

GeometrySticker: Enabling Ownership Claim of Recolorized Neural Radiance Fields

Xiufeng Huang^{1,2}, Ka Chun Cheung², Simon See², and Renjie Wan^{1*}

¹ Department of Computer Science, Hong Kong Baptist University

² NVIDIA AI Technology Center, NVIDIA

xiufenghuang@life.hkbu.edu.hk, {chcheung, ssee}@nvidia.com,
renjiewan@hkbu.edu.hk

Abstract. Remarkable advancements in the recolorization of Neural Radiance Fields (NeRF) have simplified the process of modifying NeRF’s color attributes. Yet, with the potential of NeRF to serve as shareable digital assets, there’s a concern that malicious users might alter the color of NeRF models and falsely claim the recolorized version as their own. To safeguard against such breaches of ownership, enabling original NeRF creators to establish rights over recolorized NeRF is crucial. While approaches like CopyRNeRF have been introduced to embed binary messages into NeRF models as digital signatures for copyright protection, the process of recolorization can remove these binary messages. In our paper, we present GeometrySticker, a method for seamlessly integrating binary messages into the geometry components of radiance fields, akin to applying a sticker. GeometrySticker can embed binary messages into NeRF models while preserving the effectiveness of these messages against recolorization. Our comprehensive studies demonstrate that GeometrySticker is adaptable to prevalent NeRF architectures and maintains a commendable level of robustness against various distortions. Project page: <https://kevinhuangxf.github.io/GeometrySticker>.

Keywords: Neural Radiance Fields · Digital Watermarking · Recolorization

1 Introduction

Significant progress [5, 10, 31] has been made in the recolorization of Neural Radiance Fields [18], allowing people to adjust NeRF’s color properties easily. However, as NeRF, a key representation of 3D scenes, becomes possible to be shareable digital assets [28], there is a risk that ill-intentioned users could recolorize NeRF models and illegitimately assert ownership over the modified versions. To prevent NeRF models from such ownership breaches, it is important to allow original NeRF creators to claim ownership for a recolorized NeRF.

A convenient way to assert ownership of digital assets is to embed invisible binary messages as digital watermarking into digital assets. Then, once the

* Corresponding author.

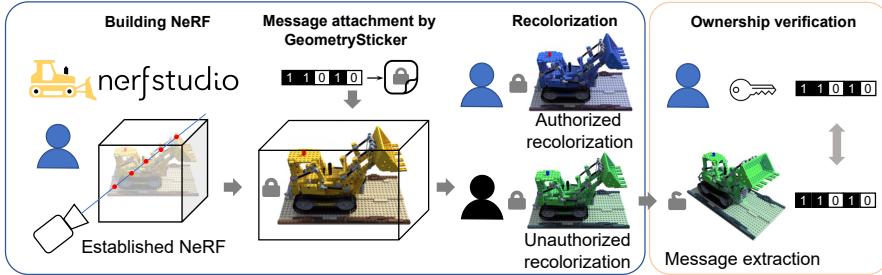


Fig. 1: Our proposed scenario for ownership claim over the recolorized NeRF. Users can construct their NeRF models using readily available platforms, such as NeRFStudio. **Ownership message attachment:** They can then swiftly stick binary messages onto those created NeRF models via our proposed GeometrySticker. These watermarked NeRF models remain suitable for standard recolorization processes (herein termed Authorized recolorization). **Ownership verification:** Should unauthorized recolorization occur, the creators of the NeRF models can retrieve the watermarks from the altered models to verify ownership.

digital assets are maliciously edited, the owners can extract binary messages from NeRF [18] to claim ownership. For example, CopyRNeRF [15] has been introduced to safeguard the copyright of NeRF. Their method embeds binary messages in a way that aligns with the color representation within NeRF. Then, they utilize a message decoder to extract binary messages from images rendered from NeRF to verify the copyright. Yet, as CopyRNeRF [15] deeply relies on the combination between binary messages and color representation, such messages directly become “irretrievable” on rendered images when simply altering the color representation. Thus, the cornerstone of ensuring ownership claim over the recolorized NeRF model lies in making binary messages robust to alterations to color representation.

If recolorization is mainly conducted on color representation, a straightforward solution would be to hide messages into cover media [8] unrelated to color representation. Then, the envisioned ownership claim and recolorization can be achieved in a concurrent way. In general, NeRF [18] relies on color and geometry to finish volume rendering. As the two key components are represented separately via two MLPs, geometry components become suitable cover media for hiding binary messages. Then, we can balance the target for claimability and recolorization.

However, three challenges stand in the way of achieving the above goals. **First**, since its introduction, several NeRF variants [2, 19, 24] have already been developed. Their internal structures used for geometry representation also have significant differences. *The watermarks should be scalable to NeRF variants.* **Second**, NeRF [18] and its variations [2, 19] utilize neural networks to learn implicit neural representations for reconstructing 3D scenes. Additional messages can easily disrupt the geometry structure represented in the networks. Then, it is

essential to ensure that *the changes made by the embedded binary messages are minimal*. At last, an effective watermarking method should extract the copyright messages from such minimal changes.

We introduce GeometrySticker for the ownership claim of recolorized NeRF. GeometrySticker is with the following characteristics to address the above challenges: 1) **Scalability**. Rather than embedding messages into geometry representation, we propose to attach messages onto geometry like stickers. Besides a simple network used for message representation, the attachment of such a sticker does not rely on any additional structure-specific optimization, which can be scalable to various variants. 2) **Subtlety**. The cover media selected for embedding binary messages occupy only small sections within the NeRF model, ensuring that modifications introduced by the message embedding remain subtle. 3) **Ubiquity**. The cover media are accessible from every viewpoint, ensuring NeRF owners can retrieve binary messages from each perspective.

In Fig. 2, we display our framework for implementing GeometrySticker. Our GeometrySticker is a lightweight Multilayer Perceptron (MLP) capable of translating binary messages into formats compatible with the geometry representation used in NeRF and its variants [1, 2, 19, 24]. Then, we can seamlessly integrate this compatible form of message representation with the chosen cover media, irrespective of the NeRF architectures, thus guaranteeing scalability. Then, we employ the 3D points sampled throughout the ray marching process as the cover media, and we specifically choose the 3D points near the objects’ surfaces to affix our binary messages. These selected 3D points around the objects’ surfaces represent a minor fraction of the total representations, thus ensuring subtlety. Ultimately, we guarantee that these cover media are accessible from every viewpoint, ensuring that the messages can be accessed from each perspective, thereby ensuring ubiquity.

