Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.051608

ARTICLE

MarkNeRF: Watermarking for Neural Radiance Field

Lifeng Chen[1,2], Jia Liu[1,2,*], Wenquan Sun[1,2], Weina Dong[1,2] and Xiaozhong Pan[1,2]

[1]Cryptographic Engineering Department, Institute of Cryptographic Engineering, Engineering University of PAP, Xi'an, 710086, China

[2]Key Laboratory of Network and Information Security of PAP, Xi'an, 710086, China

*Corresponding Author: Jia Liu. Email: liujia1022@gmail.com

Received: 10 March 2024 Accepted: 01 June 2024 Published: 18 July 2024

ABSTRACT

This paper presents a novel watermarking scheme designed to address the copyright protection challenges encountered with Neural radiation field (NeRF) models. We employ an embedding network to integrate the watermark into the images within the training set. Then, the NeRF model is utilized for 3D modeling. For copyright verification, a secret image is generated by inputting a confidential viewpoint into NeRF. On this basis, design an extraction network to extract embedded watermark images from confidential viewpoints. In the event of suspicion regarding the unauthorized usage of NeRF in a black-box scenario, the verifier can extract the watermark from the confidential viewpoint to authenticate the model's copyright. The experimental results demonstrate not only the production of visually appealing watermarks but also robust resistance against various types of noise attacks, thereby substantiating the effectiveness of our approach in safeguarding NeRF.

KEYWORDS

Neural radiation field; 3D watermark; robustness; black box

1 Introduction

Neural radiance field (NeRF) [1] is a technique employed for generating high-quality 3D reconstruction models. It utilizes neural networks to learn a continuous function mapping spatial coordinates to density and color, enabling the synthesis of novel viewpoints. NeRF has garnered significant attention in computer vision research [2–5]. It is foreseeable that, similar to the sharing of 2D images and videos, future trends will involve the online sharing of 3D content. Research on copyright protection for NeRF is currently limited, and training on NeRF has consistently posed a significant challenge. Consequently, safeguarding the copyright of NeRF models has emerged as an important and pressing issue.

Common copyright protection technologies include data encryption [6,7], digital watermarking [8–10], digital signatures [11], etc. However, since NeRF models need to be displayed to users and require a certain degree of robustness, data encryption and digital signature technologies are not applicable. Therefore, we have focused our discussion on digital watermarking. Digital watermarking embeds a copyright identifier into digital media through embedding algorithms. When a copyright

dispute arises, the copyright owner can extract the watermark information from the media through the inverse operation of the embedding algorithm to confirm copyright ownership. Traditional watermarking algorithms [12] mainly rely on specific mathematical functions to modify the media for embedding watermarks. However, traditional algorithms often fail to achieve a good balance between imperceptibility, robustness, and watermark capacity. With the application of deep learning technology in the field of watermarking, the embedding and extraction methods of watermarks no longer require manually designing complex mathematical functions and demonstrate good performance. In deep learning watermarking [13], copyright owners embed watermark information into carrier images through encoders and extract watermark information from images containing watermarks processed through noise layers through decoders. The embedding and extraction process is close to a black box. However, although existing deep learning-based watermarking algorithms exhibit strong robustness, excellent imperceptibility, and large embedding capacity, most watermark algorithms are designed for multimedia data such as images, sound, and video, lacking research on watermark algorithms for implicit data like NeRF. One intuitive solution is to directly embed watermarks into samples rendered by NeRF models using existing watermark methods. However, this method only protects the copyright of rendered samples, not the NeRF model itself. If the NeRF model is stolen, malicious users may generate new samples using new rendering methods, making this method unsuitable for protecting 3D model copyrights. Traditional 3D data are primarily represented in the form of point clouds [14], voxels [15], or triangular meshes [16]. Copyright protection strategies for these types of 3D data generally fall into three categories: directly embedding watermarks by translating, rotating, or scaling 3D shapes [17]; modifying 3D model parameters to embed watermarks [18]; and employing deep learning techniques for watermark embedding [19]. However, NeRF models do not have specific structural information, so these methods cannot protect the copyright of NeRF.

Li et al. [20] established a connection between information hiding and NeRF, proposing the StegaNeRF scheme for information embedding. This approach involves training a standard NeRF model, which is subsequently utilized as a generator for generating new viewpoints. During the training process of the message extraction network, the NeRF network is trained twice to ensure accurate message extraction from the 2D images rendered from the StegaNeRF network. Additionally, Luo et al. [21] introduced the CopyRNeRF scheme, which employs watermark colors as a substitute for the original colors in NeRF to protect the model's copyright. Furthermore, a distortion-resistant rendering scheme was designed to ensure robust information extraction in the 2D rendering of NeRF. This method directly safeguards the copyright of NeRF models while maintaining high rendering quality and bit precision. However, both of the aforementioned methods involve secondary training of NeRF, incurring substantial training costs.

To address the issue of needing to retrain the NeRF model for copyright protection purposes, in this paper we propose a novel watermarking algorithm tailored for NeRF. We employ a conventional approach, utilizing an embedding network to embed the watermark into the images within the training set. Then, we utilized the NeRF model for 3D modeling. Copyright validation is achieved by the generation of a secret image from a confidential viewpoint using the NeRF model, followed by the design of a watermark extractor using neural network overperparameterization techniques to extract the embedded watermark from this image. In a black-box scenario [22], when suspicion arises regarding the unauthorized use of 3D models, the verifier can extract the watermark from the confidential viewpoint to authenticate the model's copyright. In order to fortify the model's robustness, a noise layer was incorporated during the optimization process to achieve anti-distortion rendering. In the event of a malicious theft of the model, even when attackers employ diverse rendering methods or process the

rendered image, the copyright verifier retains the capability to extract watermark information from the model. This paper makes the following contributions:

1. Our proposal presents a novel watermarking scheme for NeRF, taking advantage of its ability to generate new perspective images and utilize perspective information as the key. The security of the watermark algorithm is ensured by the continuity of perspective synthesis and the large key space.
2. To implement the watermarking scheme, we rely on traditional watermarking techniques and eliminate the requirement for secondary NeRF training. By training a simple extraction network, we extract watermarks from a specific perspective of the model.
3. To achieve robustness, we introduced a noise layer during the training process of the NeRF model, achieving anti distortion rendering.

