MultiNeRF: Multiple Watermark Embedding for Neural Radiance Fields

Yash Kulthe, Andrew Gilbert & John Collomosse
Centre for Vision Speech and Signal Processing
University of Surrey, UK
{y.kulthe, a.gilbert, j.collomosse}@surrey.ac.uk

Abstract

We present **MultiNeRF**, a novel 3D watermarking method that enables the embedding of multiple uniquely keyed watermarks within images rendered by a single Neural Radiance Field (NeRF) model while maintaining high visual quality. Our approach extends the TensoRF NeRF model by incorporating a dedicated watermark grid alongside the existing geometry and appearance grids. This ensures higher watermark capacity without entangling watermark signals with scene content. We propose a FiLM-based conditional modulation mechanism that dynamically activates watermarks based on input identifiers, allowing multiple independent watermarks to be embedded and extracted without requiring model retraining. We validate MultiNeRF on the NeRF-Synthetic and LLFF datasets, demonstrating statistically significant improvements in robust capacity without compromising rendering quality. By generalizing single-watermark NeRF methods into a flexible multi-watermarking framework, MultiNeRF provides a scalable solution for securing ownership and attribution in 3D content.

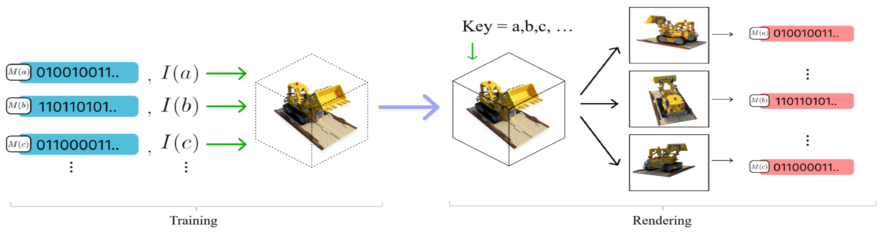

Figure 1: **MultiNeRF** embeds multiple watermarks within the representation learned by a NeRF model (TensoRF) at training time. Watermarks are keyed by a unique ID specified at rendering time to trigger the embedding of the watermark into the image independent of the viewing position.

1 Introduction

Neural Radiance Fields (NeRFs) (Mildenhall et al., 2021) have become a powerful high-fidelity 3D scene representation method, enabling photorealistic novel view synthesis. Their uses include online gaming, immersive experiences, and large-scale metaverse environments (Fabra et al., 2024; Zhang, 2024; Li et al., 2022). However, this also introduces new challenges in intellectual property (IP) protection, as NeRF models – which can be expensive to produce – can be easily shared or leaked, whether representing products, scenes, or avatars. Establishing the provenance of a NeRF model is crucial for asserting ownership and rights, helping to mitigate these risks, and even opening up novel compensation frameworks for their reuse (Collomosse & Parsons, 2024).

Digital watermarking has been a cornerstone of IP protection for visual media, including images (Zhu et al., 2018; Fernandez et al., 2022; Bui et al., 2023a) and video (Fernandez et al., 2024). However, these methods fall short in the NeRF context because they protect only the 2D outputs (the rendered images) rather than the underlying 3D representation itself. Recent work has therefore focused on watermarking models directly, for example, text-to-image diffusion models Fernandez et al. (2023) and NeRFs (Luo et al., 2023; Zhang et al., 2024; Jang et al., 2024; Song et al., 2024). Embedding watermark signals into the NeRF representation itself ensures persistent identification of models even under novel rendering viewpoints.

This paper introduces MultiNeRF, a novel framework for conditionally embedding *multiple* watermarks within a NeRF model. Existing NeRF watermarking techniques are limited to encoding a single watermark within the rendered image, typically with low capacity (*e.g.*, 16–48 bits). Even within the broader image watermarking literature Zhu et al. (2018); Tancik et al. (2020); Fernandez et al. (2022); Bui et al. (2023a) it is difficult to surpass capacities of this magnitude order (*i.e.*, < 100 bits), whilst maintaining acceptable visual quality for creative use cases. However, this is insufficient even to accommodate a URL *e.g.*, to an end-user license agreement. NeRF raises the intriguing possibility of embedding multiple distinct watermarks within a single model, each of which may bear only small capacities but which, taken together, can encode higher payload capacities. Further, multiple independently keyed watermarks admit scenarios requiring multiple licenses or stakeholders (Figure 1). For example, in collaborative environments such as co-developed metaverse worlds, different contributors may need distinct watermarks to establish ownership or track usage. Our technical contributions are:

1. **Watermark Grid**. We introduce a dedicated watermark grid alongside the existing geometry and appearance grids of the learned NeRF representation. This grid prevents the entanglement of the encoded watermark with scene content, improving watermark capacity and preserving rendering quality whilst adding only a small model size overhead.
2. **Conditional Modulation**. We introduce FiLM-based modulation Perez et al. (2018), applying a lightweight input embedding network to encode watermark-specific identifiers and dynamically control the activation of watermark features. This enables the conditional rendering of multiple watermarks within the same NeRF model.

Figure 3 illustrates our architecture to achieve multiple watermark embedding. We train the modified NeRF model end-to-end, using a pre-trained HiDDeN decoder (Zhu et al., 2018) as the base model for watermark retrieval. To ensure robustness, we augment training with differentiable noise sources. We later show (Sec. 5) our approach to deliver statistically significant improvements in capacity (*i.e.*, the ability to store many watermarks with high bit accuracy), without significant quality change. We demonstrate this for two standard NeRF datasets of multiple scenes (NeRF-Synthetic Mildenhall et al. (2021) and LLFF Mildenhall et al. (2019)), so extending NeRF watermarking into a more flexible paradigm suitable for real-world collaborative and commercial settings.

