Ultra-high Resolution Watermarking Framework Resistant to Extreme Cropping and Scaling

Nan Sun[1], **LuYu Yuan**[1], **Han Fang**[2], **Yuxing Lu**[3], *__Hefei Ling__[1], **Sijing Xie**[1], **Chengxin Zhao**[1]
[1]School of Computer Science and Technology, Huazhong University of Science and Technology
[2]National University of Singapore
[3]Peking University
{sunnan, yuanly, lhefei,xiesijing,chengxinzhao }@hust.edu.cn
fanghan@nus.edu.sg, yxlu0613@gmail.com

Abstract

Recent developments in DNN-based image watermarking techniques have achieved impressive results in protecting digital content. However, most existing methods are constrained to low-resolution images as they need to encode the entire image, leading to prohibitive memory and computational costs when applied to high-resolution images. Moreover, they lack robustness to distortions prevalent in large-image transmission, such as extreme scaling and random cropping. To address these issues, we propose a novel watermarking method based on implicit neural representations (INRs). Leveraging the properties of INRs, our method employs resolution-independent coordinate sampling mechanism to generate watermarks pixel-wise, achieving ultra-high resolution watermark generation with fixed and limited memory and computational resources. This design ensures strong robustness in watermark extraction, even under extreme cropping and scaling distortions. Additionally, we introduce a hierarchical multi-scale coordinate embedding and a low-rank watermark injection strategy to ensure high-quality watermark generation and robust decoding. Experimental results show that our method significantly outperforms existing schemes in terms of both robustness and computational efficiency while preserving high image quality. Our approach achieves an accuracy greater than 98% in watermark extraction with only 0.4% of the image area in 2K images. These results highlight the effectiveness of our method, making it a promising solution for large-scale and high-resolution image watermarking applications.

1 Introduction

With the rapid progress of the digital age, images have become a fundamental medium of information exchange, reaching unprecedented scales in dissemination and application across various fields. Concurrently, image watermarking techniques have emerged as pivotal tools for copyright protection, data security, and integrity verification. However, with the increasing demand for processing large-scale and high-resolution images, DNN-based watermarking approaches face significant challenges in adapting to the requirements of large-scale image watermarking.

Typical deep learning-based image watermarking methods are generally designed for low-resolution images, requiring full-image processing (Zhu et al., 2018; Tancik et al., 2020a; Fang et al., 2022). As a result, these methods face significant limitations when applied to ultra-high resolution (UHR) images. First, processing the entire UHR image incurs high computational costs, resulting in long processing times and potential memory overflow, which impacts efficiency and feasibility. Second, in the decoding phase, UHR images are more vulnerable to scaling and cropping distortions during

*Corresponding Author (lhefei@hust.edu.cn)

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

transmission. Existing watermarking methods, optimized for low-resolution content, struggle to preserve watermark integrity under these severe transformations.

As shown in Figure 1 (a), existing methods for embedding information in high-resolution images typically use a block-based approach (Guo et al., 2023), where the image is divided into non-overlapping blocks, and the same watermark is embedded into each block. However, this approach has two main drawbacks. First, the accuracy of block localization is crucial for decoding performance; misalignment can significantly reduce extraction effectiveness. Second, because the watermark block is smaller than the original image, it is highly susceptible to scaling distortions. An alternative approach embeds the watermark on low-resolution images and then interpolates the residuals to higher resolutions for embedding (Bui et al., 2023). While this reduces the computational burden, it also makes the watermark more vulnerable to local cropping attacks, compromising its robustness.

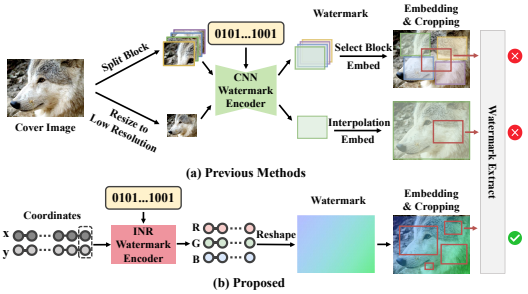


Figure 1: Different high resolution watermark embedding schemes. Where the color blocks on the image represent the watermark embedding area and the red boxes represent the area where the image is cropped for decoding the watermark.

To address the challenges of watermarking large-scale images, we propose an innovative solution based on Implicit Neural Representations (INRs). As illustrated in Figure 1 (b), our method maps continuous pixel coordinates directly to the corresponding RGB values of the watermark. This eliminates the constraint of fixed image resolutions, enabling watermark embedding in UHR images while ensuring robustness against extreme cropping and scaling distortions. The main contributions of this paper can be summarized as follows:

- We propose an innovative INR-based framework for ultra-high resolution watermarking, offering a groundbreaking solution to the challenges of watermarking high-resolution images.
- We introduce a hierarchical multi-Scale coordinates embedding mechanism for accurate watermark generation across scales. In addition, we introduce a low-rank injection scheme for efficient integration of watermarks.
- Extensive experiments on widely representative datasets demonstrate the exceptional performance and significant advantages of our proposed method in handling high-resolution images and accommodating diverse resolution scenarios. In addition, our method exhibits excellent resistance to extreme cropping and scaling that often occurs in high-resolution images.

2 Related Work

DNN-based Image Watermarking. Recent advances in deep learning have significantly impacted digital image watermarking. HiDDeN (Zhu et al., 2018) introduced an end-to-end DNN-based watermarking framework, resembling an autoencoder, setting the stage for future models. StegaStamp (Tancik et al., 2020a) improves print-shooting robustness by simulating the printing and photographing process. RIHOOP (Jia et al., 2020) further refines this by introducing a differentiable distortion model that preserves the integrity of the watermark under camera imaging conditions. Subsequent works (Jia et al., 2021; Fang et al., 2023; Li et al., 2024; Sun et al., 2024) focus on increasing robustness against various distortions. However, all of these methods are limited to fixed low-resolution images (typically less than 512). As the resolution of images increases, these methods encounter substantial challenges, including a significant rise in computational complexity and memory usage, which not only slow down processing times but also make their practical application with high-resolution images increasingly difficult.