2 Proposed Method

This section describes the application scenarios of our algorithm and the specific details of algorithm implementation. Existing watermarking schemes for neural radiance fields require secondary training of the model, and the quality of watermark extraction is not high. Therefore, this paper proposes a black-box watermarking scheme for neural radiance fields, where we directly embed watermark information into the NeRF model without the need for secondary training of the model. By leveraging the ability of the NeRF model to synthesize new viewpoints, we use viewpoint information as a key and train an extractor to extract watermark information from secret viewpoint images. In real-world scenarios, when our model is stolen, we can prove the model's copyright by extracting the watermark information using the key.

2.1 Application Scenarios

Algorithm 1: Typical application scenario of MarkNeRF

1: Alice obtains a set of images depicting a 3D scene and trains MarkNeRF M to embed watermark information.
2: Alice openly shares Model M on the Internet for others to enjoy the 3D scene.
3: Bob obtained model M without Alice's permission and uploaded it to the network in his own name.
4: After discovering that Bob had published model M, Alice used a secret perspective to extract the watermark. This is to verify that Alice is the legitimate owner of Model M, not Bob.
5: Bob's unauthorized publication constitutes copyright infringement, therefore the infringing content must be withdrawn.

The specific process of the application scenario is shown in Fig. 1.

2.2 Algorithm Process

This approach comprises five key stages: watermark embedding, MarkNeRF training, noise layer processing, watermark extraction, and copyright verification. Fig. 2 shows the process of our scheme. The process begins with the embedding of the watermark $\mathbf{W}$ into the original image set $\{Y_i\}$, achieved through the use of an embedding network G_ϕ, resulting in the generation of the image set $\left\{\hat{Y}_i\right\}$ with an embedded watermark. Subsequently, MarkNeRF M is trained with $\left\{\hat{Y}_i\right\}$ and its corresponding camera pose $\{P_i\}$. To prevent attackers from creating pirated models by processing and retraining the rendered images after obtaining our model. During the training process, we introduce a noise

layer N for processing and perform noise processing on the rendered image set $\{H_i\}$ to obtain $\left\{\hat{H}_i\right\}$. Simultaneously, the copyright verifier selects a secret camera perspective P_s as the key, renders the secret image $\mathbf{S}$ through M, and subsequently trains an extractor D_φ to recover the watermark image $\hat{\mathbf{W}}$. The model parameters of D_φ are saved, and MarkNeRF M is uploaded to the social cloud. Finally, during the copyright verification stage, the copyright verifier renders image $\mathbf{S}$ from M using the key P_s, inputs it into extractor D_φ, and obtains the watermark image $\hat{\mathbf{W}}$ to verify the copyright of model M.

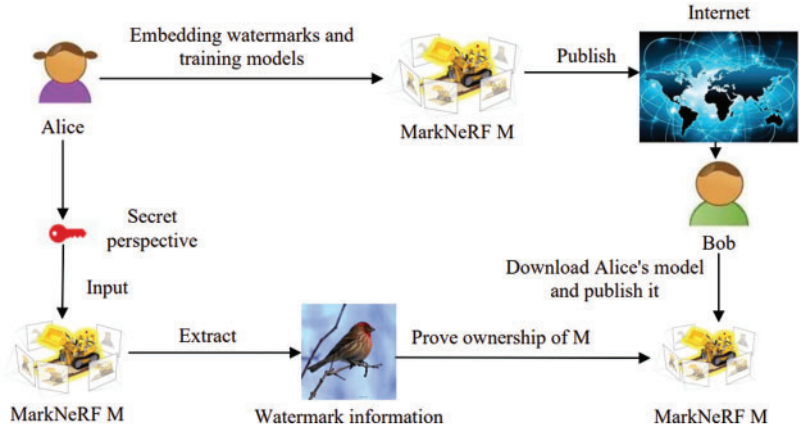


Figure 1: Application scenario process. The copyright owner embeds watermark information into the MarkNeRF model before publishing it on the Internet. In the event of malicious model theft, the copyright owner can extract the watermark from the model using the designated key

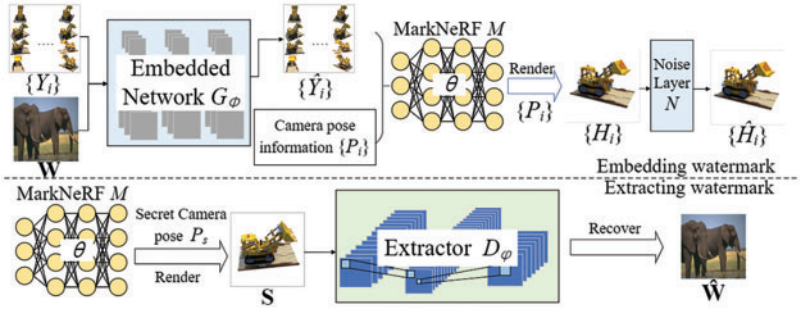


Figure 2: Algorithm process of MarkNeRF. The network is embedded to generate a set of watermarked images, after which the MarkNeRF model and extractor are trained. The attack layer is designed to simulate various types of noise attacks, while the extractor is responsible for extracting watermark information from a concealed perspective

Modeling. We treat embedding and extraction as two separate tasks, and define the processes for embedding and extraction as follows:

$$\hat{Y}_i = G_\phi(Y_i, \mathbf{W}), i \in [1, k] \tag{1}$$

$$\mathbf{S} = \theta(P_s), \hat{\mathbf{W}} = D_\varphi(\mathbf{S}) \tag{2}$$

Furthermore, the embedding and extraction procedures, we conducted training for MarkNeRF M and applied a noise layer to the rendered image. The overall process can be described as follows:

$$(\{\hat{Y}_i\}, \{P_i\}) \rightarrow \theta,\ i \in [1,\ k] \tag{3}$$

$$H_i = \theta(P_i),\ \hat{H}_i = N(H_i),\ i \in [1,\ k] \tag{4}$$

subject to $E(\hat{Y}_i) = Y_i$, $E(\hat{H}_i) = \hat{Y}_i$, and $E\left(\hat{\mathbf{W}}\right) = \mathbf{W}$. In this context, θ denotes the weight associated with MarkNeRF M, N represents the noise layer processing, and four different types of noise attacks (Gaussian noise, Pepper noise, Speckle noise, and Poisson noise) are applied to $\{H_i\}$ in a flexible manner, resulting in the generation of $\left\{\hat{H}_i\right\}$. k denotes the total number of images utilized for training purposes.