2 Related work

Visual Watermarking and Media Provenance. Traditional watermarking techniques embed information in the spatial Taha et al. (2022); Ghazanfari et al. (2011) or frequency domains Navas et al. (2008); Li & Wang (2007); Pevný et al. (2010). Deep watermarking methods such as HiDDeN Zhu et al. (2018), StegaStamp Tancik et al. (2020), RoSteALS Bui et al. (2023b), SSL Fernandez et al. (2022), and TrustMark Bui et al. (2023a) use learned embedding networks to improve imperceptibility and robustness against common transformations. Watermarking has been used to help trace digital content's provenance (including ownership and rights) in combination with cryptographic metadata standards *e.g.*, C2PA. These standards attach signed metadata to digital assets, but metadata is frequently removed during redistribution. Watermarking provides a complementary approach by embedding identifiers directly into content, allowing provenance to persist even when metadata is stripped Collomosse & Parsons (2024); MultiNeRF explores NeRF watermarking to achieve the same goal.

Watermarking NeRF models. While deep watermarking has been extensively studied for 2D images and videos, its application to 3D generative models, such as NeRFs, is relatively new. Conventional 2D watermarking fails to ensure persistence across novel viewpoints, leading to recent research into NeRF watermarking methods Luo et al. (2023); Zhang et al. (2024); Jang et al. (2024); Song et al. (2024). CopyRNeRF Luo et al. (2023) embeds watermark signals in the color feature field to ensure extraction from arbitrary viewpoints, while NeRFProtector Song et al. (2024) focuses on embedding watermark from the start of training a NeRF scene. WateRF Jang et al. (2024) introduces a frequency-based embedding, enhancing resilience to noise and compression. Existing NeRF watermarking methods are constrained by limited capacity and the inability to handle multiple watermarks, which we address with MultiNeRF.

Watermarking in Generative Models. With the rise of generative models in images Aditya et al. (2022), video Ho et al. (2022), and 3D asset creation Mildenhall et al. (2021), protecting model IP has become a key concern. Model provenance methods include watermarking of training data Asnani et al. (2024); Sablayrolles et al. (2020), model fingerprinting Zhang et al. (2021), and proactive tagging techniques Wang et al. (2021). Recent work in diffusion model watermarking, such as Stable Signature Fernandez et al. (2023), introduces an in-model watermarking approach that fine-tunes the decoder of a latent diffusion model to embed robust identifiers directly into generated images.

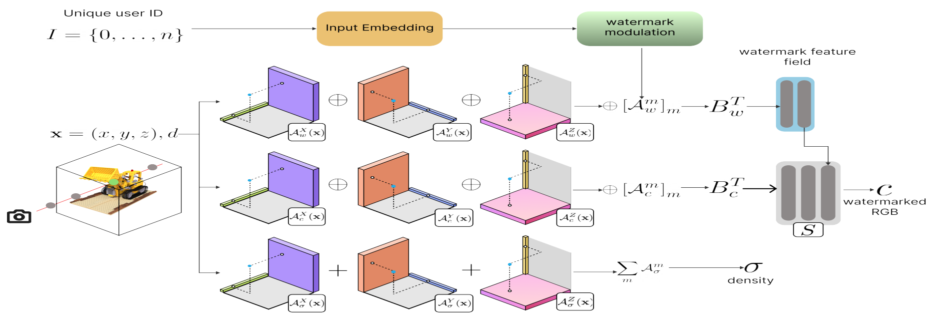
Unique user ID
$I = \{0, \dots, n\}$
Input Embedding
watermark modulation
watermark feature field
$\mathbf{x} = (x, y, z), d$
$\mathcal{A}^X_w(\mathbf{x})$
$\mathcal{A}^Y_w(\mathbf{x})$
$\mathcal{A}^Z_w(\mathbf{x})$
$\oplus [\mathcal{A}^m_w]_m$
B^T_w
$\mathcal{A}^X_c(\mathbf{x})$
$\mathcal{A}^Y_c(\mathbf{x})$
$\mathcal{A}^Z_c(\mathbf{x})$
$\oplus [\mathcal{A}^m_c]_m$
B^T_c
C
watermarked RGB
S
$\mathcal{A}^X_\sigma(\mathbf{x})$
$\mathcal{A}^Y_\sigma(\mathbf{x})$
$\mathcal{A}^Z_\sigma(\mathbf{x})$
$\sum_m \mathcal{A}^m_\sigma$
σ
density

Figure 2: MultiNeRF extends TensoRF by introducing a watermark grid A_w alongside the geometry A_σ and appearance A_c grids. Unique watermark IDs $(1..n)$ are first encoded via a learnable embedding network into compact vectors that are then transformed into per-channel scaling (γ) and shifting (β) parameters. These parameters modulate the watermark grid's features, ensuring that each distinct message is selectively activated and merged with the appearance grid during inference.

Other methods, such as Tree-Ring Wen et al. (2023), inject patterns into the noise initialization step of diffusion models to provide provenance guarantees. MultiNeRF extends these approaches to 3D by embedding provenance signals into NeRF representations, ensuring watermark persistence across novel views while enabling multiple uniquely keyed watermarks within a single model.

3 PRELIMINARIES (TENSORF)

This paper builds on TensoRF Chen et al. (2022), a popular explicit tensor-based representation for neural radiance fields (NeRFs). Unlike the traditional NeRF Mildenhall et al. (2021), which relies on multi-layered perceptions (MLPs), TensoRF Chen et al. (2022) represents a scene as a set of factorized tensors: the geometry grid, denoted by $G_\sigma \in \mathbb{R}^{I\times J\times K}$, which encodes the volume density σ at each voxel in the 3D grid; and the appearance grid $G_c \in \mathbb{R}^{I\times J\times K\times P}$, which encodes the view-dependent color c; where I, J, K represent the resolutions of the feature grid along the X, Y, Z axes. P denotes the number of appearance feature channels. Given a 3D location $\mathbf{x} = (x, y, z) \in \mathbb{R}^3$ and a view direction $\mathbf{d}$, by trilinear interpolation is applied to sample the two grids and the corresponding density σ and color c is estimated by:

$$\sigma, \mathbf{c} = (G_\sigma(\mathbf{x}), S(G_\mathbf{c}(\mathbf{x}), \mathbf{d})) \tag{1}$$