High-Resolution Image Watermarking. Several approaches have been proposed to address the challenges of watermark embedding in high-resolution images. DWSF (Guo et al., 2023) uses a block-based strategy, selecting fixed-size watermark blocks for embedding, but it struggles with block localization and scaling resistance. TrustMark (Bui et al., 2023), on the other hand, generates

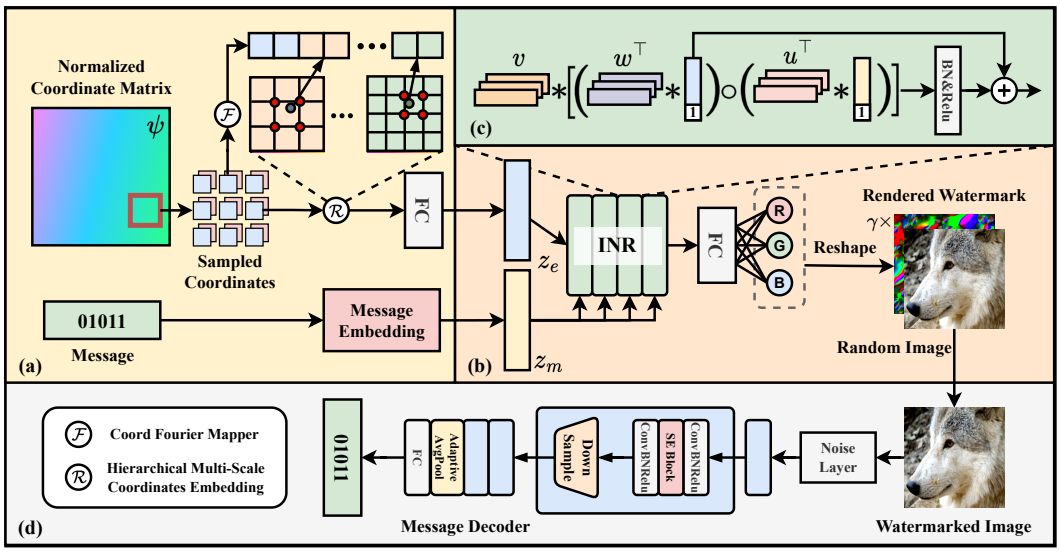


Figure 2: Architecture of the proposed model. (a) shows the process of embedding coordinates and messages, and (b) shows the process of rendering into watermarks via INR. (c) shows the process of low-rank watermark injection. (d) shows the decoding process of our method.

watermark residues on low-resolution images and then uses linear interpolation to scale them to high-resolution images. However, this approach is vulnerable to local cropping attacks. Wang et al. (Wang et al., 2024) use Implicit Neural Representations to fit the host image and then fine-tune the INR to embed watermark information at various resolutions. However, this method has significant drawbacks, as it requires the training of separate networks for each individual image. This process not only increases the computational load but also makes it highly time-consuming, particularly when dealing with large datasets, limiting its practicality for real-time applications. Although the above methods attempt to embed watermarks in large-sized images, none of them effectively balance high-rato cropping, scaling resilience, and real-time performance, which are key challenges in large image watermarking.

Implicit Neural Representations. Implicit Neural Representations (INRs) use deep neural networks to model continuous mappings between inputs and outputs, rather than relying on predefined rules. INRs have been widely applied in 3D reconstruction (Mildenhall et al., 2021; Gafni et al., 2021; Hui et al., 2024), super-resolution (Chen et al., 2021; Yang et al., 2021; Chen et al., 2022), and image generation (Skorokhodov et al., 2021; Shaham et al., 2021; Anokhin et al., 2021). In the image domain, an INR takes spatial coordinates as input and outputs RGB values, representing the image as a continuous signal. CNN-based watermarking methods struggle with large images due to high memory and computation costs. In contrast, INRs model continuous signals efficiently, offering a scalable solution. We propose using INRs to parameterize the watermark signal by coordinates, enabling continuous watermark generation at arbitrary positions and resolutions, overcoming the limitations of CNN-based methods for efficient embedding.

3 Methodology

Our approach is based on Implicit Neural Representations (INRs), the key idea of our method is to parametrize a template watermark using coordinates, with INRs serving as the rendering function for the watermark signal. A comprehensive architecture is presented in Figure 2, illustrating the key components of our framework. The approach is built upon three core modules as shown in the Figure: (a) a resolution-independent sampling strategy combined with hierarchical multi-scale coordinate embedding, ensuring consistency and robustness of watermark across varying image resolutions with a fixed and limited computational cost; (b) low-rank watermark injection based on Implicit Neural Representations, which reduces computational cost while achieving robust watermark embedding and (d) noise enhancement and decoding of the watermarked image.

3.1 Sampling and Embedding

Resolution-Independent Coordinates Sampling. For an image of size (H, W), we normalize the pixel coordinates (x, y) to the range $[-1, 1]$ as follows:

$$(x, y) \rightarrow \left(\frac{2x}{H} - 1, \frac{2y}{W} - 1\right) \tag{1}$$

The normalized coordinate matrix obtained is denoted as ψ in Figure 2 (a). To sample submatrices of arbitrary size from ψ, we use a fixed $r \times r$ coordinate grid $\mathcal{C}$, where r is typically set to 128. The coordinates of the sampled submatrix are determined by the upper-left corner (x_0, y_0), and $\mathcal{C}$ is defined as:

$$\mathcal{C} = \{(x_i, y_j) \mid x_i = x_0 + i \cdot \Delta_t, y_j = y_0 + j \cdot \Delta_t\} \tag{2}$$

where $i, j \in [0, r-1]$ and Δ_t is a randomly selected interval. The value of Δ_t controls the resolution of $\mathcal{C}$, allowing flexible extraction of regions at different scales while maintaining a fixed grid.

Hierarchical Multi-Scale Coordinates Embedding. Many prior works (Müller et al., 2022; Girish et al., 2023) have shown that using only coordinates as input leads to longer training times, loss of high-frequency details, and poor scalability for high-resolution signals. This is a challenge, as we aim to decode the watermark at arbitrary resolutions while maintaining robustness to cropping and scaling. Therefore, modeling multi-scale features of the watermark is crucial.

To address this issue, we propose the use of a set of feature grids with varying resolutions $L = \{L_i\}^n$ to represent the embedded features of the watermark template, where n (default 4) denotes the number of feature grids. Each grid $L_i \in \mathbb{R}^{d \times 2^{i+4} \times 2^{i+4}}$ is a learnable parameterized matrix, with d (default 32) representing the dimension of the features. The coordinates of the feature vectors in each grid are normalized to the range $[-1, 1]$. Given an input coordinate (x, y), we identify the four nearest corner features in the matrix L_i, with the bottom-left and top-right corner features having coordinates (x^i_{bl}, y^i_{bl}) and (x^i_{tr}, y^i_{tr}), respectively. By applying bilinear interpolation, we can obtain the feature representation corresponding to any input coordinate in matrix $L_i(x, y)$:

$$L_i(x, y) = \begin{bmatrix} x^i_{tr} - x & x - x^i_{bl} \end{bmatrix} \begin{bmatrix} L_i(x^i_{bl}, y^i_{bl}) & L_i(x^i_{bl}, y^i_{tr}) \\ L_i(x^i_{tr}, y^i_{bl}) & L_i(x^i_{tr}, y^i_{tr}) \end{bmatrix} \begin{bmatrix} y^i_{tr} - y \\ y - y^i_{bl} \end{bmatrix} k \tag{3}$$

where $k = \frac{1}{(x^i_{tr} - x^i_{bl})(y^i_{tr} - y^i_{bl})}$. Additionally, to further enhance the ability of INR to represent high-frequency details, we apply Fourier feature mapping $\mathcal{F}$ to process the raw coordinates (Tancik et al., 2020b), obtaining the coordinate encoding z_p. Therefore, for any given coordinate (x, y), we can obtain its corresponding feature representation $z_e \in \mathbb{R}^{d_e}$:

$$z_e = \Gamma(L_1(x, y) \odot L_2(x, y) \cdots L_n(x, y) \odot z_p) \tag{4}$$

where "$\odot$" denotes concatenation along the feature dimension and $\Gamma(*)$ is a linear layer used to project the features into the d_e-dimensional space (default 256).