As shown in Fig. 1, after the message attachment, we can easily recolorize the watermarked NeRF with GeometrySticker for authorized recolorization. However, if unauthorized recolorization is triggered, NeRF creators can easily retrieve binary messages from 2D images rendered from recolorized NeRF for ownership verification. The proposed GeometrySticker has the following key characteristics:

- Safeguarding the ownership claim of NeRF even when the color attributes have been altered
- Using the geometry components associated with selected 3D points to achieve the subtlety and ubiquity of the embedded ownership messages.
- Designing a message sticker for ownership message attachment to achieve the scalability of our proposed solution.

Our GeometrySticker, exhibiting high scalability, is capable of being generalized across various NeRF variants [1, 2, 19, 24] that use neural representations for geometry. Based on our experiments, the use of GeometrySticker does not impair the effectiveness of current recolorization approaches [5, 10, 31] designed for NeRF. Furthermore, besides the recolorization approaches for NeRF, the binary messages can still be reliably extracted even when the 2D images rendered from NeRF are subjected to direct image-level color modifications.

2 Related work

NeRF and its variants. NeRF [18] and its variants [1, 19, 24, 29, 42, 43] show the capability to create realistic 3D representations of objects and scenes from 2D images with different perspectives. To improve the efficiency of scene representation, Plenoxel [24] reconstructs the scene in a sparse voxel grid and renders each ray sample via trilinear interpolation of the neighboring voxel coefficients. TensoRF [2] factorizes the radiance fields by vector-matrix decomposition for efficient scene modeling. InstantNGP [19] optimizes the input encoding with a multi-resolution hash table to reduce the number of floating point and memory access operations. As the creation of NeRF becomes more accessible, individuals are more inclined to easily recreate their preferred 3D scenes and share them with the public. It’s important to address potential breaches of ownership in the process of such sharing.

Recolorization of NeRF. The recolorization of NeRF [4, 5, 10, 31] has achieved remarkable performance. CLIP-NeRF [31] use text prompts to alter the color supervised by CLIP-based [23] matching loss. PaletteNeRF [10] decomposes the appearance of 3D points into a linear combination of palette-based bases across the scene for photorealistic color editing. Similarly, RecolorNeRF [5] also decomposes the scene into a set of color layers to form a palette for color altering based on the TensoRF [2] architecture. As the recolorization of NeRF has become effective in recolorizing the 3D scene, it is important to explore an effective approach to protect the intellectual property of NeRF models when their color properties are modified.

Ownership assertion of NeRF. Traditional 2D watermarking methods embed information in the least significant bits of image pixels [30]. Significant progress has been made in deep-learning-based image watermarking [27, 32–34, 37, 39, 40]. HiDDeN [44] is one of the first deep image watermarking methods that outperforms traditional methods. 3D watermarking approaches are usually designed for explicit 3D models [3, 20, 22, 26, 35, 38]. However, these methods are not applicable to the copyright protection of NeRF due to its implicit property. Recently, StegaNeRF [12] designs a framework for steganographic information embedding in NeRF renderings. CopyRNeRF [15] generates watermarked color representations to ensure the invisibility of hidden copyright messages. However, when faced with recolorization, the hidden information embedded in the two methods might become unrecoverable. This prompts us to investigate ensuring ownership claims over recolorized NeRF models.

3 Preliminaries on the watermarking of NeRF

The goal of watermarking NeRF is to safeguard its copyright and ownership by integrating binary messages into this burgeoning digital asset for 3D scenes. In general, NeRF [18] builds a function $f: (\mathbf{x}, \mathbf{d}) \rightarrow (\sigma, \mathbf{c})$ to map the position \mathbf{x} and viewing direction \mathbf{d} to the point’s density σ and color \mathbf{c} . Vanilla NeRF [18] uses a MLP Θ_σ and the encoding function $\gamma_{\mathbf{x}}$ to map the 3D location \mathbf{x} into density

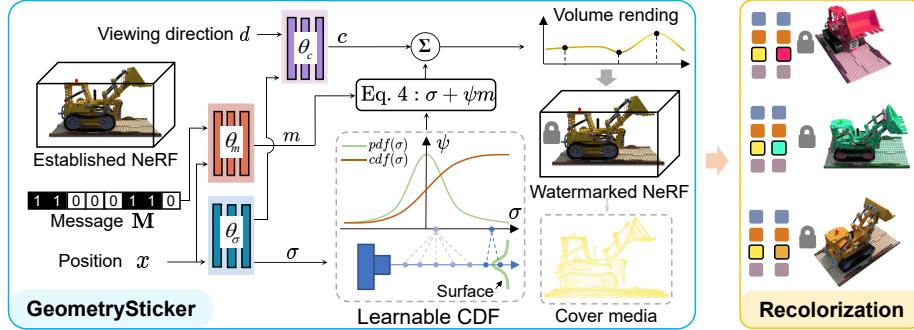


Fig. 2: The framework of our proposed GeometrySticker. We employ a Multilayer Perceptron (MLP), denoted as θ_m to convert binary messages into a format that aligns with NeRF’s geometry representation. Following this, a learnable Cumulative Distribution Function (CDF) is utilized to select 3D points close to the object surfaces with high geometry values as the cover medium. Subsequently, an addition is applied to attach the message (Equation (4)) onto the chosen cover media. Through this message attachment process, a watermarked NeRF is generated, capable of retaining its efficacy across diverse recolorizations. Should any unauthorized changes to color attributes occur, NeRF owners can retrieve the integrated watermarks to assert their ownership. θ_σ and θ_c indicate the representation for geometry and color.

value σ and the intermediate geometry feature output of \mathbf{z} : $[\sigma, \mathbf{z}] = \Theta_\sigma(\gamma_{\mathbf{x}}(\mathbf{x}))$. Another MLP Θ_c and the encoding function γ_d are used to map the geometry feature \mathbf{z} and viewing direction \mathbf{d} into the color value $\mathbf{c} = \Theta_c(\mathbf{z}, \gamma_d(\mathbf{d}))$. Once the optimization settles down, an implicit scene representation can be obtained, and all scene information can be stored in MLP as its network weights.