We utilized a joint learning approach for the embedding and extraction of watermarks, with the aim of achieving copyright protection for NeRF. A noise layer has been employed to mimic the behavior of potential attackers in the implementation of N. Even if the attacker obtains the model through secondary training without changing the perspective, we can still extract watermarks from the model, making the method proposed in this paper robust.

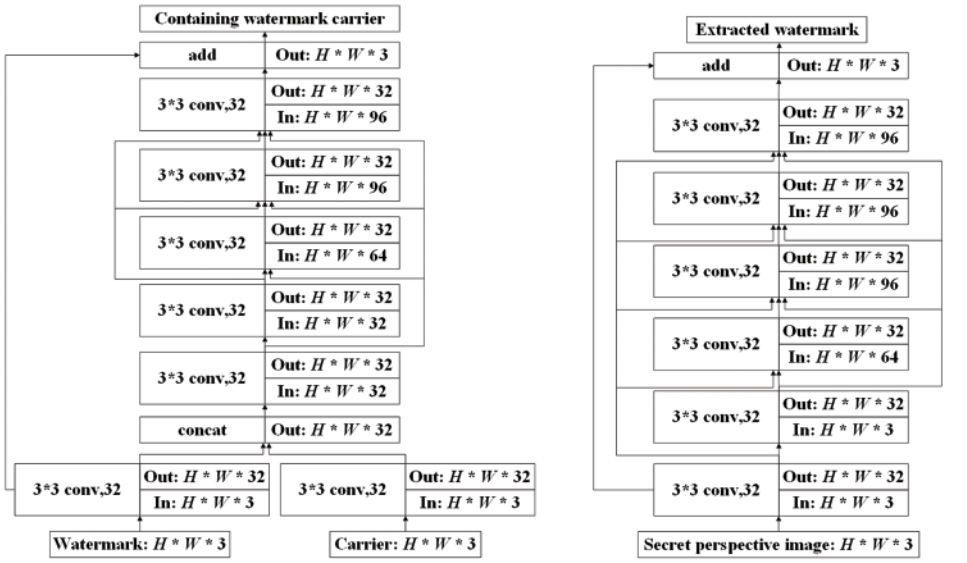
2.3 Network Implementation

Embedding. Drawing upon the remarkable capacity and superior image quality of the encoding-decoding network proposed in [23], we developed an embedding network denoted as G_ϕ, based on this architecture. The specific network structure of G_ϕ is shown in Fig. 3a. G_ϕ receives input from $\{Y_i\}$ and $\mathbf{W}$, and integrates the features from both images via convolutional and concatenation operations. It comprises seven structural layers, each consisting of dense connections and convolutional layers from the convolutional neural network. Moreover, batch normalization (BN) is employed to normalize the data, while the ReLU activation function is utilized in each convolutional layer to enhance feature transmission through dense connections. This strengthens the ability of G_ϕ to extract deep-level features from images. Furthermore, residual connections are employed to capture the global features of the feature map, thus resulting in the generation of a high-quality image $\left\{\hat{Y}_i\right\}$.

MarkNeRF. The neural radiation field is trained using an MLP network, which inputs the three-dimensional coordinate position $\mathbf{x} = (x,\ y,\ z)$ and direction $\mathbf{d} = (\theta,\ \varphi)$ of spatial points, outputs the color $\mathbf{c} = (r,\ g,\ b)$ of spatial points, and the density $\boldsymbol{\sigma}$ of corresponding positions (voxels). In the specific implementation, the position information of $\mathbf{x}$ and $\mathbf{d}$ is encoded first, then $\mathbf{x}$ is input into the MLP network and outputted with $\boldsymbol{\sigma}$ and a 256 dimensional intermediate feature. The intermediate feature and $\mathbf{d}$ are then input together into the fully connected layer to predict colors, and finally a two-dimensional image is generated through volume rendering. The network structure of NeRF is shown in Fig. 4.

In the training process of MarkNeRF, we used the same network structure as NeRF and an 8-layer MLP network. The rendered image H_i was subjected to noise processing to generate image $\hat{H}_i$ Through constraints on the loss between $\hat{H}_i$ and $\hat{Y}_i$, the weight θ of M was optimized.

Extractor. Extractor D_φ takes the secret image $\mathbf{S}$ rendered by MarkNeRF M using the secret camera position P_s as input. It employs seven convolutional layers to extract the watermark image. The specific network structure of the extractor D_φ is shown in Fig. 3b. By adjusting the number of iterations and model parameters during the training process, we successfully achieved the extraction of watermark images from specific secret perspectives by the extractor.



(a) Detailed structural information of embedded network G_ϕ (b) Detailed structural information of extractor D_φ

Figure 3: Detailed network structure of embedded networks and extractors. Both $\boldsymbol{G_\phi}$ and $\boldsymbol{D_\varphi}$ utilize 7-layer convolutional neural networks and dense residual connections for feature propagation

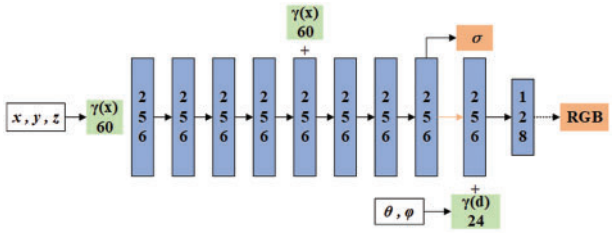


Figure 4: Detailed network structure of MarkNeRF. Similar to NeRF [1], we trained the model using an 8-layer MLP network

2.4 Objective Loss Function

Loss function. We employed an iterative approach to optimize the embedding network, MarkNeRF network, and extractor network. To achieve this, we simultaneously optimized three losses for $E\left(\hat{Y}_i\right) = Y_i$, $E\left(\hat{H}_i\right) = \hat{Y}_i$, and $E\left(\hat{\mathbf{W}}\right) = \mathbf{W}$. $E(\cdot)$ is the expectation operator. The first part of the loss L_{emb} involves assessing the similarity between the original image Y_i and the watermarked image Y_i, utilizing mean square error analysis.

$$L_{emb} = \frac{1}{k}\sum_{i=1}^{k} || Y_i - G_\phi(\hat{Y}_i, \mathbf{W})||_2^2 \tag{5}$$

The second part of the loss L_{mar} pertains to the content loss of the three-dimensional representation. This content loss L_{mar}, comprises two distinct elements: the mean square error loss and the perceptual loss.

$$L_{mar} = \frac{1}{k}\sum_{i=1}^{k} (|| \hat{Y}_i - \hat{H}_i||_2^2 + \lambda||\Psi(\hat{Y}_i) - \Psi(\hat{H}_i)||_2^2) \tag{6}$$

where $\Psi(\cdot)$ denotes the feature representation obtained from a VGG-16 network, and λ is a hyperparameter used to balance the loss functions.