Message Embedding. We use a four-layer MLP with fully connected layer, BatchNorm, and ReLU activation function as the Message Encoder $\mathcal{M}$. For a t-length binary message $m = \{0, 1\}^t$, we obtain the corresponding message feature $z_m = \mathcal{M}(m) \in \mathbb{R}^{d_m}$. d_m defaults to 128.

3.2 Low-rank Watermark Injection based on Implicit Neural Representations

INR Rendering. As shown in Figure 2 (b), given a positional feature embedding z_e and a watermark feature z_m, we use an INR to generate the watermark signal for the corresponding pixel location. This process is repeated for all pixel positions, and the results are reshaped to form the final watermark signal W. Then we randomly sample an $r \times r$ image block I and obtain the watermarked image as $I' = I + \gamma W$, where γ (default 0.02) controls embedding strength. A smaller γ helps distribute the watermark evenly, enhancing visual quality and robustness.

Low-rank Watermark Injection. A simple INR-based approach is to concatenate two features and process them through an MLP to predict RGB values. However, prior research (Zadeh et al., 2017; Liu et al., 2018) has demonstrated that such direct concatenation leads to inadequate feature

interaction between modalities. Following TFN (Zadeh et al., 2017), we compute the Cartesian product of z_e and z_m to facilitate richer cross-modal feature interactions. This can be formulated as:

$$z = M \cdot \mathrm{Vec}(\alpha\beta^\top) + bias \tag{5}$$

Where $\alpha = [z_e, 1] \in \mathbb{R}^n$, $\beta = [z_m, 1] \in \mathbb{R}^m$ and $\mathrm{Vec}(*)$ the vectorization operator. The matrix $M \in \mathbb{R}^{h\times(n\times m)}$ is a learnable parameter, and z is the final output feature. h is the dimension of the output feature, default is 256.

However watermark injection occurs independently at each pixel location in our method. This results in significant computational and memory overhead. To mitigate this, we reformulate it (ignoring bias) as:

$$z = M \cdot \mathrm{Vec}(I\alpha\beta^\top) = M(\beta \otimes I)\mathrm{Vec}(\alpha) = M(\beta \otimes I)\alpha \tag{6}$$

We expand M and $\beta \otimes I$ into a block matrix form as follows:

$$z = [M_1 \quad M_2 \quad \cdots \quad M_m] \begin{bmatrix} \beta_1 I \\ \beta_2 I \\ \vdots \\ \beta_m I \end{bmatrix} \alpha = \sum_{i=1}^{m} \beta_i M_i \alpha \tag{7}$$

Where $M_i \in \mathbb{R}^{h\times n}$ can be viewed as a slice of a third-order tensor $P \in \mathbb{R}^{m\times h\times n}$, and $\sum_{i=1}^{m} \beta_i M_i$ can be interpreted as a weighted sum of the slices of P. Thus, our goal is to reduce the parameter count of the learnable tensor P.

Using Canonical Polyadic Decomposition (CPD) for low-rank approximation, we introduce three small learnable matrices $u \in \mathbb{R}^{m\times d}$, $v \in \mathbb{R}^{h\times d}$, and $w \in \mathbb{R}^{n\times d}$, where d is the rank (defaults 32). Thus, any element in P can be expressed as $p_{ijk} = \sum_{r=1}^{d} u_{ir}v_{jr}w_{kr}$. Then for any element of the output feature z it can be expressed as:

$$\begin{aligned} z_j = \sum_{k=1}^{n}\sum_{i=1}^{m} \beta_i p_{ijk}\alpha_k &= \sum_{k=1}^{n}\sum_{i=1}^{m} \beta_i \sum_{r=1}^{d} u_{ir}v_{jr}w_{kr}\alpha_k \\ &= \sum_{k=1}^{n}\sum_{i=1}^{m}\sum_{r=1}^{d} \beta_i u_{ir}v_{jr}w_{kr}\alpha_k \end{aligned} \tag{8}$$

After simplification, we obtain: $z = v*((w^\top * \alpha) \circ (u^\top * \beta))$. Where "$*$" denotes matrix multiplication and "$\circ$" denotes element-wise multiplication. To preserve the rich semantic information contained in the features α, we employ skip-connection to mitigate potential information loss caused by low-rank decomposition. Additionally, we stack 4 identical modules to ensure the robust injection of the watermark information. Figure 2 (c) illustrates our INR Block. It can be expressed as:

$$\alpha_i = \mathrm{Relu}(\Psi(v_{i-1} * ((w_{i-1}^\top * \alpha_{i-1}) \circ (u_{i-1}^\top * \beta)))) + \alpha_{i-1} \tag{9}$$

Where $\Psi(*)$ denotes 1D batch normalization. A final FC layer maps features to RGB values, and after reshaping, the watermark $W \in \mathbb{R}^{3\times r\times r}$ is obtained.

3.3 Noise Layer and Message Decoder

Figure 2 (d) illustrates the decoding process of our model. To enhance generalization, we introduce a composite noise layer to simulate real-world distortions. The Message Decoder then extracts the watermark from the distorted image.

Noise Layer. The noise layer consists of Rotation, Cropping, Translation, Scaling, Shearing, Dropout, Cropout, Color changes, JPEG compression, Gaussian filtering, and Gaussian noise. During training, a random noise type is applied to the watermarked image I', producing $\hat{I}$.

Message Decoder. Our Message Decoder incorporates the SE Block (Hu et al., 2018) inspired by MBRS (Jia et al., 2021). It processes the input through four stacked blocks of identical structure. Each block consists of a ConvBNReLU layer with a kernel size of 3, followed by an SE Block and another convolutional layer. Finally, downsampling is performed using a convolutional layer with a kernel size of 4 and a stride of 2. The extracted features are pooled along the channel dimension, and a fully connected layer predicts the watermark information $\hat{m}$.

3.4 Loss Function

Our loss function comprises two components: the first aims to preserve the visual quality of the watermarked image $I^{'}$, while the second seeks to minimize the discrepancy between the extracted watermark $\hat{m}$ and the embedded watermark m. Both components are formulated using the mean squared error (MSE) loss. The total loss is given by:

$$\mathcal{L} = \lambda_1 \left\| I - \hat{I} \right\|_2 + \lambda_2 \left\| m - \hat{m} \right\|_2 \tag{10}$$

where λ_1 and λ_2 are hyperparameters that balance the trade-off between visual quality preservation and watermark extraction accuracy. By default, both are set to 1.

