92.66 / 80.13	97.13 / 90.14	97.06 / 90.05	96.03 / 87.95
Defocus Blur	92.53 / 71.31	93.68 / 75.41	93.69 / 75.39	93.97 / 74.95
Frost	88.75 / 86.66	96.47 / 96.09	96.28 / 95.94	95.40 / 94.60
Fog	97.93 / 90.44	99.06 / 98.03	99.03 / 97.85	98.64 / 97.09
Brightness	98.49 / 97.25	98.52 / 97.81	98.51 / 97.76	98.17 / 97.43
Contrast	98.91 / 95.25	99.23 / 98.12	99.17 / 98.05	98.88 / 97.33
Elastic Transform	76.83 / 72.85	76.48 / 75.12	76.52 / 74.97	76.69 / 74.64
Pixelate	98.94 / 97.99	98.92 / 98.51	98.88 / 98.42	98.62 / 98.17
JPEG Compression	92.62 / 81.84	97.11 / 90.16	96.98 / 90.10	95.96 / 87.97
Speckle Noise	92.63 / 86.98	96.33 / 92.43	96.26 / 92.37	95.25 / 90.82
Gaussian Blur	97.53 / 86.16	98.34 / 92.02	98.30 / 91.98	98.09 / 91.69
Spatter	96.21 / 95.42	98.20 / 97.88	98.09 / 97.68	97.57 / 97.25
Saturate	99.25 / 96.82	98.84 / 93.89	98.79 / 93.91	98.51 / 92.66
Average	**93.47 / 86.24**	**96.02 / 91.35**	**95.96 / 91.26**	**95.38 / 90.18**

A Visual Quality and Computational Cost

The module-only results show that incorporating reconstruction and perceptual objectives does not guarantee an improvement in PSNR. Instead, the learned adjustment primarily improves corrupted-image message recovery, while the subsequent residual processing improves fidelity. Neither these results nor the observed accuracy gains establish that the module moves watermark energy into a particular spatial or frequency band.

As observed during qualitative inspection, purple–green color artifacts can remain in low-texture regions. An increase in image-level PSNR does not imply that all localized, visually noticeable distortions have been removed. This is especially relevant to the KL-f8 results, where improvements in PSNR and SSIM coexist with higher LPIPS. The current observations do not identify a single component as the cause of these artifacts.

Table 6 summarizes the cumulative image presentations and estimated training times. The FROST Module is the only trainable addition, with approximately 316K parameters in the default configuration. Its training cost is an adaptation cost after the base watermarking model is available, not the cost of constructing the full system from scratch. At inference time, the complete adapter also executes the module, an unwatermarked reconstruction, and residual processing. The number of trainable parameters should therefore be distinguished from total inference cost.

Table 6: Estimated training volume and time. Image presentations are cumulative and need not correspond to unique images. Training times are converted from the collected training configurations and are intended as approximate comparisons.

Model	Image presentations	Estimated time (h)
RoSteALS	1.8M	101
Stable Signature (Fernandez et al., 2023)	24.8M	156
WAM (Sander et al., 2025)	59.0M	602
Pixel Seal	153.6M	2,216
Video Seal (Fernandez et al., 2024) image	307.2M	2,895
FROST Module	160k	7.2

B Implementation Settings

Table 7: Training settings for the default FROST Module configuration (EXP-A)

Parameter	Value
Optimizer	AdamW Loshchilov & Hutter (2019)
Learning rate	1×10^{-4}
Weight decay	1×10^{-4}
Global batch size	8
Batch size per GPU	4 / GPU
Training steps	20,000
GPU	2 $\times$ NVIDIA GeForce RTX 3090
Dataset	MIR-Flickr ($\sim$99K images, 256×256)
Secret length	100 bits
Augmentation	ImageNet-C ($p = 0.9$, severity ≤ 5)
Trainable parameters (EXP-A)	315,652