Where S is a decoding function, the decoding function can be either a small MLP or Spherical Harmonic (SH) function that covers the appearance features and view direction to an RGB color. The trilinearly interpolated grids G_σ and G_c are represented as:

$$G_\sigma(\mathbf{x}) = \sum_r \sum_m \mathcal{A}^m_{\sigma,r}(\mathbf{x}), \quad G_\mathbf{c}(\mathbf{x}) \quad = \mathbf{B}\left(\bigoplus \left[\mathcal{A}^m_{\mathbf{c},r}(\mathbf{x})\right]_{m,r}\right) \tag{2}$$

Where, $\mathcal{A}^m_{\sigma,r}(\mathbf{x})$ and $\mathcal{A}^m_{\mathbf{c},r}(\mathbf{x})$ are factorized components of the density and appearance tensors, indexed by mode m and rank r. $\mathbf{B}$ matrix acts as a global appearance dictionary that captures correlations across the scene. While $\bigoplus$ denotes the concatenation of tensor components.

TensoRF employs differentiable volume rendering. It integrates the radiance values along the rays cast through the scene, which samples Q points along the ray. Given a ray passing through a 3D point, the final pixel color C is computed as:

$$C = \sum_{q=1}^{Q} \tau_q \left(1 - \exp(-\sigma_q \Delta_q)\right) c_q \tag{3}$$

where, σ_q and c_q are the density and color of the sampled point x_q, Δ_q is the step size along the ray, and τ_q is the transmittance.

$$\tau_q = \exp\left(-\sum_{p=1}^{q-1} \sigma_p \Delta_p\right) \tag{4}$$

4 METHODOLOGY

Since the TensoRF decomposes a scene into two separate grids for geometry and appearance, Multi-NeRF extends this framework to encode a watermark with a further dedicated grid. In contrast to prior approaches that embed only a single watermark (*e.g.*, Jang et al. (2024); Song et al. (2024)), our method accommodates multiple distinct messages, each identified by a unique key (ID) which forms an additional model input. Figure 2 provides a high-level overview of the proposed pipeline.

4.1 CONSTRUCTING THE WATERMARKING MODULE

As NeRF models only take position and view direction as inputs to the model, we need an input embedding layer to condition the NeRF and embed the watermark accordingly, allowing it to switch between different watermark messages. We let $I(n)$ be the integer IDs, each corresponding to a distinct watermark message $M(n)$, and introduce a learnable embedding layer $Emb()$ which maps $I(n)$ to e_n a 16 dimensional message vector.

$$e_n = \text{Emb}(I(n)) \tag{5}$$

Where Emb is a learnable small MLP, and e_n is the embedding vector, which will serve as a condition to modulate and thereby select from multiple learned watermarks.

Watermark Grid. TensoRF (Chen et al., 2022) decomposes a scene into two explicit 3D grids: G_σ for geometry (density) and $G_\mathbf{c}$ for appearance (color). Directly embedding the watermark into the existing appearance grid $G_\mathbf{c}$ would risk entangling watermark features with scene color, potentially degrading both. We propose an additional grid $G_\mathbf{w}$, of the same spatial resolution $(I \times J \times K)$ but containing watermark-specific feature channels. Given, at any 3D location $\mathbf{x} = (x, y, z)$,

$$w(\mathbf{x}) = \bigoplus \left[\mathcal{A}^m_{\mathbf{w},r}(\mathbf{x})\right]_{m,r} \tag{6}$$

Watermark modulation. A design challenge in embedding multiple watermarks is ensuring that the model 'activates' only the selected watermark's features. To this end, we draw inspiration from FiLM (Perez et al., 2018); its FiLM layer can influence the feature space computation using feature-wise affine transform based on external information. In our context, as we want to embed multiple watermarks, our external information is the embedded watermark ID, i.e., e_n. Our watermark modulation has a modulator network, which in our case is a learnable linear layer $Mod()$, which takes e_n as input and outputs a pair of modulation parameters to scale and shift the feature space of the watermark features: scale γ_n and shift β_n:

$$[\gamma_n, \beta_n] = \text{Mod}(\mathbf{e}_n), \tag{7}$$

Thus, each $w(x)$ channel is conditioned based on the embedding e_n. We then compute:

$$\mathbf{w}'(\mathbf{x}) = \gamma_n \odot \mathbf{w}(\mathbf{x}) + \beta_n, \tag{8}$$

Where $\odot$ is the Hadamard product. The resulting $\mathbf{w}'(\mathbf{x})$ represents the *modulated watermark features* specific to $I(n)$. Intuitively, γ_n and β_n activate certain dimensions of the watermark grid differently for each watermark ID.

Merging watermark and appearance. To incorporate these modulated watermark features with appearance, we pass only $G_c(\mathbf{x})$ and viewing direction to the S MLP, which is a color decoding function of TensoRF (see Chen et al. (2022) for details), and we apply the modulated watermark features to the last linear layer of the S MLP:

$$\tilde{\mathbf{c}}(\mathbf{x}) = S(\mathbf{G}_c(\mathbf{x}), \mathbf{w}'(\mathbf{x}), \mathbf{d}), \tag{9}$$

where $\tilde{\mathbf{c}}(\mathbf{x})$ is now a watermark color at location $\mathbf{x}$. The rest of TensoRF's volume rendering proceeds unchanged. Because each ID yields different $\{\gamma_n, \beta_n\}$,, the final color $\tilde{\mathbf{c}}(\mathbf{x})$ implicitly contains an ID-specific watermark pattern. As a result, any novel view rendered from the watermarked NeRF model will contain an embedded signature that can be extracted with the watermark decoder.