4 Experiments

In this section, we conduct extensive experiments to evaluate the effectiveness of our proposed INR-based watermarking method. First, we describe the experimental setup. Then, we compare our method with previous SOTA models under low resolution. Subsequently, we evaluate our method across various resolutions against other large-image watermarking approaches, demonstrating its superior performance. Finally, ablation studies assess the contribution of proposed components.

4.1 Experimental Setting

Implementation Details. Our model is trained on the high-resolution DIV2K (Agustsson and Timofte, 2017) image dataset. For each training iteration, we randomly select an image from the dataset and apply a random scaling operation, where the scaling factor is chosen from the range $[0.06, 1]$. Following the scaling, we randomly crop a 128×128 image patch from the scaled image, which serves as the input I to the model. For the watermark information m, we randomly generate a binary bit stream of length 30 in each iteration. For the sampling coordinates $\mathcal{C}$, we perform random sampling from a normalized coordinate grid ψ, using a fixed 128×128 grid size. We randomly generate training samples, with a training set size of $50,000$ samples. The model is trained using the AdamW (Loshchilov and Hutter, 2017) optimizer with a learning rate of 4×10^{-4}. The batch size is set to 32, and the training is conducted for 2000 epochs across two NVIDIA RTX 3090 24G GPUs.

Metrics. We evaluate our method using three metrics: Peak Signal-to-Noise Ratio(PSNR) for visual quality, Structural Similarity Index(SSIM) for structural similarity and Average Bit Accuracy (ACC) for average decoding accuracy. PSNR and SSIM assess image quality, while ACC measures watermark extraction robustness.

Baseline. We use HiDDeN (Zhu et al., 2018), StegaStamp (Tancik et al., 2020a), MBRS (Jia et al., 2021), DWSF (Guo et al., 2023), TrustMark (Bui et al., 2023) and RAIM_{ARK} (Wang et al., 2024) as baselines. The first three methods are limited to low-resolution images, while others can work at different resolutions. To ensure a fair comparison, we re-train these models (excluding TrustMark, which does not provide a training script) with our proposed noise layer, as the strength of the noise layer significantly impacts their performance. Initially, we train our model on 128×128 images and compare them with the low-resolution baselines to demonstrate the effectiveness of our method. Subsequently, we relax the resolution constraint, allowing models to be trained on images with different sizes, and compare them with corresponding large-image watermarking models to validate the superiority of our approach.

4.2 Visual Quality

Table 1 presents the visual quality across different methods. Our method does not achieve the highest PSNR and SSIM values, but it consistently maintains a PSNR above 35. This result is expected, as our approach employs a template-based watermark, which does not leverage the content of the cover image for embedding. As a result, while the visual quality is slightly lower compared to methods that embed watermarks based on the cover image content, our method still achieves a PSNR above 35, ensuring it meets practical requirements for everyday use.

In addition, as shown in Figure 3, we also present the visualization results of our method at different resolutions. We observe that watermarks generated from different embedded messages exhibit

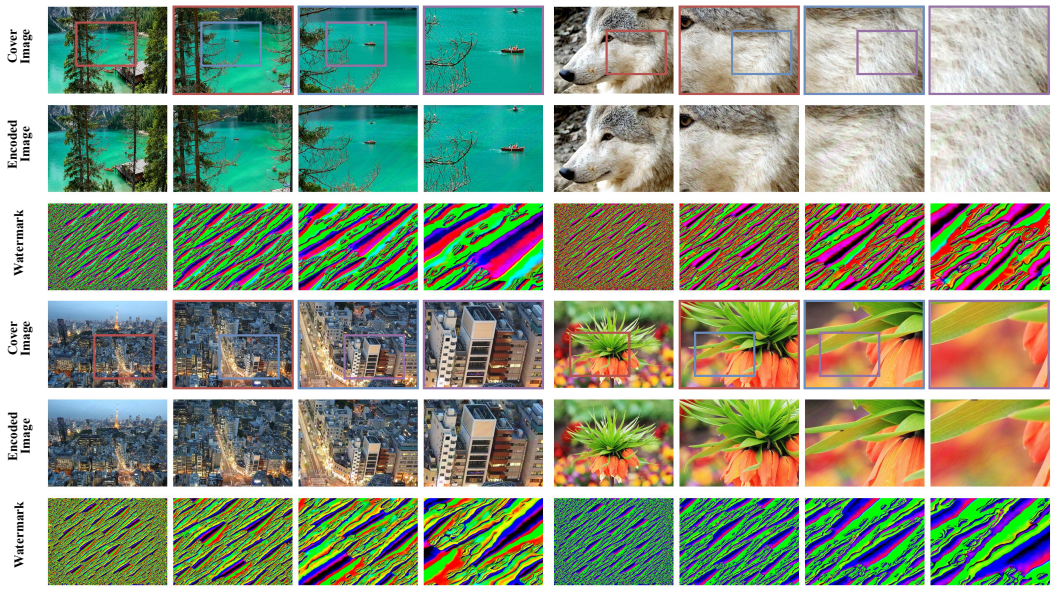


Figure 3: Visualization results at different resolutions. For each group, the first row presents cover images at different resolutions, starting from 2K size and progressively reduced by 75%. The second row displays the corresponding watermarked images, while the third row shows the embedded watermarks.

	High Resolution				Low Resolution		
	Proposed	DWSF	TrustMark	RAIM	HiDDeN	StegaStamp	MBRS
PSNR	37.27	33.18	40.76	40.52	32.67	35.25	40.73
SSIM	0.9611	0.9269	0.9900	0.9898	0.8591	0.8864	0.9866

Table 1: Average visual quality of the different methods. For High Resolution methods, we measured at 2K size (except for DWSF). For Low Resolution methods, we measured at 128×128 size. For DWSF, it essentially embeds the watermark into many 128×128 blocks, so we measured the visual quality of the block.

highly similar overall structural patterns, with variations primarily reflected in color and fine details. Moreover, the watermark consistently maintains a similar striped pattern and distribution across various resolutions, indicating its structural consistency across different scales. This ensures that the watermark information can be successfully decoded from cropped patches at different resolutions.

4.3 Comparison with Previous Methods

We compare our method with previous SOTA models at a fixed resolution of 128×128 to evaluate its effectiveness. For a comprehensive evaluation, we test the decoding accuracy of our trained model using a variety of distortions: Identity, Gaussian Noise($std = 0.01$), Gaussian Filter ($\sigma = 2$), JPEG Compression ($Q = 50$), Dropout ($p = 0.5$), Rotation ($deg = 10$), Translation ($dis = 0.1$) and Color Transform($f = 0.1$). To ensure a fair comparison, we fix the PSNR at 35 for all watermarked images following the approach in the MBRS (Jia et al., 2021) method.