Given the challenge of retrieving binary messages from the implicit representation of NeRF, existing strategies [12, 15] typically design a method to convey copyright messages from the implicit neural representation to the 2D rendered images. Following that, a CNN-based message extractor is commonly employed to retrieve binary messages from the 2D images. The binary message embedding can be represented as $f_{\mathbf{M}}$, where \mathbf{M} is the binary messages with length $N_{\mathbf{M}}$. The message embedding is fused with the implicit neural representation for rendering the 2D watermarked images \mathbf{I}_w . The corresponding message extraction process can be denoted as $\hat{\mathbf{M}} = D(\mathbf{I}_w)$, where D denotes a CNN-based message extractor used to extract the hidden information $\hat{\mathbf{M}}$ from the 2D watermarked image \mathbf{I}_w .

4 Proposed method

We have shown the scenario for our GeometrySticker in Fig. 1, where people can claim ownership over the recolorized NeRF. In our scenario, for a NeRF model established via public resources such as NeRFStudio [28], creators can

effortlessly integrate binary messages into these established NeRF models using GeometrySticker for potential ownership claims. Once ownership messages are embedded into NeRF models, authorized recolorization can still be easily performed on the watermarked NeRF for legitimate uses. Yet, in the event of unauthorized recolorization, NeRF creators can utilize their available message extraction tool to retrieve binary messages from the rendered images, thereby affirming their ownership of their digital assets.

4.1 GeometrySticker

Choosing cover media. Choosing suitable cover media to conceal ownership messages is crucial in digital watermarking [8]. CopyRNeRF [15] has shown that embedding binary messages directly into geometry components can readily result in visible artifacts, thereby degrading the quality of scene representation. In fact, these artifacts partly originate from 3D points in empty spaces that have lower geometry values. In volume rendering [11], N_p points are sampled along the camera marching rays with color and geometry values $\{(\mathbf{c}_i, \sigma_i)\}_{i=1}^{N_p}$. These points exhibit low geometry values in empty spaces and high values at the surfaces of objects [7]. Incorporating additional information into low geometry value 3D points located in empty spaces can lead to easily detectable alterations. Thus, rather than directly incorporating the hidden messages into the whole geometry representation, we consider those 3D points located on the object surfaces as the cover media. These 3D points on the object surfaces often exhibit high values, making the attachment of messages into them result in less conspicuous changes.

To precisely pinpoint the 3D points located on the surfaces of objects, we utilize the Laplace Cumulative Distribution Function (CDF) equipped with a learnable parameter. This approach can help in determining an appropriate threshold to filter out 3D points that exhibit high geometry values as:

$$\psi = \frac{1}{2} + \frac{1}{2} \cdot \text{sign}(\sigma - \mu) \cdot \left(1 - \exp \left(-\frac{|\sigma - \mu|}{\beta} \right) \right), \quad (1)$$

where μ and β are the average and deviation of the geometry field, and $\psi \in [0, 1]$ is a probability that any geometry values in the geometry field is less than or equal to the given geometry values σ . The probability ψ can be used as an importance value to indicate whether the geometry value is a large number or not.

While utilizing fixed parameters for thresholding points on object surfaces is feasible, adopting a naive strategy for determining mean and standard deviation values for such thresholding might lead to inflexible cutoffs. This could unintentionally cover too many points, causing notable distortions, or on the flip side, too few points, thus hindering the efficient embedding of messages for future extraction by a message extractor.

Rather than using a fixed threshold for cover media generation, we consider β to be a learnable parameter to be optimized during the cover media generation. Then, the range used for the cover media generation can be adaptively

adjusted according to rendered contents and message attachment efficiency to ensure the invisibility of embedded messages. The outcomes illustrated in Fig. 2 demonstrate that the chosen 3D points, serving as the cover media, effectively form sparse point clouds that capture the essential information of the target objects. We optimize the importance value ψ with a sparsity loss [14] as follows:

$$\mathcal{L}_{sparse} = \frac{1}{|N_p|} \sum_{\psi_i} [\log(\psi_i) + \log(1 - \psi_i)], \quad (2)$$

which forces the importance value ψ to be close to either zero or one. The importance values ψ close to one indicate the 3D points with high geometry values. These points are selected as the cover media and only occupy small sections of the NeRF geometry to ensure subtlety.

Message sticker. To maintain scalability, we avoid the data hiding techniques used in previous methods, such as CopyRNeRF [15]. CopyRNeRF [15] inherently alters the underlying NeRF structure, which depends on particular configurations and exhibits reduced scalability. Instead, we propose a message sticker that can attach messages to the selected cover media by **summation** like a sticker. The message sticker Θ_m can be achieved via an MLP as:

$$m = \Theta_m(\gamma_x(\mathbf{x}), \mathbf{M}), \quad (3)$$

where \mathbf{M} is the binary message with length N_b and m is the one dimensional message embedding. Then, we can directly attach the message embedding m into geometry σ via ψ defined in Equation (1):

$$\tilde{\sigma} = \sigma + \psi m, \quad (4)$$

where $\tilde{\sigma}$ is a watermarked geometry value. During volume rendering, the information attached via the message sticker can be incorporated into the rendered pixel values \tilde{C} as follows:

$$\tilde{C} = \sum_{i=1}^N \exp\left(-\sum_{j=1}^{i-1} \tilde{\sigma}_j \delta_j\right) (1 - \exp(-\tilde{\sigma}_i \delta_i)) \mathbf{c}_i, \quad (5)$$

where δ is the distance between adjacent sample points, $\tilde{\sigma}$ is the watermarked geometry with a message attachment and \mathbf{c} is the color sampled along the ray. During training, to guarantee ubiquity, we repeat the above operations in each viewing point, which ensures that such 3D points exist at each perspective. During message extraction, the binary messages incorporated on cover media can be easily extracted from the watermarked image \mathbf{I}_w via a message extractor D_χ as $\hat{\mathbf{M}} = D_\chi(\mathbf{I}_w)$, where $\hat{\mathbf{M}}$ is the binary messages extracted by the message extractor and χ is a trainable parameter.