Differentiable Augmentation layer. To promote robustness against image attacks, we train with *differentiable augmentations* on the fully rendered image. Each image is transformed by a small set of augmentations (brightness, contrast, color jiggle, gaussian blur, gaussian noise, hue, posterize, RGB shift, saturation, median blur, box blur, motion blur, sharpness, and differentiable JPEG compression) before passing them into the watermark decoder function. Formally, if $\tilde{\mathbf{I}}$ is the full rendered image, we apply a random pair of augmentations from $\mathbf{A}$ from a set of $\{A_1, ..., A_m\}$.:

$$\hat{\mathbf{I}} = \mathcal{A}(\tilde{\mathbf{I}}), \tag{10}$$

where $\hat{\mathbf{I}}$ is an augmented image, and we pass this image to the watermark decoder.

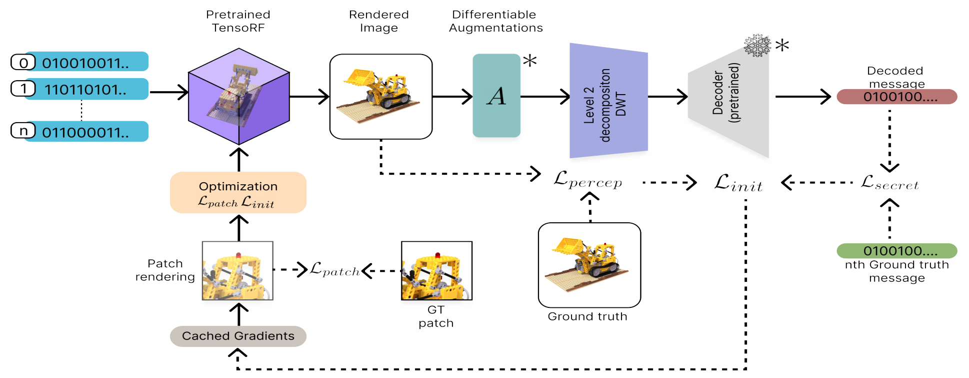

Figure 3: MultiNeRF training process A learnable encoder transforms multiple distinct watermark IDs into compact embedding vectors. TensoRF is trained with a HiDDEN decoder module end-to-end with rendered images passing through differentiable noise augmentations. Perceptual (12) and patch-based (13) reconstruction losses balance visual quality against a message loss (11).

4.2 Training MultiNeRF to embed watermarks

We begin by training a TensoRF NeRF model, using the geometry and appearance grids to initialize those parts of our MultiNeRF architecture. We perform a separate initialization of the watermark decoder training a HiDDeN decoder following (Zhu et al., 2018). Other parts of our architecture are initialized with white noise, and training proceeds in two phases.

Phase 1. We begin by first freezing the watermark decoder D. We generate a random watermark ID $I(n)$, with $\mathbf{m}_n$ the ground truth message for $n-th$ ID. For each training iteration, we render a full-resolution image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$. As directly injecting watermarks into the spatial domain can be vulnerable to image compression, we apply a level 2 decomposition of Discrete Wavelet Transform (DWT) on the rendered image (after Jang et al. (2024)) and use the LL_2 sub-band as it retains the majority of the energy and broad structural features. This sub-band image (denoted $\mathbf{I}_{LL_2}$) is then passed to the watermark decoder D obtaining a decoded message $\mathbf{m}'_n$. A BCE loss is calculated between secrets $\mathbf{m}_n$ and $\mathbf{m}'_n$:

$$\mathcal{L}_{secret} = \text{BCE}(\mathbf{m}_n, \mathbf{m}'_n) \tag{11}$$

Alongside the secret loss $\mathcal{L}_{secret}$, we also calculate a perceptual loss $\mathcal{L}_{percept}$ via Watson-VGG (Czolbe et al., 2020) to create a total loss for this first phase of training:

$$\mathcal{L}_{init} = \lambda_m \mathcal{L}_{secret} + \lambda_i \mathcal{L}_{percept} \tag{12}$$

Phase 2. Training continues end-to-end via a patch-based process using deferred backpropagation (Zhang et al., 2022). As optimizing at a full resolution in each iteration can be memory-intensive. Therefore, we make patches and re-render the images from those patches. A patch loss $\mathcal{L}_{patch}$ is then calculated, which uses the RGB loss across the rendered pixels, SSIM loss, and total variation regularization:

$$\mathcal{L}_{patch} = \lambda_{rgb} \mathcal{L}_{RGB} + \lambda_{TV} \mathcal{L}_{TV} + \lambda_{SSIM} \mathcal{L}_{SSIM} \tag{13}$$

We unfreeze the decoder D and introduce the *differentiable augmentations* A at each iteration on the rendered image, on which the 2-level DWT is applied and is passed into the watermark decoder D.

5 Experiments and Discussion

5.1 Experimental Setup

Datasets: We train and evaluate our method using the Nerf-Synthetic Mildenhall et al. (2021) (hereafter, SYN) and the LLFF datasets Mildenhall et al. (2019). SYN consists of eight representative scenes: *chair, drums, ficus, hotdog, lego, materials, mic, ship*; and the LLFF dataset consists of eight scenes: *fern, flower, fortress, horns, leaves, orchids, room, trex*. In all evaluations, we train and test MultiNeRF using the partitions of those public datasets.

Baselines: As no multiple watermark frameworks are available for NeRFs, we evaluate our proposed method against two state-of-the-art NeRF watermarking methods for the single watermarking task:

WateRF (Jang et al., 2024) and NeRFProtector (Song et al., 2024). For fair comparisons, we use the models that those baselines have implemented in their methods; for WateRF, we choose TensoRF (Chen et al., 2022) as the NeRF model, and for NeRFProtector (Song et al., 2024) we use Instant-NGP (Müller et al., 2022). As no prior work embeds multiple watermarks into a single NeRF, we tune WateRF (Jang et al., 2024) by adding an input embedding layer to condition it to embed multiple watermarks (denoted 'WateRF-modified').

Metrics: We use bit accuracy to evaluate the accuracy of the decoder for a given capacity. For visual quality, we measure PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) distances between the ground truth and watermarked image.