Table 2 presents the experimental results. Notably, despite employing INRs rather than traditional CNNs for information embedding, our proposed method achieves the best decoding accuracy under most distortion conditions. Specifically, the proposed method maintains extremely high accuracy under conditions such as Gaussian noise, Gaussian filtering, rotation, translation, and color perturbation. Even under more severe transformations like JPEG compression and dropout operations, the method demonstrates strong robustness, achieving accuracy rates of 93.36% and 95.13%, respectively.

Method	Identity	Gaussian Noise ($std = 0.01$)	Gaussian Filter ($\sigma = 2$)	JPEG Compression ($Q = 50$)	Dropout (0.5)	Rotation ($deg = 10$)	Translation ($dis = 0.1$)	Color ($f = 0.1$)
HiDDeN	88.34	87.96	60.14	52.96	74.74	82.94	82.85	90.79
StegaStamp	92.16	91.72	90.78	84.42	77.53	87.19	88.46	91.21
MBRS	99.18	98.06	94.93	**96.56**	**95.75**	95.37	96.27	98.37
DWSF	90.17	89.76	89.40	87.33	76.47	86.56	65.90	78.36
TrustMark	87.65	83.72	82.60	75.22	76.58	49.50	62.13	87.25
RAIM_{ARK}	78.67	78.67	78.67	54.33	57.67	77.24	74.58	77.67
Proposed	**100**	**99.83**	**98.86**	93.36	95.13	**99.73**	**99.59**	**99.93**

Table 2: Benchmark comparisons on robustness against different distortions. Where the mean value of Gaussian Noise is 0, the kernel size of Gaussian Filter is 3, and JPEG Compression is simulated using Kornia (Riba et al., 2020).

Distortions	Model	128×128			512×512			2048×2048			4096×4096		
		DIV2K	COCO	FFHQ	DIV2K	COCO	FFHQ	DIV2K	COCO	FFHQ	DIV2K	COCO	FFHQ
Cropping	DWSF	90.17	89.83	78.93	53.28	54.10	52.73	50.80	50.73	50.64	50.03	50.33	50.12
	TrustMark	87.65	92.15	96.10	49.55	49.45	50.23	49.77	49.58	49.41	49.50	54.57	47.25
	RAIM_{ARK}	78.67	77.33	78.30	53.33	48.67	45.33	54.00	48.67	46.67	54.35	49.46	46.85
	Proposed	**99.99**	**99.85**	**99.91**	**99.86**	**99.84**	**99.95**	**98.74**	**94.80**	**93.83**	**85.94**	**86.93**	**83.23**
Scaling	DWSF	90.17	89.83	78.93	89.56	89.40	77.90	80.70	80.83	67.07	63.73	63.43	56.57
	TrustMark	87.65	92.15	96.10	79.15	85.90	89.43	76.07	86.25	88.88	79.51	82.26	86.77
	RAIM_{ARK}	78.67	77.33	78.30	61.67	63.76	66.52	62.33	64.37	63.16	62.67	64.48	62.89
	Proposed	**99.99**	**99.85**	**99.91**	**99.86**	**99.88**	**99.96**	**98.66**	**99.16**	**99.33**	**99.33**	**92.83**	**98.66**

Table 3: Average decoding accuracy of different models for extreme cropping and scaling at different resolutions.

This robust performance strongly demonstrates that INR-based embedding is an effective and highly resilient mechanism for steganography.

4.4 Evaluation across Varying Image Resolutions

In this section, we investigate the performance of our method across different resolutions. We compare it with several SOTA watermarking methods capable of operating at various resolutions.

As modern applications increasingly involve high-resolution images such as digital media, medical imaging, and professional photography, scalable and reliable watermarking technology has become essential. For high-resolution images, two key challenges are commonly encountered during transmission: (1) High-ratio scaling, where the image is significantly reduced in size and (2) High-ratio random cropping, where only a small portion of the image is retained. These operations severely compromise the integrity of embedded watermarks, particularly when using traditional spatial domain or CNN-based methods.

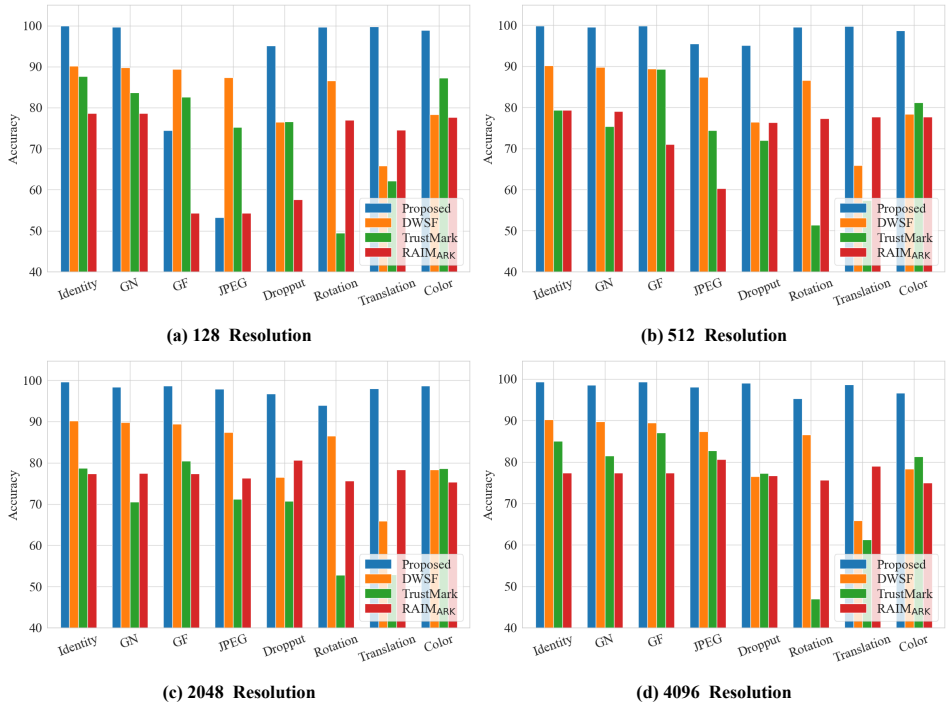
Table 3 presents the experimental results. To evaluate the generalizability of our model, we test it not only on DIV2K but also on 200 separately sampled images from each of the COCO (Lin et al., 2014) and FFHQ (Karras et al., 2019) datasets. We embed watermarks at different resolutions and evaluate the performance of various methods under extreme cropping and scaling. Specifically, for cropping, we randomly extract 128×128 patches for decoding, while for scaling, we uniformly resize the images to 128×128 before decoding. To ensure a fair comparison, the PSNR is fixed at 35 for all watermarked images.