Optimization. The message attachment in Equation (4) is based on simple addition operation, with the training just focused on enabling the message sticker to adapt binary messages into a form that aligns with the diverse architectures employed by NeRF [18] and its variants. Our optimization contains three components: 1) we optimize the learnable variable β defined in Equation (1) to find

an appropriate threshold for identifying 3D points with high geometry values as the cover media; 2) we train the message sticker in Equation (3) for embedding the binary messages into the 3D points; 3) we also train the message extractor D_χ to extract the hidden message from the watermarked rendered images \mathbf{I}_w . The above three components can be simply optimized via a message loss as $\mathcal{L}_{msg} = \text{BCE}(D_\chi(\mathbf{I}_w), \mathbf{M})$, where BCE is the binary cross entropy loss, \mathbf{I}_w is the watermarked rendered image and \mathbf{M} is the ground truth message. Then, the classical MSE loss adopted by NeRF [18] is employed to ensure that the GeometrySticker does not compromise the visual quality of rendered contents as $\mathcal{L}_{cont} = \|\mathbf{I}_w - \mathbf{I}_o\|_2^2$, where \mathcal{L}_{cont} is the content loss and \mathbf{I}_o is the original image. We also train a CNN-based classifier C_ϕ to classify whether the rendered images contain watermarks as $\mathcal{L}_{cls} = \text{BCE}(C_\phi(\mathbf{I}_w), C_\phi(\mathbf{I}_u))$, where \mathbf{I}_u is the unwatermarked rendered image and ϕ is a trainable parameter. The overall loss functions for our GeometrySticker can be incorporated by optimizing the following loss functions:

$$\mathcal{L}_{total} = \mathcal{L}_{cont} + \mathcal{L}_{msg} + \mathcal{L}_{cls} + \mathcal{L}_{sparse}. \quad (6)$$

4.2 Recolorization

Once the binary messages have been attached to the geometry components via GeometrySticker, watermarked NeRF models can be easily recolorized in a claimable manner via off-the-shelf approaches for recolorization. We consider two off-the-shelf recolorization approaches. The first one is CLIP-based recolorization proposed in CLIPNeRF [31]. In CLIP-based recolorization, the color representation within a NeRF model is modified using CLIP [23] features derived from a specified text prompt. We adhere to the protocols set forth in CLIPNeRF [31], employing a CLIP [23] feature loss to guide the update of the color representation. Our second recolorization strategy is the palette-based recolorization. This approach begins by establishing a color palette that encompasses all fundamental color components. Subsequent precise recolorization is accomplished by adjusting the color palette, specifically by allotting distinct RGB values to designated color layers. We randomly select 10 reference colors from the Standard sRGB / Rec.709 color gamut to recolorize the NeRF model. Additionally, users have the option to apply image-level recolorization techniques to directly alter the colors in rendered images using traditional methods (*e.g.*, color jittering). Some recolorization results are displayed in Fig. 3. Due to page limitations, **more details about this part can be found in the supplementary materials**.

4.3 Implementation details

Our GeometrySticker is implemented on PyTorch. The whole pipeline can be easily combined with popular NeRF architectures like InstantNGP [19] or TensoRF [2]. Besides, we also implement on vanilla NeRF [18]. NeRF and Instant-NGP [19] utilize MLP layers, and TensoRF [2] utilizes a density grid to predict

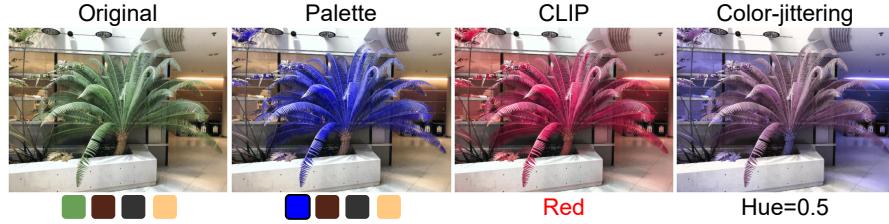


Fig. 3: Recolorized samples from our selected approaches. From left to right: original image, the result obtained via palette-based recolorization [10] by changing the green base color to blue, the CLIP-based recolorization [31] result by giving “red” text prompt, and the color-jittering result by changing hue.

volume density. We train our GeometrySticker to find the important geometry component for attaching the copyright messages.

The patch size is set to 400×400 for each rendered image during training. During training, we apply several types of 2D distortions on the watermarked rendered images to achieve robustness, including Gaussian noise, random rotation, random cropout, and Gaussian blur. As our motivation is to protect the copyright of NeRF that has already been created, we first create several NeRF models by training them on Blender and LLFF datasets [18, 19] following standard settings. Then, we use GeometrySticker to attach binary messages on established NeRF and apply the aforementioned recolorization methods in Section 4.2 on the NeRF watermarked by GeometrySticker to test the watermarking robustness against recolorizations. Specifically, we employ the VGG16 [25] network as the backbone of the CNN-based message extractor. An average pooling is then performed, followed by a final linear layer with a fixed output dimension N_b to produce the continuous predicted message $\hat{\mathbf{M}}$. Training typically takes 5000 steps and can be completed within 45 minutes. All experiments are conducted on a single V100 GPU.

5 Experiments

5.1 Experimental settings

Dataset. We use two benchmark datasets **Blender** [18] and **LLFF** [17] for evaluation. For Blender, we directly follow the dataset splitting to use 100 viewpoints from the training set to train our GeometrySticker and then render 200 views from the testing set to validate whether the binary messages can be extracted under different viewpoints and color editing conditions. For LLFF, we follow the dataset splitting in NeRF [18]. In general, 1/8 images in each scene are used to test the visual quality and bit accuracy of binary message extraction, and others are used to train our GeometrySticker. We report average values across all testing viewpoints in our experiments. All testing viewpoints are used for computing the average values during the evaluation session.

Baselines. Our evaluations consist of three parts to verify **claimability**, **re-colorization**, and **scalability**. For **claimability**, we compare our proposed GeometrySticker with four baselines for a fair comparison: 1) **HiDDeN** [44] + **NeRF** [18]. We process images with HiDDeN [44], a classical image watermarking method, before the training of NeRF; 2) **CopyRNeRF** [15]. A state-of-the-art method for protecting the copyright of NeRF [18] by using digital watermarking; 3) **StegaNeRF** [12]. A state-of-the-art data hiding method for steganographic information embedding of NeRF. We adapt StegaNeRF [12] to embed binary messages with the CNN-based message extractor for message retrieval; 4) **Unwatermarked NeRF**. We also compare the rendered results from NeRF watermarked by GeometrySticker with the rendered results from the non-watermarked version of NeRF to evaluate whether GeometrySticker undermines visual quality. For **recolorization**, we compare the differences between the watermarked recolorized images with the corresponding unwatermarked recolorized images to investigate whether our GeometrySticker undermines recolorization. For **scalability**, we validate whether GeometrySticker can be easily adapted into various NeRF architectures, including vanilla NeRF [18], InstantNGP [19], and TensoRF [2]. We also evaluate whether GeometrySticker is compatible with the existing recolorization schemes mentioned in Section 4.2.