Training setup: MultiNeRF is implemented in PyTorch and trained with a batch size of 1 on a single NVIDIA RTX 4080. The ADAM optimizer (Kingma, 2014) is used with $\beta_1 = 0.9$ and $\beta_2 = 0.99$, and an exponential learning rate decay is applied. All the grid parameters are optimized with an initial learning rate of 0.02, and the basis matrix for all grids has a learning rate of $1e - 3$.

5.2 SINGLE WATERMARK EMBEDDING EVALUATION

We evaluate two variants of our model trained with (MultiNeRF-Noised) and without (MultiNeRF) noise augmentations. All methods embed a single 48-bit message. As the quality is slightly affected by different messages, we average over 50 sets of unique watermarks and report the average results for all the synthetic and LLFF datasets. We enforce a minimum Hamming distance between the watermark messages to ensure embedded messages are distinct. Note that NeRFProtector reported on only 3 scenes from SYN and LLFF datasets (Song et al., 2024). Using the official implementation, we cannot reconstruct 3 of the full set of scenes (ficus, flower, leaves) and omit these outliers from their average.

Table 1 shows that MultiNeRF achieves a mean bit accuracy of 93.18% on SYN outperforming both WateRF (91.51%) and NeRFProtector (90.81%); and like WaterRF saturates performance on LLLF at $\sim 99\%$. Whilst the noised variant (MultiNeRF-Noised) exhibits slightly lower bit accuracies in some scenes, this is traded for improved robustness (c.f. subsec.5.4). MultiNeRF generally achieves comparable quality and slightly higher accuracy at the single watermarking task.

Table 1: Comparing raw bit accuracy ↑ (no error correction) of the proposed method (MultiNeRF) to baseline methods (NeRFProtector, WaterRF) for the single message task on datasets: SYN (upper) and LLFF datasets (lower). Values to 2 d.p.

Method (on SYN)	Avg.	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
WateRF (Jang et al., 2024)	91.51	98.31	92.19	79.83	96.21	93.16	82.33	95.92	94.10
NeRFProtector (Song et al., 2024)	90.81	96.41	89.73	-	93.47	90.12	84.05	90.39	91.54
MultiNeRF (ours)	93.18	98.35	95.14	83.06	96.97	94.86	85.16	96.89	95.03
MultiNeRF-Noised (ours)	89.70	92.60	93.61	78.60	94.36	92.49	83.54	89.72	92.65
Method (on LLFF)	Avg.	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	Trex
WateRF (Jang et al., 2024)	99.32	99.75	99.56	99.95	99.92	99.53	96.07	99.89	99.91
NeRFProtector (Song et al., 2024)	95.73	94.68	-	99.58	98.77	-	82.23	99.73	99.37
MultiNeRF (ours)	99.23	99.39	99.48	99.82	99.87	99.68	95.92	99.77	99.88
MultiNeRF-Noised (ours)	98.55	99.04	99.05	99.90	99.86	99.28	91.81	99.65	99.81

Table2 shows that quality is equivocal for the baselines versus MultiNeRF, which attains an average PSNR of 30.83 dB for SYN and 26.78 dB for the LLFF dataset. We are slightly higher (SSIM, PSNR) or lower (LPIPS) than our baselines on some metrics. Overall, we conclude that MultiNeRF is comparable in quality and slightly outperforms bit accuracy for the single watermarking task.

5.3 EVALUATING MULTIPLE WATERMARK EMBEDDING

We evaluate the model setting of embedding multiple distinct watermarks within a single NeRF model. Since all baselines target a single watermark per model, we have adapted WateRF into a accepting an input variable to select on watermark and trained it as a baseline (WateRF-modified). Each watermark has a 16-bit capacity, and we embed multiple watermarks into a single model.

Figure 6 presents the average bit accuracy across the SYN and LLFF datasets for varying numbers of unique watermarks per model. For SYN, WateRF-modified returns a random response (50% bit accuracy) for 32 watermarks and onwards versus MultiNeRF at 70% bit accuracy on 32 watermarks, which degrades to a random response at 64 watermarks. Similarly, for LLFF, we observe that both methods start with similar bit accuracies at 2 watermarks, with WateRF-modified dropping to random response at 32 watermarks, whilst MultiNeRF achieves 82% bit accuracy at 32 watermarks

Table 2: Comparing visual quality metrics (LPIPS ↓, PSNR ↑, SSIM ↑) of MultiNeRF to baselines (NeRFProtector, WateRF) for single message task on SYN and LLFF datasets. Values to 2 d.p.

LPIPS (Lower is better)

Method (on SYN)	Avg.	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
WateRF	0.04	0.02	0.05	0.02	0.03	0.02	0.04	0.02	0.08
NeRFProtector	0.08	0.04	0.07	-	0.08	0.03	0.08	0.05	0.19
MultiNeRF (ours)	0.04	0.02	0.05	0.02	0.04	0.02	0.04	0.02	0.08
MultiNeRF-Noised (ours)	0.04	0.02	0.06	0.03	0.04	0.02	0.04	0.03	0.09

Method (on LLFF)	Avg.	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	Trex
WateRF	0.10	0.13	0.09	0.07	0.08	0.12	0.17	0.06	0.06
NeRFProtector	0.07	0.10	-	0.07	0.15	-	0.08	0.05	0.06
MultiNeRF	0.09	0.14	0.09	0.06	0.08	0.12	0.18	0.05	0.07
MultiNeRF-Noised (ours)	0.10	0.14	0.09	0.07	0.08	0.12	0.17	0.08	0.06

PSNR (Higher is better)

Method (on SYN)	Avg.	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
WateRF	30.58	32.33	25.30	31.04	33.61	32.74	28.45	31.94	29.30
NeRFProtector	28.44	31.40	24.41	-	32.51	31.00	26.20	30.61	23.00
MultiNeRF (ours)	30.83	32.81	25.58	31.22	34.12	32.83	28.15	32.05	29.87
MultiNeRF-Noised (ours)	30.61	32.59	25.38	30.64	34.07	33.14	28.16	31.35	29.53