The third component of the loss L_{ext} incorporates the loss of watermark information. Specifically, L_{ext} consists of two components: the mean square error loss and the structural similarity index ($SSIM$) loss.

$$L_{ext} = ||\mathbf{W} - \hat{\mathbf{W}}||_2^2 + \alpha(1 - SSIM(\mathbf{W}, \hat{\mathbf{W}})) \tag{7}$$

where α is a hyperparameter used to balance the loss functions. Therefore, the overall loss to train the copyright-protected neural radiance fields can be obtained as follows:

$$L = \gamma_1 L_{emb} + \gamma_2 L_{mar} + \gamma_3 L_{ext} \tag{8}$$

where γ_1, γ_2, and γ_3 are hyperparameters used to balance the loss functions.

3 Experimental

This section delineates the experimental settings and analyzes the performance of our algorithm under different experimental conditions. The performance of watermark algorithms is typically assessed based on two main criteria: invisibility and robustness. We evaluated the invisibility and robustness of our algorithm by comparing it with other algorithms. Subsequently, to analyze the impact of different modules within the algorithm on the overall framework, we conducted ablation experiments by selectively removing certain modules.

3.1 Experimental Settings

Dataset. We evaluated our algorithm using the NeRF Semantic and Layered Light Field Flow (LLFF) datasets from the NeRF dataset. Among them, LLFF's forward scenes include {*flower*}, {*room*}, {*leaves* . . . } and NeRF Synthetic's 360 degree scenes include {*lego*}, {*drums*}, {*chair*}, {*hotdog*}, {*ship* . . . }. To evaluate the effectiveness of our method, in the NeRF Semantic dataset, our training process involved inputting 100 views per scene. To evaluate the visual quality of our method, we selected 20 images from the test dataset associated with each scenario. Additionally, we conducted renderings of 200 views per scene to examine the accuracy of watermark extraction under different camera perspectives. Furthermore, we randomly selected images from the ImageNet dataset to serve as watermark images. Throughout the experiment, we present all the results as average outcomes.

Training. Our method was implemented using PyTorch. The images were resized to a dimension of 256×256. The hyperparameters were set as follows: $\lambda = 0.01$, $\alpha = 0.5$, $\gamma_1 = 1$, $\gamma_2 = 1$, and $\gamma_3 = 5$. We utilized the Adam optimizer with default values of $\beta_1 = 0.9$, $\beta_1 = 0.999$, $\epsilon = 1 \times 10^{-8}$, and a learning rate of 1×10^{-4}. Throughout the optimization process, we performed 20 K iterations of optimization on the extraction network G_ϕ using the loss function Eq. (5). Subsequently, we conducted 200 K joint optimizations on the parameters θ of the model M and the extractor D_φ, employing Eq. (8). The experiment was executed on an NVIDIA A100 GPU.

Baselines. To Baselines our knowledge, there is currently limited research on watermarking for NeRF. As a result, we compared four strategies to ensure a fair comparison: (1) **LSB [24]+NeRF [1]:** Utilizing the classic LSB algorithm to embed watermark information into the dataset images before training the NeRF model; (2) **DeepStega [25]+NeRF [1]:** Employing the two-dimensional watermarking method DeepStega to process the image prior to training the NeRF model; (3) **HiDDeN [26]+NeRF [1]:** Processing the image using the HiDDeN scheme before training the NeRF model; (4) **StegaNeRF [20]**; (5) **CopyRNeRF [21]**.

Evaluation. We assessed the efficacy of our proposed method relative to other approaches based on the invisibility and robustness of digital watermarking. For *invisibility*, we utilized the Peak Signal-to-Noise Ratio (PSNR), Structural Similarity (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS) [27] to compare the visual quality of the output results watermark image embedding. For *robustness*, we investigated the efficiency of extracting watermark images by evaluating the quality of the rendered images under various distortions. Furthermore, we conducted a study on the ability to extract watermark images from a secret camera perspective.

3.2 Algorithm Performance

Quality of embedded images. In Fig. 5, we randomly select images from various scenes as the original images and embed them into the same watermark image **W** Generally, the details of the watermark are nearly imperceptible within the image. Although enlarged differences may be discernible, the visibility of these differences is not crucial, as long as the labeled image is perceived to be closely aligned with the original image. We conducted additional embedding experiments involving more than 500 images in the dataset, resulting in an average PSNR of 36.41 dB and an average SSIM of 0.975 between the embedded images and the original images.

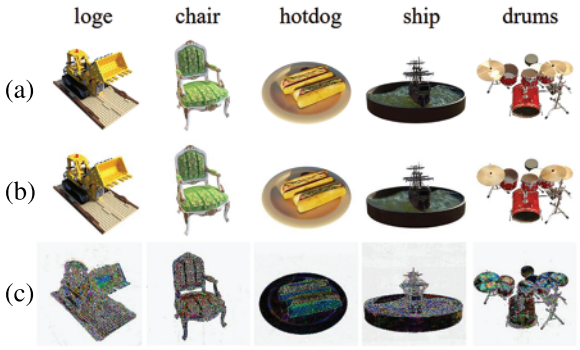


Figure 5: Results of watermark embedding. (a) Original image; (b) Embedded image; (c) Residual ($\times 10$)

Quality of rendering effects and watermark extraction. We conducted a quantitative assessment of the reconstruction quality using various baseline methods, and the experimental results for the lego and trex datasets are presented in Tables 1 and 2. It is worth noting that all watermarking algorithms have a relatively small impact on the quality of NeRF rendering, achieving a high level of reconstruction quality. The qualitative results of different baseline methods are shown in Figs. 6 and 7. Specifically, while **LSB [24]+NeRF [1]**, **DeepStega [25]+NeRF [1]**, and **HiDDeN [26]+NeRF [1]** demonstrated favorable outcomes in terms of steganography or watermarking for two-dimensional images, they were unable to effectively extract information from rendered images due to the alterations caused by NeRF-based view synthesis. In contrast, MarkNeRF demonstrated accurate recovery of watermark images through the incorporation of an additional extractor D_{φ}, and the impact on the rendering quality, as measured by PSNR, remained relatively minimal due to the joint training applied. The results of watermark extraction using the extractor D_{φ} are presented in Fig. 8.