As can be seen, our method is far superior to the others. DWSF, RAIM_{ARK} and TrustMark are all struggling to resist cropping attacks in large-image watermarking. Although these methods perform reasonably well at low resolutions (128×128), they encounter issues in high-resolution scenarios. For instance, DWSF cannot withstand cropping at high resolutions because it is difficult to ensure that the cropped image block contains a complete embedding block. TrustMark, which embeds watermarks at high resolutions through interpolation, retains less watermark information after cropping, leading to extraction failure. On the contrary, on 2K images, our method requires only **0.4%** of the image area to maintain a **98%** decoding accuracy. Notably, this high level of robustness extends even to 4K resolution, where our method still achieves over 85% decoding accuracy using only a 128×128 patch, which corresponds to just 0.1% of the total image area. This superiority primarily stems from our coordinate sampling strategy and INR's capability to model continuous signals.

Model	Proposed	DWSF	TrustMark	RAIM$_{\text{ARK}}$
CPU	✔	✖	✖	✖
Embedding Rate	**4ms**	74ms	308ms	$>$ 20min

Table 4: Embedding rates for different methods.Where the second row represents whether the embedding is done using only the CPU or not, and the third row represents the average watermark embedding rate for a single image with a resolution of 2K. We use AMD Ryzen 7 7840HS for our CPU and NVIDIA RTX 3090 24G for our GPU for testing.

In addition, we test the performance of different distortions at different resolutions. As shown in Figure 4, our model achieves a performance of more than 90% at most resolutions, which is much better than other models. At the same time, we observe an interesting phenomenon: at low resolutions, the performance of Gaussian filtering and JPEG compression deteriorates. Both types of distortion are related to the image's frequency content, suggesting that the current method still has limitations.



(a) 128 Resolution **(b) 512 Resolution**

(c) 2048 Resolution **(d) 4096 Resolution**

Figure 4: Average decoding accuracy of different models at different resolutions and distortions. Where GN stands for Gaussian Noise and GF stands for Gaussian Filter. To ensure fairness, the PSNR of all embedded images is standardized at 35.

4.5 Embedding Rate Comparison

In this subsection, we compare the computational efficiency between the different methods by the embedding rate of the watermark. Since our focus is on watermark embedding for large images, which often incur high computational costs due to their large resolution, the real-time performance of watermarking systems can be significantly impacted. As shown in Table 4, our method is significantly faster. This result primarily becauses it is based on template watermarking, allowing all watermarks to be pre-generated once the model is trained. In contrast, other methods require the network to process the carrier image during embedding, resulting in slower performance. Worth mentioning is that RAIM$_{\text{ARK}}$ requires INR encoding and watermark embedding fine-tuning for each image, making its embedding rate extremely slow.

	Coordinates Embedding (num of layers)				Low-rank Injection (rank)			
	✖	1	2	4	✖	8	32	64
ACC	90.91	91.66	93.86	**95.17**	93.77	84.69	95.17	**95.76**
PSNR	35.98	36.88	36.75	**37.27**	36.04	37.08	**37.27**	36.37
SSIM	**0.9706**	0.9590	0.9565	0.9611	0.9639	**0.9656**	0.9611	0.9550

Table 5: Model performance with different components. Here, "✖" indicates the absence of the corresponding component. For Coordinates Embedding, this means that only the Coordinate Fourier Mapper is used as an embedding feature. For Low-rank Injection, it signifies that features are concatenated and directly predicted by an MLP.

Num of Bit	PSNR	SSIM	Avg ACC
30	37.27	0.9656	95.76
50	35.59	0.9563	95.62
100	31.32	0.9123	88.26

Table 6: The relationship between visual quality and decoding accuracy at varying numbers of bits.

4.6 Ablation Study

In this subsection, we examine the effectiveness of each component of our proposed method, focusing on two main points: (1) the impact of the hierarchical multi-Scale coordinates embedding layers on model performance, and (2) the effectiveness of low-rank watermark injection.

Table 5 presents the results of our ablation study. We embed watermarks into 2K-resolution images and evaluate the average decoding accuracy after applying random attacks. The results show that removing multi-scale coordinate embedding significantly degrades performance, while increasing the number of embedding layers progressively improves it. For Low-rank Injection, replacing our design with feature concatenation followed by MLP prediction results in inferior performance. Moreover, a lower rank significantly degrades accuracy.

Additionally, we evaluated the watermarking capacity of our method. Table 6 illustrates the trade-off between watermark capacity, visual quality, and decoding robustness. As the embedded bit count increases from 30 to 100, the visual quality of images deteriorates, while accuracy remains at a high level. Even under 100 bits, the accuracy exceeds 88%. Since our method handles high-intensity cropping and scaling while maintaining high accuracy under current high payloads, it indicates that our embedding strategy inherently introduces significant redundancy. Therefore, exploring more efficient embedding patterns in the future to reduce redundancy and increase watermark capacity while maintaining robustness will be a highly meaningful research direction.

5 Conclusion

In this paper, we introduce a novel watermarking framework that leverages implicit neural representations and a resolution-independent coordinate sampling for efficient watermark embedding and extraction across images of high resolution. Unlike CNN-based methods that require processing entire images, our approach can embed watermark at the pixel level, enabling watermark embedding across different scales while avoiding excessive computational overhead and ensuring robustness against extreme cropping and scaling distortions. Additionally, we introduce a hierarchical multi-scale coordinate embedding and low-rank watermark injection to enhance model robustness. Experimental results show that our method outperforms existing approaches in both performance and computational efficiency. These findings highlight the potential of INR-based methods for high resolution watermarking solutions, offering valuable insights for future research on resolution-independent image watermarking.

Acknowledgments and Disclosure of Funding

This work was supported in part by the Natural Science Foundation of China under Grant 62372203 and 62302186, in part by the Major Scientific and Technological Project of Shenzhen (202316021), in part by the National key research and development program of China(2022YFB2601802), in part by the Major Scientific and Technological Project of Hubei Province (2022BAA046, 2022BAA042).

References

Eirikur Agustsson and Radu Timofte. 2017. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 126–135.

Ivan Anokhin, Kirill Demochkin, Taras Khakhulin, Gleb Sterkin, Victor Lempitsky, and Denis Korzhenkov. 2021. Image generators with conditionally-independent pixel synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14278–14287.

Tu Bui, Shruti Agarwal, and John Collomosse. 2023. TrustMark: Universal Watermarking for Arbitrary Resolution Images. *arXiv preprint arXiv:2311.18297* (2023).

Yinbo Chen, Sifei Liu, and Xiaolong Wang. 2021. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 8628–8638.

Zeyuan Chen, Yinbo Chen, Jingwen Liu, Xingqian Xu, Vidit Goel, Zhangyang Wang, Humphrey Shi, and Xiaolong Wang. 2022. Videoinr: Learning video implicit neural representation for continuous space-time super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2047–2057.

Han Fang, Zhaoyang Jia, Zehua Ma, Ee-Chien Chang, and Weiming Zhang. 2022. PIMoG: An Effective Screen-shooting Noise-Layer Simulation for Deep-Learning-Based Watermarking Network. In *MM '22: The 30th ACM International Conference on Multimedia, Lisboa, Portugal, October 10 - 14, 2022*, João Magalhães, Alberto Del Bimbo, Shin'ichi Satoh, Nicu Sebe, Xavier Alameda-Pineda, Qin Jin, Vincent Oria, and Laura Toni (Eds.). ACM, 2267–2275. doi:10.1145/3503161.3548049

Han Fang, Yupeng Qiu, Kejiang Chen, Jiyi Zhang, Weiming Zhang, and Ee-Chien Chang. 2023. Flow-based robust watermarking with invertible noise layer for black-box distortions. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 37. 5054–5061.