Method (on LLFF)	Avg.	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	Trex
WateRF	26.31	25.08	27.60	30.26	27.88	20.96	19.98	31.30	27.43
NeRFProtector	26.08	27.58	-	28.13	23.05	-	20.19	30.73	26.83
MultiNeRF (ours)	26.78	24.99	27.53	30.20	28.15	20.97	19.99	34.68	27.70
MultiNeRF-Noised (ours)	26.36	24.89	27.45	30.45	28.05	20.97	19.96	31.48	27.65

SSIM (Higher is better)

Method (on SYN)	Avg.	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
WateRF	0.94	0.96	0.93	0.97	0.96	0.96	0.93	0.97	0.87
NeRFProtector	0.91	0.95	0.90	-	0.94	0.95	0.90	0.96	0.78
MultiNeRF (ours)	0.95	0.97	0.93	0.97	0.96	0.96	0.93	0.97	0.88
MultiNeRF-Noised (ours)	0.94	0.97	0.93	0.96	0.96	0.97	0.93	0.97	0.88

Method (on LLFF)	Avg.	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	Trex
WateRF	0.82	0.80	0.83	0.87	0.87	0.73	0.64	0.93	0.90
NeRFProtector	0.85	0.87	-	0.87	0.77	-	0.74	0.91	0.89
MultiNeRF (ours)	0.82	0.80	0.84	0.88	0.88	0.73	0.64	0.94	0.91
MultiNeRF-Noised (ours)	0.83	0.80	0.84	0.88	0.89	0.73	0.64	0.94	0.91

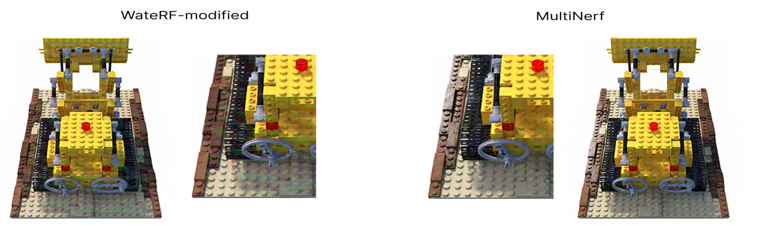

Figure 4: Visual artifacts (colored ripples) present in WateRF-modified versus MultiNeRF.

and 68% at 64 watermarks. The error bars confirm a significant bit accuracy improvement using MultiNeRF for the multiple watermarking task (*i.e.*, $p < 0.0001$ for both SYN and LLFF).

Figure 5 shows all methods performed with similar visual quality, and the error bars indicate no statistically significant difference in quality between the methods. Table 3 illustrates the trade-off between model size and performance for 16 watermarks. Compared to TensoRF (Chen et al., 2022), introducing a watermark grid increases parameter count and storage overhead by $\sim 12\%$.

5.4 Evaluating watermark robustness

Robustness evaluation was performed on the SYN/Lego scene, and results averaged for 50 different watermarks embedded into the model. Figure8, plots bit accuracy vs the types of attacks used i.e. cropping, blur, JPEG compression, etc. The MultiNeRF-Noised model has greater bit accuracy on these attacks vs. baselines and a 3% bit lower accuracy than MultiNeRF.

5.5 Ablation Study

We present ablation experiments to validate the design choices in MultiNeRF (Table 3) conducted on the Lego scene with 16 watermarks per model. First, we remove the proposed watermark grid

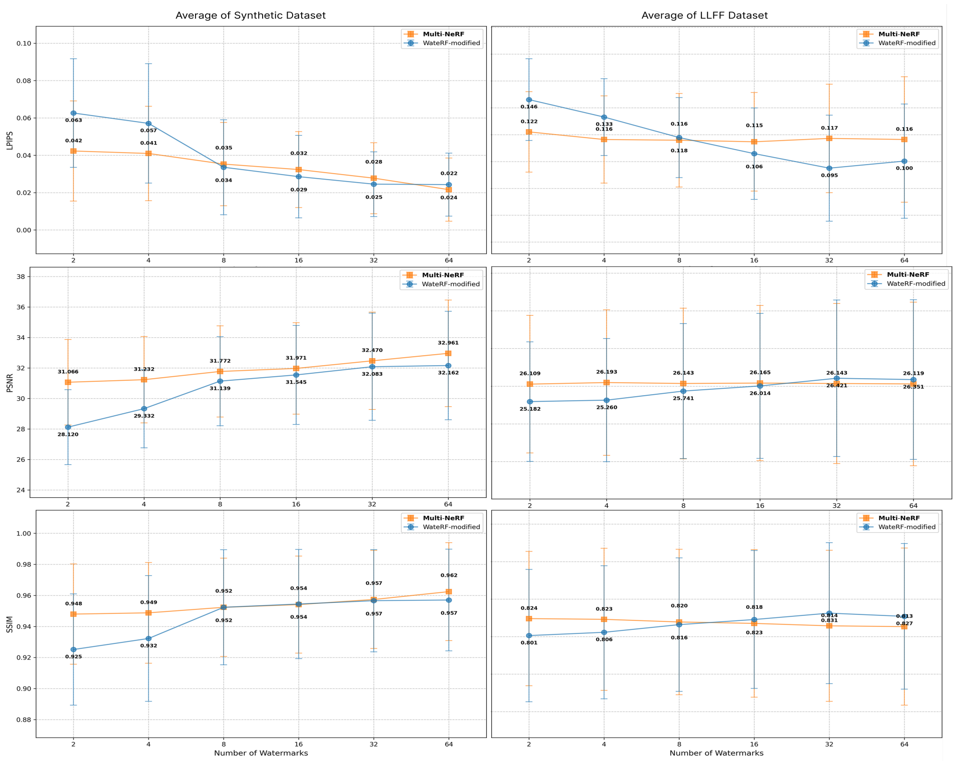

Figure 5: Evaluating the visual quality of MultiNeRF vs. baseline WateRF-modified for the multi-watermarking task: LPIPS (top); PSNR (mid.); SSIM (bot.).