Evaluation methodology. We evaluate the performance of GeometrySticker compared with other digital watermarking methods using the standard of capacity, invisibility, and robustness. For *capacity*, we set hidden messages bit length to 48 bits, aligning with the maximum length previously employed in 3D model watermarking methods [15, 38]. For *invisibility*, we evaluate the visual quality with PSNR, SSIM, and LPIPS [41] by comparing the visual quality of the rendered images before and after GeometrySticker watermarking. For *robustness*, we evaluate whether the hidden messages can keep consistent against various distortions and recolorizations. Besides normal situations, we consider different distortions including Gaussian noise, rotation, cropout, and Gaussian blur. Different recolorization pipelines are employed to ensure adequate comparisons.

5.2 Experimental results

Can we claim ownership over recolorized NeRF models? We first assess whether our GeometrySticker can maintain its effectiveness under various recolorization operations. We consider model-level recolorization, including CLIP-based and palette-based recolorization. We also apply color jittering to randomly change images’ hues as a general image-level color alternation. As shown in Table 1, HiDDeN [44] + NeRF [18] completely fails to perform well under both image- and model-level recolorization. CopyRNeRF [15] also shows degraded performance since CopyRNeRF [15] uses the color representation for hiding the binary message. Since StegaNeRF [12] depends on the complete geometry and color representation for data hiding, the hidden messages are susceptible to being compromised under both image- and model-level recolorization. Moreover, the intrinsic architectures of CopyRNeRF [15] and StagaNeRF [12] do not fully support palette-based recolorization, which leads to “N.A.” in Table 1. In contrast,

Datasets	Methods	PSNR/SSIM↑ LPIPS↓	Bit accuracy ↑ (%)		
			Color-jitter	CLIP	Palette
Blender	HiDDeN [44] + NeRF [18]	30.80/0.9999	0.0167	50.13	51.08 50.91
	CopyRNeRF [15]	29.99/0.9999	0.0171	51.32	49.96 N.A.
	StegaNeRF [12]	31.48/0.9999	0.0149	54.18	52.48 N.A.
	GeometrySticker	32.13/0.9999	0.0136	99.33	99.50 99.40
LLFF	HiDDeN [44] + NeRF [18]	28.55/0.9999	0.2329	50.74	50.28 50.56
	CopyRNeRF [15]	27.40/0.9998	0.2322	47.64	49.89 N.A.
	StegaNeRF [12]	29.15/0.9998	0.2242	52.51	50.51 N.A.
	GeometrySticker	30.82/0.9999	0.2165	96.50	97.12 96.88

Table 1: Reconstruction qualities and message decoding bit accuracies under various recolorizations. PSNR/SSIM and LPIPS are computed between the watermarked recolorized images and the corresponding unwatermarked recolorized images. ↑(↓) means higher (lower) is better. “N.A.” means Not Applicable since CopyRNeRF [15] and StegaNeRF [12] are not compatible with palette-based recolorization.

Datasets	Methods	PSNR/SSIM↑ LPIPS↓	Bit accuracy ↑ (%)				
			None	Noise ($\nu = 0.1$)	Rotation ($\alpha = \pm\pi/6$)	Cropout ($s \leq 25\%$)	Blur ($\xi = 0.1$)
Blender	HiDDeN [44] + NeRF [18]	30.44/0.9490	0.0829	50.19	49.84	50.12	50.09 50.16
	CopyRNeRF [15]	30.29/0.9478	0.0813	66.80	65.92	64.52	63.44 66.22
	StegaNeRF [12]	30.96/0.9583	0.0564	100	90.21	57.17	60.30 92.88
	GeometrySticker	32.39/0.9713	0.0503	100	99.25	98.87	98.75 99.88
LLFF	HiDDeN [44] + NeRF [18]	23.80/0.7670	0.2515	51.18	49.20	50.59	50.33 49.98
	CopyRNeRF [15]	24.03/0.7747	0.2575	66.07	65.23	64.83	63.06 65.35
	StegaNeRF [12]	24.96/0.8011	0.2498	100	88.36	63.32	61.03 90.48
	GeometrySticker	25.67/0.8023	0.2465	100	99.36	98.55	98.94 99.59

Table 2: Reconstruction qualities and bit accuracy compared with different baselines. PSNR/SSIM and LPIPS are computed between the original and watermarked rendered images. The results are computed on the average of all samples.

our approach is uniquely adaptable to all three existing recolorization schemes, further underscoring the scalability of our method. In contrast, the binary messages embedded by our GeometrySticker remain effective against both image- and model-level recolorization and achieve high bit accuracy.

Does GeometrySticker undermine recolorization? We discuss whether our GeometrySticker undermines recolorization in Table 1 and Fig. 4. We evaluate the variations between recolored samples from NeRF models that are not watermarked and those that have been watermarked. In Table 1, as it is difficult to obtain identical recolored pairs via CLIP-based recolorization, we use Palette-based colorization for our approach and HiDDeN [44] + NeRF [18]. We utilize color-jittering to change the images’ hue for CopyRNeRF [15] and StegaNeRF [12], since the two methods are not compatible with Palette-based colorization. The better quantitative values in Table 1 show that our samples can achieve higher similarity to the unwatermarked recolored images. This is further supported by the qualitative results shown in Fig. 4. Moreover, our technique

Architecture	PSNR/SSIM↑	LPIPS↓	Bit accuracy ↑
NeRF [18]	27.44/0.8759	0.1667	100%
InstantNGP [19]	28.59/0.8868	0.1304	100%
TensoRF [2]	29.18/0.8907	0.1370	100%

Table 3: Evaluation of scalability by incorporating our GeometrySticker into different NeRF architectures. PSNR/SSIM and LPIPS are computed between the samples rendered from watermarked NeRF models and the corresponding ground truth images without watermarks.

uniquely maintains a balance between recolorization quality and bit accuracy post-recolorization, unlike other methods, which show significantly reduced bit accuracy following recolorization.

Can GeometrySticker function properly without recolorization? We evaluate if GeometrySticker functions properly in standard scenarios without recolorization. We also evaluate its robustness by applying several types of 2D distortions to rendered images, including Gaussian noise with deviation ν , random rotation with parameters α , random cropout with a parameter s , and Gaussian blur with deviation ξ . As shown in the Table 2, HiDDeN [44] fails to extract the binary messages from the renderings of NeRF. CopyRNeRF [15] can limitedly extract hidden messages from the renderings and show undermined robustness to different image distortions. Although StegaNeRF [12] can extract the hidden messages, it shows vulnerability to different types of image distortions. Our GeometryStricker reliably extracts hidden messages and shows robustness to different image distortions.