Guy Gafni, Justus Thies, Michael Zollhofer, and Matthias Nießner. 2021. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8649–8658.

Sharath Girish, Abhinav Shrivastava, and Kamal Gupta. 2023. Shacira: Scalable hash-grid compression for implicit neural representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 17513–17524.

Hengchang Guo, Qilong Zhang, Junwei Luo, Feng Guo, Wenbin Zhang, Xiaodong Su, and Minglei Li. 2023. Practical deep dispersed watermarking with synchronization and fusion. In *Proceedings of the 31st ACM International Conference on Multimedia*. 7922–7932.

Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7132–7141.

Mude Hui, Zihao Wei, Hongru Zhu, Fei Xia, and Yuyin Zhou. 2024. MicroDiffusion: Implicit Representation-Guided Diffusion for 3D Reconstruction from Limited 2D Microscopy Projections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11460–11469.

Jun Jia, Zhongpai Gao, Kang Chen, Menghan Hu, Xiongkuo Min, Guangtao Zhai, and Xiaokang Yang. 2020. RIHOOP: Robust invisible hyperlinks in offline and online photographs. *IEEE Transactions on Cybernetics* 52, 7 (2020), 7094–7106.

Zhaoyang Jia, Han Fang, and Weiming Zhang. 2021. Mbrs: Enhancing robustness of dnn-based watermarking by mini-batch of real and simulated jpeg compression. In *Proceedings of the 29th ACM international conference on multimedia*. 41–49.

Tero Karras, Samuli Laine, and Timo Aila. 2019. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 4401–4410.

Yiyi Li, Xin Liao, and Xiaoshuai Wu. 2024. Screen-Shooting Resistant Watermarking with Grayscale Deviation Simulation. *IEEE Transactions on Multimedia* (2024).

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*. Springer, 740–755.

Zhun Liu, Ying Shen, Varun Bharadhwaj Lakshminarasimhan, Paul Pu Liang, Amir Zadeh, and Louis-Philippe Morency. 2018. Efficient low-rank multimodal fusion with modality-specific factors. *arXiv preprint arXiv:1806.00064* (2018).

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101* (2017).

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)* 41, 4 (2022), 1–15.

Edgar Riba, Dmytro Mishkin, Daniel Ponsa, Ethan Rublee, and Gary Bradski. 2020. Kornia: an open source differentiable computer vision library for pytorch. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 3674–3683.

Tamar Rott Shaham, Michaël Gharbi, Richard Zhang, Eli Shechtman, and Tomer Michaeli. 2021. Spatially-adaptive pixelwise networks for fast image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14882–14891.

Ivan Skorokhodov, Savva Ignatyev, and Mohamed Elhoseiny. 2021. Adversarial generation of continuous images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10753–10764.

Nan Sun, Han Fang, Yuxing Lu, Chengxin Zhao, and Hefei Ling. 2024. END^2: Robust Dual-Decoder Watermarking Framework Against Non-Differentiable Distortions. *arXiv preprint arXiv:2412.09960* (2024).

Matthew Tancik, Ben Mildenhall, and Ren Ng. 2020a. Stegastamp: Invisible hyperlinks in physical photographs. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2117–2126.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. 2020b. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems* 33 (2020), 7537–7547.

Yuchen Wang, Xingyu Zhu, Guanhui Ye, Shiyao Zhang, and Xuetao Wei. 2024. Achieving Resolution-Agnostic DNN-based Image Watermarking: A Novel Perspective of Implicit Neural Representation. In *Proceedings of the 32nd ACM International Conference on Multimedia*. 10354–10362.

Jingyu Yang, Sheng Shen, Huanjing Yue, and Kun Li. 2021. Implicit transformer network for screen content image continuous super-resolution. *Advances in Neural Information Processing Systems* 34 (2021), 13304–13315.

Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2017. Tensor fusion network for multimodal sentiment analysis. *arXiv preprint arXiv:1707.07250* (2017).

Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. 2018. Hidden: Hiding data with deep networks. In *Proceedings of the European conference on computer vision (ECCV)*. 657–672.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Section 1 explains our contributions

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 4.2 and Supplementary Material B explain our limitations

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Section 3.2 gives the full proof.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 4.1 gives details.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Section 4.1 and Supplementary Material A.3 give details.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4.1 gives details.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Section 4 gives details.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section 4.1 gives details.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.

- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: this paper ensures that the research adheres to ethical standards and the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: The paper is no societal impact of the work performed.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: the paper poses no such risks.

    Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification: he paper does not use existing assets.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: the paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

Justification: the paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: the paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: the core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Appendix

A.1 Details of the Noise Layer

The combined noise layer is implemented using Kornia (Riba et al., 2020), incorporating various transformations such as Identity, Rotation, Cropping, Translation, Scaling, Shearing, Dropout, Cropout, Color Transformation, JPEG Compression, Gaussian Filtering, and Gaussian Noise. These transformations are applied as follows:

- **Rotation:** Random rotation within the angle range of $[-30°, 30°]$.
- **Cropping:** Randomly retains a scale of $[0.5, 1]$ of the original image.
- **Translation:** Generates random displacements of $[-0.1, 0.1]$ times the image's edge length along the x and y axes.
- **Scaling:** Random scaling within a factor of $[0.5, 1.2]$.
- **Shearing:** Random shear transformation with an angle range of $[-0.1, 0.1]$.
- **Dropout:** Randomly discards $[10\%, 30\%]$ of the pixels.
- **Cropout:** Randomly discards blocks of the image with a scale of $[0.05, 0.1]$.
- **Color Transformation:** Perturbs Brightness, Saturation, and Hue with intensities of $[-0.4, 0.4]$, $[-0.4, 0.4]$, and $[-0.1, 0.1]$, respectively.
- **JPEG Compression:** Applies random quality factors between $[50, 100]$.
- **Gaussian Filter:** Applies Gaussian filters with kernel sizes in the range of $[3, 8]$ and intensities of $[0.05, 0.1]$ with $\sigma \in [0.1, 2]$.
- **Gaussian Noise:** Adds Gaussian noise with a mean of 0 and variance of 0.01.

During training, a random noise type is selected and applied to perturb the watermarked image.