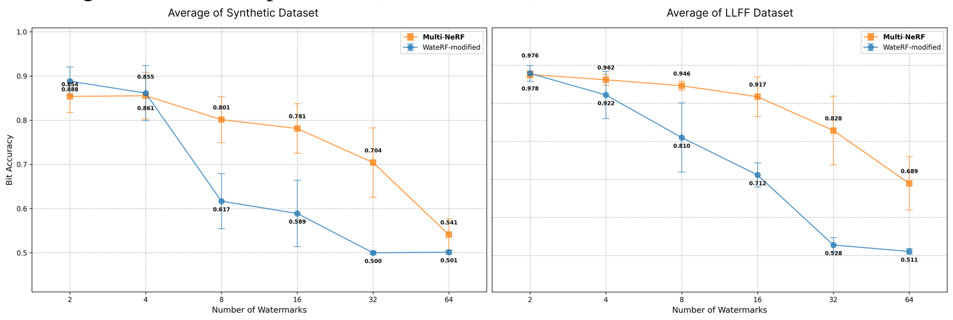

Figure 6: Evaluating bit accuracy ↑ of MultiNeRF versus baseline WateRF-modified for the multi-watermarking task. Accuracies averaged for SYN (left) and LLFF (right) datasets. Performance is significantly higher for MultiNeRF beyond the single watermarking case.

(-GRID), which falls back to an MLP to encode the watermark information. Without a separate spatial grid, bit accuracy drops to 51.60%, suggesting that an explicit grid provides higher capacity for multiple watermarks. We also examine the effect of removing modulation (-FiLM), causing a drop from 82.79 to 70 %. We explored a training simplification where the watermark grid is learned without adjusting the appearance or geometry grid (G_σ frozen), showing that bit accuracy suffers when we do not allow the appearance parameters to adapt. Another variation injects the FiLM-modulated features into earlier layers (+EARLY) of MLP rather than the last layer. While the bit accuracy (78.04%) remains high, visual quality suffers *e.g.*, as with WateRF-modified, scene reflections are absent.

6 Conclusion

We presented **MultiNeRF**, the first NeRF watermarking technique to embed multiple conditional watermarks simultaneously within a NeRF (TensoRF) model. MultiNeRF achieves competitive

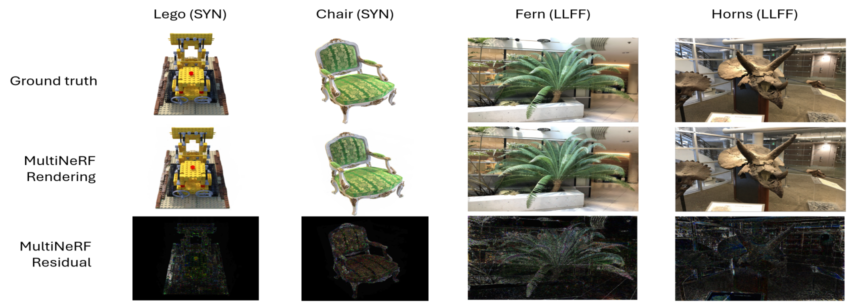

Figure 7: Visualizing the imperceptible watermark residual (bottom, amplified) introduced by Multi-NeRF between the ground truth (top) and watermarked image (middle).

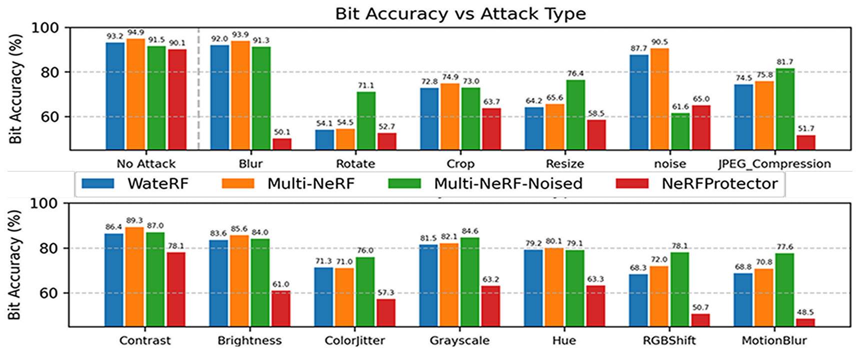

Figure 8: Evaluating the robustness to various degrading transformations for the proposed Multi-NeRF method trained with (-noised) and without noise augmentations to baselines WateRF and NeRFProtector for watermarking a single message (averaged for 50 runs over Lego).

Table 3: Left: Ablation Study of MultiNeRF (abbrev. MN), evaluating the impact of key components. Right: Comparison of performance gain (accuracy) versus size increase. Figures to 2.d.p.

Method	Bit Acc. ↑	PSNR ↑	LPIPS ↓	SSIM ↑
MN(-GRID)(+MLP)	51.60	35.50	0.01	0.98
MN(-FiLM)(-GRID)	66.28	33.57	0.02	0.97
MN(-FiLM)	70.00	34.69	0.01	0.97
MN G_σ frozen	67.72	34.56	0.01	0.97
MN(+EARLY)	78.04	34.29	0.02	0.97
MN(-GRID)	66.35	33.79	0.02	0.97
MN (ours)	**82.79**	34.03	0.01	0.97

Method	Bit Acc. ↑	Size (MB) ↓	Params (M) ↓
TensoRF	-	68.6	17.75
MN-GRID-FiLM	66.28	68.8	17.76
MN (ours)	**82.79**	77.1	19.98

watermark embedding performance at single message watermarking while introducing the novel capability of multiple watermark support. This increases effective model capacity because the watermark grid enables multiple messages to be simultaneously encoded without degrading the visual quality. FiLM-based modulation enables selection between the distinct watermarks at rendering time. Future work could explore learnable perceptual metrics or NeRF-specific benchmarks to help better quantify artifacts introduced by watermark embedding. As NeRFs continue to evolve and find new applications, frameworks like MultiNeRF will play an essential role in securing the intellectual property rights of 3D content creators, and their integration with emerging media provenance standards presents another future direction.