Table 1: Quantitative analysis of invisibility (lego)

Method	NeRF rendering			Watermark extraction	
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑
Standard NeRF	33.51	0.9156	0.1121	N/A	N/A
LSB+NeRF	27.58	0.8445	0.1356	N/A	N/A
DeepStega+NeRF	26.23	0.8486	0.1475	N/A	N/A
HiDDeN+NeRF	27.74	0.8961	0.1423	N/A	N/A
StegaNeRF	30.67	0.9589	0.0280	30.15	0.9722
CopyRNeRF	30.32	0.9657	0.0331	30.42	0.9579
MarkNeRF (Ours)	32.21	0.8943	0.1224	31.59	0.9667

Table 2: Quantitative analysis of invisibility (trex)

Method	NeRF rendering			Watermark extraction	
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑
Standard NeRF	31.24	0.9123	0.0267	N/A	N/A
LSB+NeRF	30.23	0.9026	0.0322	N/A	N/A
DeepStega+NeRF	30.41	0.9078	0.0331	N/A	N/A
HiDDeN+NeRF	30.56	0.9123	0.0321	N/A	N/A
StegaNeRF	31.29	0.9416	0.0355	30.23	0.9421
CopyRNeRF	30.43	0.9312	0.0376	31.47	0.9341
MarkNeRF (Ours)	31.66	0.9457	0.0312	32.12	0.9451

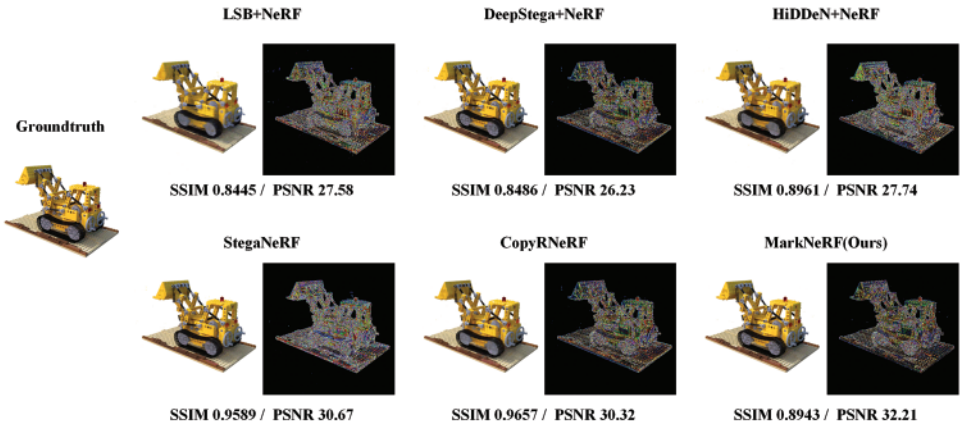


Figure 6: The rendering effect of NeRF model (lego) under different baselines. We demonstrated the residuals (×10) between images rendered with different schemes and ground truth values

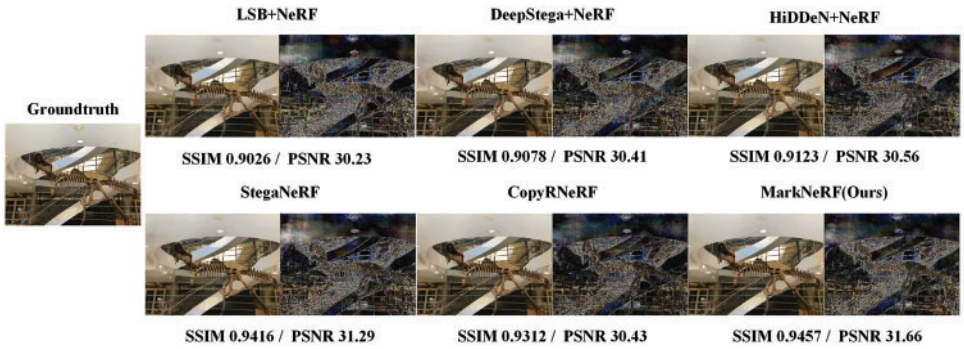


Figure 7: The rendering effect of NeRF model (trex) under different baselines. We demonstrated the residuals ($\times 10$) between images rendered with different schemes and ground truth values

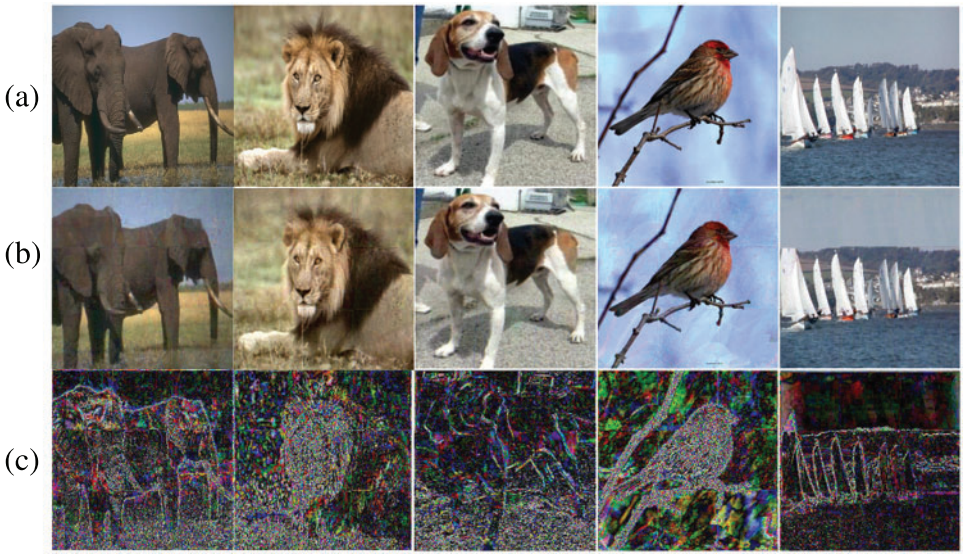


Figure 8: Extracting watermark image effects. (a) Original watermark image; (b) Extracted watermark image; (c) Residual ($\times 10$)