Is GeometrySticker scalable? We have shown that our method can achieve scalability over the three main recolorization schemes in Table 1. We further evaluate the scalability of our proposed GeometrySticker on three typical NeRF architectures, including vanilla NeRF [18], InstantNGP [19], and TensoRF [2]. From Table 3, GeometrySticker can achieve high invisibility and bit accuracy in these conditions, which reconfirms our scalability.

Ablation study. The selection strategy of 3D points is a key architecture in our framework. We focus on investigating this part in our ablation study. As shown in Fig. 6, attaching messages to all geometry components can cause obvious distortion, which is aligned with the previous findings in CopyRNeRF [15]. Applying simple Laplace CDF with fixed thresholds for message attachment can reduce perturbation on the NeRF geometry but still with noticeable distortion. Our learnable Laplace CDF can find an optimal threshold for message attachment, making the visual distortion imperceptible.

Potential threat analysis. The experiments in Table 2 have highlighted the robustness of our method to common image distortions. Besides, as the recolorization is also a very powerful modification operation, our previous experiments also demonstrate that GeometrySticker can show robustness to different recolorizations. We further investigate the robustness of the embedded watermarks

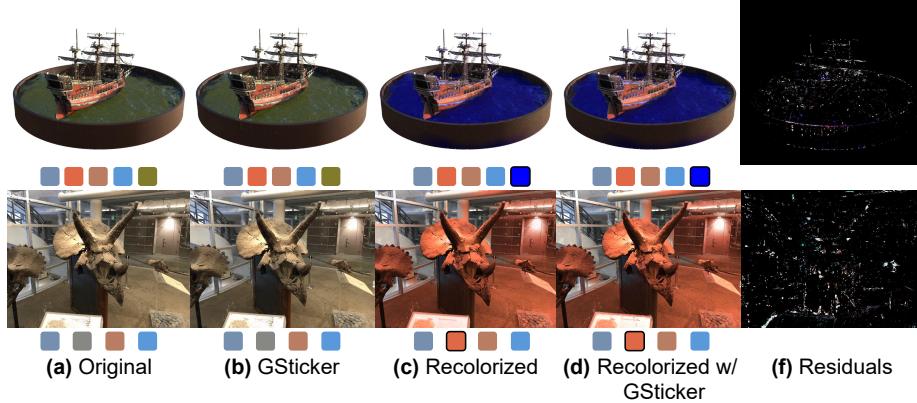


Fig. 4: Qualitative evaluation of whether GeometrySticker (GSticker) compromises the recolorization. Column (a) is rendered from an original NeRF without GSticker. Column (b) is rendered from NeRF watermarked by GeometrySticker. Column (c) is the result of applying palette-based recolorization on NeRF models in Column (a). Column (d) is the result of applying palette-based recolorization on NeRF models in Column (b). Column (e) shows **residual maps** between Column (c) and Column (d). The minor differences in the residual map indicate that GeometrySticker does not undermine the recolorization.

against various possible deliberate interferences and security threats including the adversarial attack and model purification.

Adversarial attack. We consider the situation if the message extractor has leaked. A malicious user can use an adversarial attack such as PGD [16] to remove the hidden messages by optimizing the rendered images via a PSNR constraint. The goal is to minimize the Euclidean distance between a pre-sampled random binary message and the message extractor’s output, which could replace the original hidden message with a random one. As shown in Fig. 5, adversarial attacks can indeed result in a reduction of bit accuracy while causing only minimal visual distortion. Thus, it’s essential to keep the message extractor private.

Model purification. Another one is the model purification by fine-tuning the model with non-watermarked images. We consider an extreme situation that the attackers can directly access the original non-watermarked images used for the NeRF creation. Based on this assumption, we implement this attack on GeometrySticker by eliminating the message loss and fine-tuning the model solely through perceptual loss.

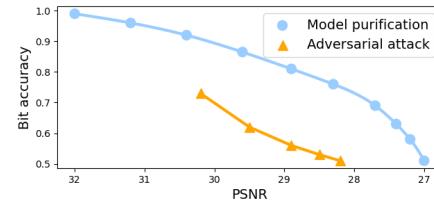


Fig. 5: Robustness to model purification and adversarial attacks. We show the correlations between the PSNR and bit accuracy.

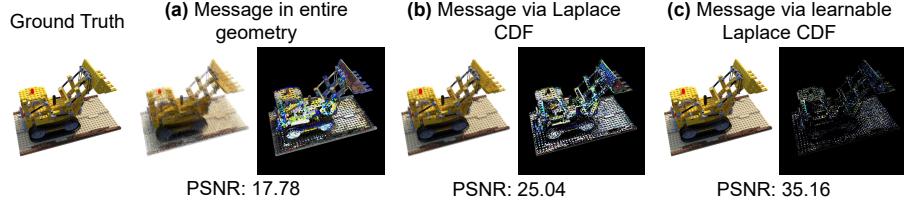


Fig. 6: Ablation study to our cover media selection. **(a)** indicates including all geometry points for messages sticking, which can cause obvious distortion on the scene surface. **(b)** indicates using Laplace CDF with a fixed threshold for message attachment, which reduces perturbation on the NeRF geometry but still shows noticeable distortion. **(c)** indicates using our learnable Laplace CDF can find an optimal threshold, making the visual distortion almost imperceptible. The visual quality results of PSNR are averaged on all examples in the selected scene.

As shown in Fig. 5, the bit accuracy starts to decrease if the model purification sacrifices the rendering quality, but it can keep a relatively high bit accuracy if we want to keep the rendered image quality. These results show that model purification is hard to reduce the bit accuracy significantly without sacrificing the image quality.

6 Conclusion

In our research, we introduce a novel approach for asserting ownership of recolorized NeRF models through the innovative GeometrySticker. By embedding binary messages within the high geometry value components of NeRF, we ensure that these messages remain robust even after recolorization. Comprehensive testing demonstrates that our method establishes ownership claims on recolorized NeRF models, which guarantees the safe application of NeRF recolorization across various scenarios, thereby ensuring positive societal impacts for protecting the copyrights of artists and creators.