Acknowledgement

This research was funded by the UKRI Grant EP/T022485/1.

REFERENCES

Ramesh Aditya, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Vishal Asnani, John Collomosse, Tu Bui, Xiaoming Liu, and Shruti Agarwal. Promark: Proactive diffusion watermarking for causal attribution. In *CVPR*, 2024.

Tu Bui, Shruti Agarwal, and John Collomosse. Trustmark: Universal watermarking for arbitrary resolution images. *arXiv preprint arXiv:2311.18297*, 2023a.

Tu Bui, Shruti Agarwal, Ning Yu, and John Collomosse. Rosteals: Robust steganography using autoencoder latent space. In *Proc. CVPR WMF*, pp. 933–942, 2023b.

Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European conference on computer vision*, pp. 333–350. Springer, 2022.

John Collomosse and Andy Parsons. To authenticity, and beyond! building safe and fair generative ai upon the three pillars of provenance. *IEEE Computer Graphics and Applications*, 44(3):82–90, 2024.

Steffen Czolbe, Oswin Krause, Ingemar Cox, and Christian Igel. A loss function for generative neural networks based on watson's perceptual model. *Advances in Neural Information Processing Systems*, 33:2051–2061, 2020.

Lidia Fabra, J Ernesto Solanes, Adolfo Muñoz, Ana Martí-Testón, Alba Alabau, and Luis Gracia. Application of neural radiance fields (nerfs) for 3d model representation in the industrial metaverse. *Applied Sciences*, 14(5):1825, 2024.

Pierre Fernandez, Alexandre Sablayrolles, Teddy Furon, Hervé Jégou, and Matthijs Douze. Watermarking images in self-supervised latent spaces. In *Proc. ICASSP*, pp. 3054–3058. IEEE, 2022.

Pierre Fernandez, Guillaume Couairon, Hervé Jégou, Matthijs Douze, and Teddy Furon. The stable signature: Rooting watermarks in latent diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22466–22477, 2023.

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

Kazem Ghazanfari, Shahrokh Ghaemmaghami, and Saeed R Khosravi. Lsb++: An improvement to lsb+ steganography. In *TENCON 2011-2011 IEEE Region 10 Conference*, pp. 364–368. IEEE, 2011.

Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Imagen video: High definition video generation with diffusion models. In *NeurIPS*, 2022.

Youngdong Jang, Dong In Lee, MinHyuk Jang, Jong Wook Kim, Feng Yang, and Sangpil Kim. Waterf: Robust watermarks in radiance fields for protection of copyrights. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12087–12097, 2024.

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Chaojian Li, Sixu Li, Yang Zhao, Wenbo Zhu, and Yingyan Lin. Rt-nerf: Real-time on-device neural radiance fields towards immersive ar/vr rendering. In *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, pp. 1–9, 2022.

Xiaoxia Li and Jianjun Wang. A steganographic method based upon jpeg and particle swarm optimization algorithm. *Information Sciences*, 177(15):3099–3109, 2007.

Ziyuan Luo, Qing Guo, Ka Chun Cheung, Simon See, and Renjie Wan. Copyrnerf: Protecting the copyright of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22401–22411, 2023.

Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (ToG)*, 38(4):1–14, 2019.

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

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

K. A. Navas, Mathews Cheriyan Ajay, M. Lekshmi, Tampy Archana, and M. Sasikumar. DWT-DCT-SVD based watermarking. In *COMSWARE'08*, pp. 271–274. IEEE, 2008.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018.

Tomáš Pevnỳ, Tomáš Filler, and Patrick Bas. Using high-dimensional image models to perform highly undetectable steganography. In *Proc. Int. Conf. Information Hiding*, pp. 161–177. Springer, 2010.

Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, and Hervé Jégou. Radioactive data: Tracing through training. *International Conference on Machine Learning (ICML)*, 2020.

Qi Song, Ziyuan Luo, Ka Chun Cheung, Simon See, and Renjie Wan. Protecting nerfs' copyright via plug-and-play watermarking base model. In *European Conference on Computer Vision*, pp. 57–73. Springer, 2024.

Mustafa Sabah Taha, Mohd Shafry Mohd Rahem, Mohammed Mahdi Hashim, and Hiyam N Khalid. High payload image steganography scheme with minimum distortion based on distinction grade value method. *Multimedia Tools and Applications*, 81(18):25913–25946, 2022.

Matthew Tancik, Ben Mildenhall, and Ren Ng. Stegastamp: Invisible hyperlinks in physical photographs. In *Proc. CVPR*, pp. 2117–2126, 2020.

Shan Wang, Ting Liu, and Yizhong Wang. Faketagger: Automated fake news detection using proactive watermarks. In *NeurIPS Workshop on AI for Social Good*, 2021.

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.

Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-ring watermarks: Invisible fingerprints for diffusion images. In *Advances in Neural Information Processing Systems*, 2023.

Kai Zhang, Nick Kolkin, Sai Bi, Fujun Luan, Zexiang Xu, Eli Shechtman, and Noah Snavely. Arf: Artistic radiance fields. In *European Conference on Computer Vision*, pp. 717–733. Springer, 2022.

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*, pp. 586–595, 2018.

Ruier Zhang. Moving museums into the metaverse. *Science and Technology of Engineering, Chemistry and Environmental Protection*, 1(5), 2024.

Xuanyu Zhang, Jiarui Meng, Runyi Li, Zhipei Xu, Jian Zhang, et al. Gs-hider: Hiding messages into 3d gaussian splatting. *Advances in Neural Information Processing Systems*, 37:49780–49805, 2024.

Yifan Zhang, Fangchang Wu, Wei Yan, Yun Fu, Si Wu, Xiaojun Wang, and Xiaolei Zhang. Deepmarks: A secure fingerprinting framework for digital content protection. *IEEE Transactions on Information Forensics and Security*, 16:2700–2715, 2021.

Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks. In *Proc. ECCV*, pp. 657–672, 2018.