Model robustness on 2D noise. We assess the robustness of our method by subjecting it to various traditional noise attacks. Comparing the SSIM values of images rendered by different methods after passing through noise layers with the original image, the experimental results are detailed in Table 3. We examine several common types of noise, such as Gaussian noise, Poisson noise, Speckle noise and Pepper noise. It is evident from the results that our method demonstrates strong resilience against various two-dimensional noises. Specifically, our approach delivers comparable performance in image rendering to alternative methods in the absence of noise. However, when confronted with different distortions, the superior rendering quality highlights the efficacy of our antidistortion rendering during the training process.

Effectiveness of a secret perspective. To assess the efficacy of the secret perspective in MarkNeRF, images rendered from various perspectives are subjected to testing using an extractor. The experimental results are depicted in Fig. 9. From the obtained results, it is evident that as the rotation angle increases, the extracted watermark image progressively becomes more blurred, eventually becoming

unextractable. However, when the rotation angle is small, adjacent views of the secret perspective can still extract some watermark information. In the following work, we will optimize the extractor network structure so that watermark information can only be extracted from the secret perspective.

Table 3: Effect of noise on image rendering quality

Method	No noise	Gaussian noise	Poisson noise	Speckle noise	Pepper noise
Standard NeRF	0.9452	0.4491	0.9113	0.1415	0.1654
LSB+NeRF	0.9216	0.4123	0.8965	0.1345	0.1547
DeepStega+NeRF	0.9432	0.4288	0.8859	0.1387	0.1621
HiDDeN+NeRF	0.9142	0.4157	0.9014	0.1364	0.1642
StegaNeRF	0.9132	0.4316	0.9074	0.1574	0.1653
CopyRNeRF	0.9245	0.8997	0.9124	0.8421	0.8697
MarkNeRF (Ours)	0.9341	0.9124	0.9289	0.8654	0.8895

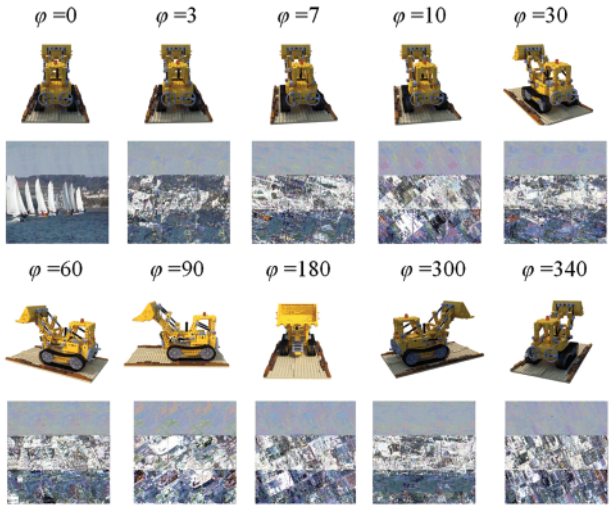


Figure 9: Comparison of watermark extraction effects on different visual images. We test the images rendered from different perspectives in our extractor, and as the rotation angle increases, the extracted watermark information gradually becomes blurred

To quantitatively analyze the influence of different perspectives on the effectiveness of watermark extraction, this study also provides the PSNR and SSIM values between the extracted watermark images and the original watermark images when inputting images from different perspectives. These experimental results are illustrated in Fig. 10. The angle variation φ in the figure represents the angle of counterclockwise rotation around the central z-axis.

3.3 Ablation Study

Impact of noise adding modules. To evaluate the impact of the noise processing module in our watermarking algorithm, we removed the noise module and retrained our model. We compared the performance of models without the noise module and our noise-resistant model in extracting

watermarks under noise attacks. Fig. 11 illustrates the visual results of watermark extraction under different noise attacks for both models. We conducted tests under Poisson noise, speckle noise, pepper noise, and Gaussian noise, respectively. Experimental results demonstrate that compared to the NeRF model without the noise module, our model exhibits less distortion and demonstrates certain robustness against noise influence.

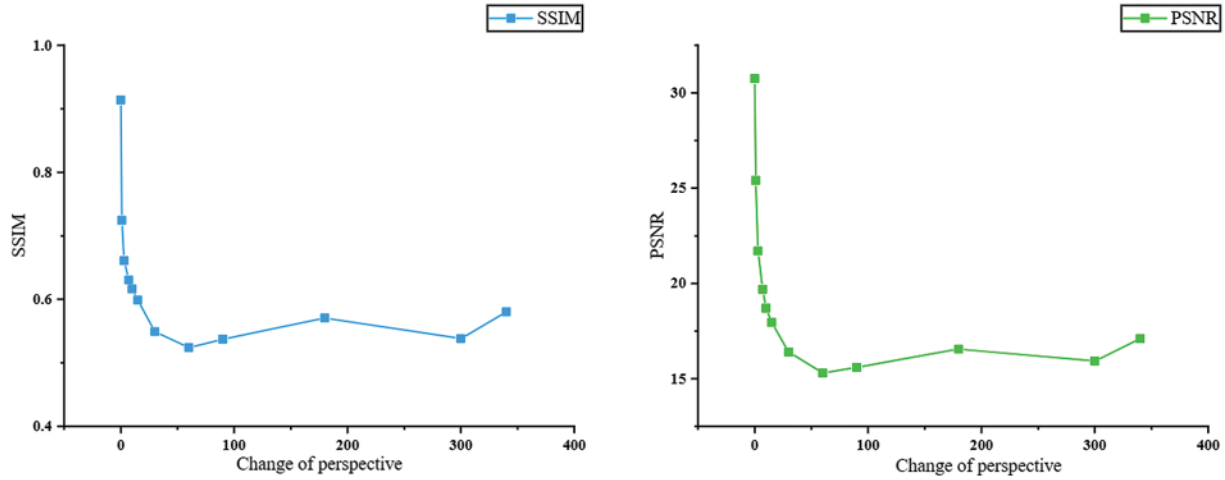


Figure 10: Quantitative analysis of the watermark extraction effect. (a) The influence of the angle φ on the SSIM value between the extracted watermark image and the original watermark image; (b) The influence of the angle φ on the PSNR between the extracted watermark image and the original watermark image