Limitations and future work. Our method is an effective technical solution for copyright protection against the recolorization of NeRF models. However, as we discussed before, our mechanism may still face threats from some malicious operations. We will consider further enhancing the adversary robustness through adversary learning approaches [13, 16, 21]. Besides, we will explore enhancing the robustness of GeometrySticker towards geometry editing scenarios, enabling the manipulation of NeRF model and rendered images through techniques such as cage-based deformation [36] or motion transfer [6] in future work. Moreover, we will improve the GeometrySticker to align with the emerging 3D Gaussian Splatting (3DGS) method [9], which utilizes an explicit point cloud representation, distinguishing itself from the NeRF implicit neural representation. Our aim is to elevate the versatility of GeometrySticker to be compatible with different 3D representation baselines.

Acknowledgement This work was done at Renjie’s Research Group at the Department of Computer Science of Hong Kong Baptist University. Renjie’s Research Group is supported by the National Natural Science Foundation of China under Grant No. 62302415, Guangdong Basic and Applied Basic Research Foundation under Grant No. 2022A1515110692, 2024A1515012822, and the Blue Sky Research Fund of HKBU under Grant No. BSRF/21-22/16.

References

1. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2021)
2. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: TensoRF: Tensorial radiance fields. In: Proceedings of the European Conference on Computer Vision (ECCV) (2022)
3. Chen, X., Deng, Y., Wang, B.: Mimic3D: Thriving 3D-Aware GANs via 3D-to-2D Imitation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2023)
4. Cheng, Y., Wan, R., Weng, S., Zhu, C., Chang, Y., Shi, B.: Colorizing Monochrome Radiance Fields. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI) (2024)
5. Gong, B., Wang, Y., Han, X., Dou, Q.: RecolorNeRF: Layer Decomposed Radiance Field for Efficient Color Editing of 3D Scenes. In: Proceeding of the ACM International Conference on Multimedia (MM) (2023)
6. Huang, X.F., Po, L.M., Ou, W.F.: Motion transfer-driven intra-class data augmentation for finger vein recognition. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2024)
7. Kajiya, J.T., Von Herzen, B.P.: Ray tracing volume densities. ACM SIGGRAPH Computer Graphics (1984)
8. Kalita, M., Majumder, S.: Steganography using biometrics. In: Encyclopedia of Information Science and Technology (2018)
9. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3D Gaussian Splatting for Real-Time Radiance Field Rendering. ACM Transactions on Graphics (ToG) (2023)
10. Kuang, Z., Luan, F., Bi, S., Shu, Z., Wetzstein, G., Sunkavalli, K.: PaletteNeRF: Palette-based Appearance Editing of Neural Radiance Fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2023)
11. Levoy, M.: Efficient ray tracing of volume data. ACM Transactions on Graphics (ToG) (1990)
12. Li, C., Feng, B.Y., Fan, Z., Pan, P., Wang, Z.: StegaNeRF: Embedding Invisible Information within Neural Radiance Fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2023)
13. Li, J., Pang, M., Dong, Y., Jia, J., Wang, B.: Graph neural network explanations are fragile. In: International Conference on Machine Learning (ICML) (2024)
14. Lombardi, S., Simon, T., Saragih, J., Schwartz, G., Lehrmann, A., Sheikh, Y.: Neural Volumes: Learning Dynamic Renderable Volumes from Images. ACM Transactions on Graphics (ToG) (2019)

15. Luo, Z., Guo, Q., Cheung, K.C., See, S., Wan, R.: CopyRNeRF: Protecting the CopyRight of Neural Radiance Fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2023)
16. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083 (2017)
17. Mildenhall, B., Srinivasan, P.P., Ortiz-Cayon, R., Kalantari, N.K., Ramamoorthi, R., Ng, R., Kar, A.: Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. ACM Transactions on Graphics (ToG) (2019)
18. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM (2021)
19. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Transaction of Graph (ToG) (2022)
20. Ohbuchi, .R., Mukaiyama, .A., Takahashi, .S.: A frequency-domain approach to watermarking 3d shapes. In: Computer graphics forum (2002)
21. Pang, M., Wang, B., Ye, M., Cheung, Y.M., Zhou, Y., Huang, W., Wen, B.: Heterogeneous prototype learning from contaminated faces across domains via disentangling latent factors. IEEE Transactions on Neural Networks and Learning Systems (TNNLS) (2024)
22. Praun, E., Hoppe, H., Finkelstein, A.: Robust mesh watermarking. In: Proceedings of the Conference on Computer Graphics and Interactive Techniques (PACM-CGIT) (1999)
23. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: Proceeding of the International Conference on Machine Learning (ICML) (2021)
24. Sara Fridovich-Keil and Alex Yu, Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance fields without neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)
25. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
26. Son, J., Kim, D., Choi, H.Y., Jang, H.U., Choi, S.: Perceptual 3D Watermarking Using Mesh Saliency. In: Proceedings of International Conference on Information Science and Applications (ICISA) (2017)
27. Tancik, M., Mildenhall, B., Ng, R.: StegaStamp: Invisible Hyperlinks in Physical Photographs. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
28. Tancik, M., Weber, E., Ng, E., Li, R., Yi, B., Kerr, J., Wang, T., Kristoffersen, A., Austin, J., Salahi, K., Ahuja, A., McAllister, D., Kanazawa, A.: Nerfstudio: A Modular Framework for Neural Radiance Field Development. In: ACM SIGGRAPH Conference Proceedings (2023)
29. Tang, Y., Zhu, C., Wan, R., Xu, C., Shi, B.: Neural Underwater Scene Representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2024)
30. Van Schyndel, R.G., Tirkel, A.Z., Osborne, C.F.: A digital watermark. In: Proceedings of International Conference on Image Processing (ICIP) (1994)
31. Wang, C., Chai, M., He, M., Chen, D., Liao, J.: CLIP-NeRF: Text-and-Image Driven Manipulation of Neural Radiance Fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)