A.2 The Role of Hyperparameter γ

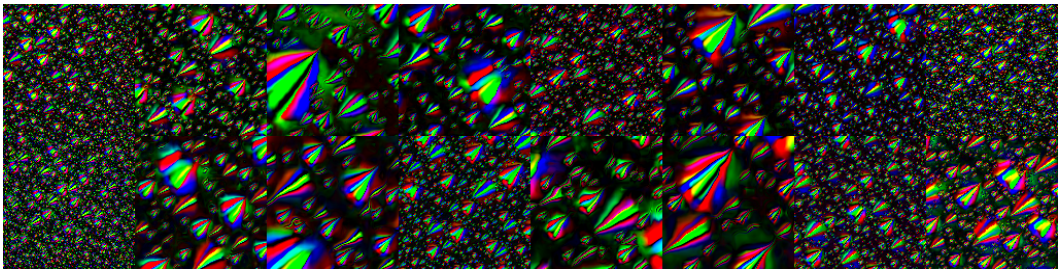
In the section "Low-rank Watermark Injection based on Implicit Neural Representations", we introduced the γ parameter to regulate the embedding strength of the watermark. Since our INR-based watermarking approach essentially functions as a template watermark, its pattern remains independent of the carrier image. As a result, the model naturally tends to generate sparse high-frequency textures to minimize visual loss, leading to significant spatial inefficiencies.

As illustrated in Figure 5, the left side shows the watermark generated without constraints. A substantial portion of the area contains null values, which not only wastes available space but also poses a critical issue—If the image is cropped to a region containing only null values, the essential watermark information may be lost entirely during transmission. To address this, we impose a constraint on the embedding strength, ensuring that the watermark is more uniformly distributed across the image.

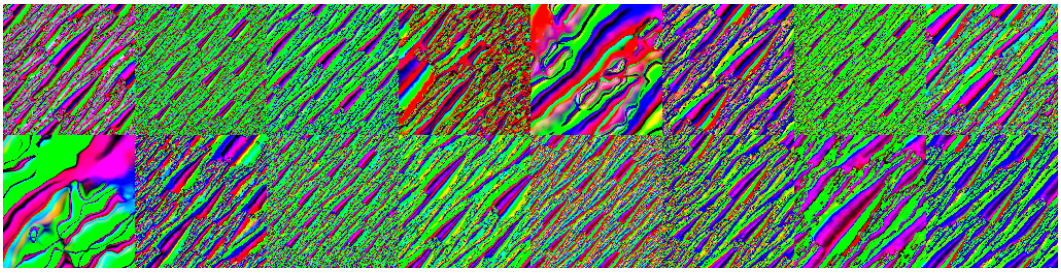
Moreover, we empirically set $\gamma = 0.02$, as it maintains a PSNR of approximately 34 while preserving good visual quality, even for a completely randomized watermark template. If γ is reduced further, the watermark's robustness deteriorates due to insufficient redundancy.

A.3 TrustMark Settings in Baseline

Since TrustMark does not provide a training script, we rely on its pre-trained weights for testing. However, TrustMark's pre-trained watermarks are 100 bits long and are available in four open-source versions. We use the version with a 40-bit payload (BCH_SUPER), with the remaining 60 bits reserved for error correction and versioning. To ensure a relatively fair comparison, we modify the decoding process: after extracting the full 100-bit sequence, we first apply error correction to the 40 valid bits and discard the remaining 60 bits. The evaluation is then based solely on the decoding accuracy of the 40 valid bits.



(a) w/o γ



(b) Proposed

Figure 5: Different watermark styles generated by the hyperparameter γ.

A.4 More Results

In the main text, we present visualization results across different resolutions. Here, we provide additional experimental results, as shown in Figure 6. We randomly select various resolutions for embedding, where each group's first row represents the cover images, the second row shows the watermarked images, and the third row displays the watermarks. It can be observed that while the watermarks exhibit scale-dependent variations at different resolutions, they consistently retain a similar stripe-like structure and maintain considerable complexity at each resolution. This ensures the successful decoding of watermark information across varying image scales.

B Limitations and Future Work

Our approach shows promise but has some limitations. The visual quality of the watermarked images is lower than content-dependent methods, mainly due to INR's weaker feature representation compared to CNN. Additionally, INR fitting can be slow. Future work will aim to address these issues by combining CNN with INR to enhance both visual quality and embedding efficiency. We will also explore optimizing INR fitting to reduce training time and improve scalability. Moreover, how to better enhance watermark capacity is another issue that needs to be addressed.

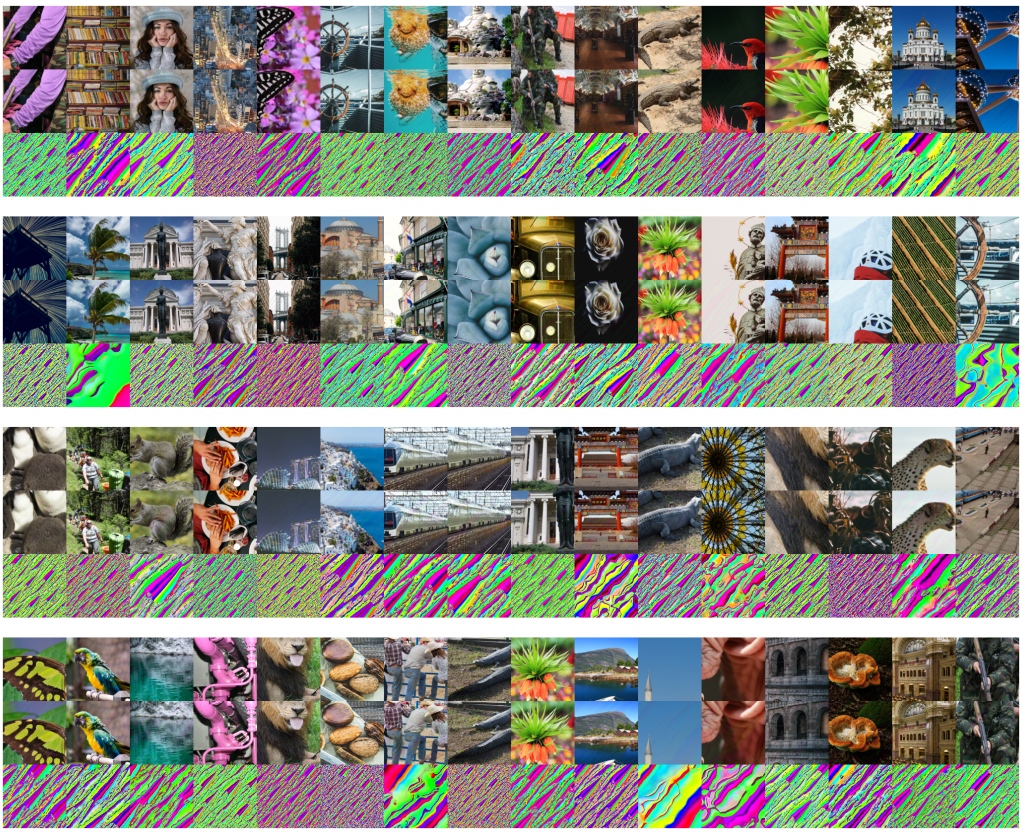


Figure 6: Visualization results at different resolutions.