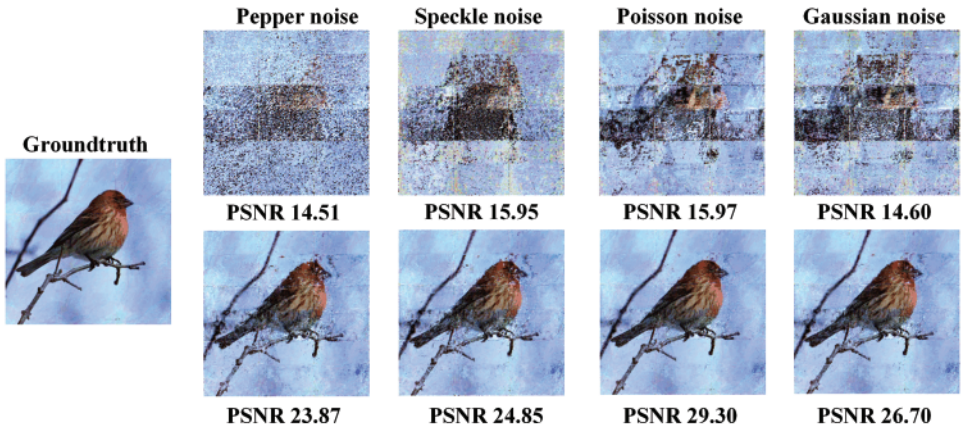


Figure 11: Comparison of experimental results between models without the noise module and our model. The first column illustrates the watermark extraction performance of the model without the noise module under different noise attacks, while the second column depicts the watermark extraction performance of our model under the same noise attacks. We also displayed the PSNR values between each image and ground truth values

Influence of hyperparameters and watermark size. For the hyperparameter settings, we conducted control experiments with different values of γ_3 to balance embedding and extraction capabilities. As shown in Table 4, although different values of γ_3 impose varying constraints on the training and extraction processes, potentially affecting the quality of model rendering and watermark extraction, the experimental results demonstrate that both the rendered images and extracted watermark images produced by the final model exhibit satisfactory quality, indicating insensitivity of our algorithm to γ_3. Furthermore, we attempted embedding watermark images of different sizes into the model to test its generalization ability. As presented in Table 5, our method demonstrates effective generalization to watermark images of different sizes.

Table 4: Experimental results under different γ_3 values

γ_3	NeRF rendering			Watermark extraction	
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑
0.1	33.66	0.9124	0.1126	28.86	0.8964
0.5	32.58	0.8997	0.1075	29.63	0.9013
1	31.41	0.9012	0.1246	30.31	0.9126
2	31.78	0.9047	0.1167	30.24	0.9231
5	32.14	0.9169	0.1099	30.65	0.9314

Table 5: Experimental results under different sizes of watermarks

Size	NeRF rendering			Watermark extraction	
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑
32	33.31	0.9144	0.1022	30.21	0.9169
64	33.46	0.9071	0.1146	31.57	0.9155
96	32.18	0.9164	0.1362	31.22	0.9126
128	33.45	0.9052	0.1274	30.46	0.9074
256	32.97	0.9146	0.1387	31.78	0.9137

Impact of embedding networks. To test the effectiveness of embedding watermark information into the NeRF dataset images, we trained two NeRF models using the original dataset images without embedded watermark information and the dataset images with embedded watermark information, respectively, and compared their rendering results. We conducted experiments on multiple datasets, and the results are shown in Fig. 12. From the experimental results, it can be observed that there is no significant visual difference between the images rendered by the NeRF model trained with embedded watermark information and the NeRF model trained without embedded watermark information. This suggests that during the NeRF model training process, our watermark information was optimally lost, resulting in ineffective transmission of the embedded watermark information to the NeRF model. In future work, we will consider adopting more effective methods to embed watermark information into the NeRF model.

Figure 12: Extracting watermark image effects. (a) Groundtruth; (b) The images rendered by the NeRF model trained with embedded watermark information; (c) The images rendered by the NeRF model trained without embedded watermark information; (d) Residual ($\times 10$) image between (b) and (c)

4 Conclusion

With the proposal of Neural Radiation Field (NeRF), NeRF technology has developed rapidly in 3D content generation and editing, street view maps, and robot positioning and navigation in recent years. However, training NeRF models requires a large amount of resources, and there is currently limited research on NeRF copyright protection. How to effectively protect NeRF has become an important issue. This article introduces a new method of NeRF copyright protection, namely MarkNeRF. We propose a framework that embeds watermark information into the NeRF model through an embedding network. The copyright owner can use the secret perspective information as a key, and then extract the watermark from the image rendered from the secret perspective through an extractor, achieving copyright protection of the NeRF model. In addition, we have also designed anti distortion rendering to enhance the robustness of the model. The experimental results show that our method not only exhibits high visual quality in watermark extraction, but also has a certain degree of robustness, thus verifying the effectiveness of our method in NeRF copyright protection.

Limitations: Although we considered the robustness of the model in the design process, when malicious users attack the weight of the model, the model may be compromised, affecting rendering quality and watermark extraction performance. In addition, the extractor we designed can still extract some watermark information if the secret perspectives are relatively close, and there is still room for optimization in the network structure of the extractor. In future work, we will actively consider how to solve these problems.

Acknowledgement: None.

Funding Statement: This study is supported by the National Natural Science Foundation of China, with Fund Number 62272478.

Author Contributions: The authors confirm the following contributions to this article: Research concept and design: Lifeng Chen and Weina Dong; Experimental, analytical, and interpretive results: Lifeng Chen and Wenquan Sun; Drafted by Lifeng Chen. Jia Liu and Xiaozhong Pan reviewed the results and approved the final version of the manuscript. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: If readers need data, they can contact my email: 3011745933@qq.com.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," *Commun. ACM*, vol. 65, no. 1, pp. 99–106, Jan. 2022. doi: 10.1145/3503250.

[2] K. Liu *et al.*, "StyleRF: Zero-shot 3D style transfer of neural radiance fields," Mar. 24, 2023. Accessed: Jan. 15, 2024. [Online]. Available: http://arxiv.org/abs/2303.10598

[3] Y. J. Yuan, Y. T. Sun, Y. K. Lai, Y. Ma, R. Jia and L. Gao, "NeRF-editing: Geometry editing of neural radiance fields," in *2022 IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, New Orleans, USA, IEEE, 2022, pp. 18332–18343.

[4] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer, "D-NeRF: Neural radiance fields for dynamic scenes," in *2021 IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Nashville, USA, IEEE, 2021, pp. 10313–10322.

[5] W. Xian, J. B. Huang, J. Kopf, and C. Kim, "Space-time neural irradiance fields for free-viewpoint video," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Nashville, USA, 2021, pp. 9421–9431.

[6] C. L. Chowdhary, P. V. Patel, K. J. Kathrotia, M. Attique, K. Perumal and M. F. Ijaz, "Analytical study of hybrid techniques for image encryption and decryption," *Sensors*, vol. 20, no. 18, pp. 5162, 2020. doi: 10.3390/s20185162.

[7] C. M. L. Etoundi *et al.*, "A novel compound-coupled hyperchaotic map for image encryption," *Symmetry*, vol. 14, no. 3, pp. 493, Feb. 2022. doi: 10.3390/sym14030493.

[8] J. Park, J. Kim, J. Seo, S. Kim, and J. H. Lee, "Illegal 3D content distribution tracking system based on DNN forensic watermarking," in *2023 Int. Conf. Artif. Intell. Inform. Commun. (ICAIIC)*, Kumamoto, Japan, 2023, pp. 777–781.

[9] L. Y ariv *et al.*, "Multiview neural surface reconstruction by disentangling geometry and appearance," in *Advances in Neural Information Processing Systems*, 2020, vol. 3, pp. 2492–2502.

[10] I. Yoo *et al.*, "Deep 3D-to-2D watermarking: Embedding messages in 3D meshes and extracting them from 2D renderings," in *2022 IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, New Orleans, LA, USA, IEEE, 2022, pp. 10021–10030.

[11] B. D. Rouhani, H. Chen, and F. Koushanfar, "DeepSigns: A generic watermarking framework for IP protection of deep learning models," May 31, 2018. Accessed: Apr. 14, 2024. [Online]. Available: http://arxiv.org/abs/1804.00750

[12] W. Chen, C. Zhu, N. Ren, T. Seppänen, and A. Keskinarkaus, "Screen-cam robust and blind watermarking for tile satellite images," *IEEE Access*, vol. 8, pp. 125274–125294, 2020. doi: 10.1109/ACCESS.2020.3007689.

[13] C. Zhang, A. Karjauv, P. Benz, and I. S. Kweon, "Towards robust deep hiding under non-differentiable distortions for practical blind watermarking," in *Proc. 29th ACM Int. Conf. Multimed.*, Chengdu, China, 2021, pp. 5158–5166.

[14] G. Y. Zhang, J. H. Liu, and J. Mi, "Research on watermarking algorithms for 3D color point cloud models," (in Chinese), *Comput. Technol. Dev.*, vol. 33, no. 5, pp. 62–68, 2023.
[15] M. Hamidi, A. Chetouani, M. El Haziti, M. El Hassouni, and H. Cherifi, "Blind robust 3D mesh watermarking based on mesh saliency and wavelet transform for copyright protection," *Information*, vol. 10, no. 2, pp. 67, 2019. doi: 10.3390/info10020067.
[16] G. Y. Zhang and J. Cui, "Anti simplified blind watermarking algorithm based on vertex norm 3D mesh model," (in Chinese), *Comput. Eng. Design*, vol. 44, no. 3, pp. 692–698, 2023.
[17] G. N. Pham, S. H. Lee, O. H. Kwon, and K. R. Kwon, "A 3D printing model watermarking algorithm based on 3D slicing and feature points," *Electronics*, vol. 7, no. 2, pp. 23, 2018. doi: 10.3390/electronics7020023.
[18] X. G Xiong, L. Wei, and G. Xie, "A robust color image watermarking algorithm based on 3D-DCT and SVD," (in Chinese), *Comput. Eng. Sci.*, vol. 37, no. 6, 2015.
[19] F. Wang, H. Zhou, H. Fang, W. Zhang, and N. Yu, "Deep 3D mesh watermarking with self-adaptive robustness," *Cybersecurity*, vol. 5, no. 1, pp. 24, 2022. doi: 10.1186/s42400-022-00125-w.
[20] C. Li, B. Y. Feng, Z. Fan, P. Pan, and Z. Wang, "StegaNeRF: Embedding invisible information within neural radiance fields," Dec. 03, 2022. Accessed: Jan. 15, 2024. [Online]. Available: http://arxiv.org/abs/2212.01602
[21] Z. Luo, Q. Guo, K. C. Cheung, S. See, and R. Wan, "CopyRNeRF: Protecting the copyright of neural radiance fields," Jul. 29, 2023. Accessed: Jan. 15, 2024. [Online]. Available: http://arxiv.org/abs/2307.11526
[22] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, "Turning your weakness into a strength: Watermarking deep neural networks by backdooring," in *27th USENIX Secur. Symp.*, Baltimore, MD, USA, 2018, pp. 1615–1631.
[23] K. A. Zhang, A. Cuesta-Infante, L. Xu, and K. Veeramachaneni, "SteganoGAN: High capacity image steganography with GANs," Jan. 29, 2019. Accessed: Jan. 15, 2024. [Online]. Available: http://arxiv.org/abs/1901.03892
[24] S. D. Muyco and A. A. Hernandez, "Least significant bit hash algorithm for digital image watermarking authentication," in *Proc. 2019 5th Int. Conf. Comput. Artif. Intell.*, New York, USA, 2019, pp. 150–154.
[25] S. Baluja, "Hiding images in plain sight: Deep steganography," in *Advances in Neural Information Processing Systems*, Barcelona, Spain, 2017, vol. 30.
[26] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, "HiDDeN: Hiding data with deep networks," Jul. 25, 2018. Accessed: Jan. 15, 2024. [Online]. Available: http://arxiv.org/abs/1807.09937
[27] R. Zhang *et al.*, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, USA, 2018, pp. 586–595.