32. Wang, R., Wan, R., Guo, Z., Guo, Q., Huang, R.: Spy-Watermark: Robust Invisible Watermarking for Backdoor Attack. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2024)
33. Weng, X., Li, Y., Chi, L., Mu, Y.: High-Capacity Convolutional Video Steganography with Temporal Residual Modeling. In: Proceedings of the International Conference on Multimedia Retrieval (ICMR) (2019)
34. Wengrowski, E., Dana, K.: Light Field Messaging with Deep Photographic Steganography. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
35. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3D ShapeNets: A deep representation for volumetric shapes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
36. Xu, T., Harada, T.: Deforming radiance fields with cages. In: Proceedings of the European Conference on Computer Vision (ECCV) (2022)
37. Yang, P., Lao, Y., Li, P.: Robust watermarking for deep neural networks via bi-level optimization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2021)
38. Yoo, I., Chang, H., Luo, X., Stava, O., Liu, C., Milanfar, P., Yang, F.: Deep 3D-to-2D Watermarking: Embedding Messages in 3D Meshes and Extracting Them from 2D Renderings. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)
39. Zhang, C., Benz, P., Karjauv, A., Sun, G., Kweon, I.S.: UDH: Universal deep hiding for steganography, watermarking, and light field messaging. In: Advances in Neural Information Processing Systems (NeurIPS) (2020)
40. Zhang, K.A., Xu, L., Cuesta-Infante, A., Veeramachaneni, K.: Robust invisible video watermarking with attention. arXiv preprint arXiv:1909.01285 (2019)
41. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceeding of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
42. Zhu, C., Wan, R., Shi, B.: Neural Transmitted Radiance Fields. In: Advances in Neural Information Processing Systems (NeurIPS) (2022)
43. Zhu, C., Wan, R., Tang, Y., Shi, B.: Occlusion-Free Scene Recovery via Neural Radiance Fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2023)
44. Zhu, J., Kaplan, R., Johnson, J., Fei-Fei, L.: HiDDeN: Hiding data with deep networks. In: Proceedings of the European Conference on Computer Vision (ECCV) (2018)

Supplementary Material: GeometrySticker: Enabling Ownership Claim of Recolorized Neural Radiance Fields

Xiufeng Huang^{1,2}, Ka Chun Cheung², Simon See², and Renjie Wan^{1*}

¹ Department of Computer Science, Hong Kong Baptist University

² NVIDIA AI Technology Center, NVIDIA

xiufenghuang@life.hkbu.edu.hk, {chcheung, ssee}@nvidia.com,
renjiewan@hkbu.edu.hk

1 Overview

This supplementary document provides more discussions, implementation details, and further results that accompany the paper:

- Section 2 explains the uniqueness of our method by comparing with the current NeRF ownership claiming methods under NeRF recolorizations.
- Section 3 explains the effectiveness of applying the Laplace Cumulative Distribution Function (CDF) with learnable parameters.
- Section 4 introduces the details of our reference colors and visualizes their corresponding recolorization results for NeRF. These recolorization methods are applied to different NeRF architectures to validate ownership for the recolorized NeRF.
- Section 5 presents the implementation details of our method, including the network architectures and the training process.
- Section 6 provides additional results, including additional qualitative results of the main paper.

2 Uniqueness

As shown in Fig. 1, we demonstrate the uniqueness of using our GeometrySticker to claim ownership of a recolorized NeRF model. The current ownership protection methods such as CopyRNeRF [15] and StegaNeRF [12] can only claim the ownership when recolorization is not conducted. However, since the recent developments of NeRF recolorization methods [5, 10, 31], if a model owner *Bob* creates a NeRF model and watermark the model with CopyRNeRF [15] or StegaNeRF [12], the hidden ownership information could be vulnerable when a malicious user applies unauthorized recolorization on the NeRF model. Our GeometrySticker can be robust under different recolorizations. A model owner *Alice* can watermark her NeRF model by GeometrySticker, which can keep the hidden information consistent under different recolorizations and reliably extract the binary message from the recolorized NeRF renderings.

* Corresponding author.

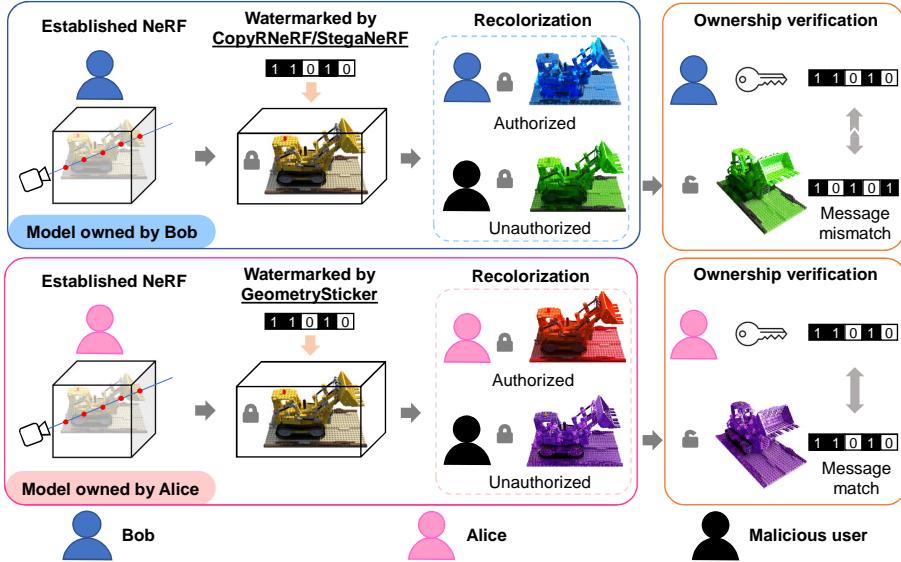


Fig. 1: Illustration of the uniqueness of our method. The first row illustrates the NeRF model owner *Bob* claims the ownership by using CopyRNeRF [15] or StegaNeRF [12]. However, when a malicious user applies unauthorized recolorization on *Bob*'s model, the hidden ownership information can be corrupted and mismatch the original secret messages. The second row illustrates the NeRF model owner *Alice* claims the ownership by GeometrySticker. The NeRF model watermarked by GeometryStricker can be robust to different recolorizations. Even if a malicious user applies unauthorized recolorization on *Alice*'s model, the hidden ownership information can still be reliably extracted and match the original secret messages.

3 Learnable Laplace CDF

We provide more ablation studies for our learnable Laplace CDF used for the selection of cover medium. As shown in Fig. 2, we calculate the mean μ and deviation β of the geometry values and use the Laplace distribution to model the geometry values distribution of a selected scene. As shown in Fig. 2 (a), attaching messages to all NeRF geometry values can cause obvious distortion since the low geometry values take up the majority of the entire NeRF geometry. We apply the Laplace CDF with the fixed parameters μ and β and the CDF value $\psi = 0.99$ as the threshold to filter large geometry values for messages attachment. As shown in Fig. 2 (b), applying Laplace CDF with calculated parameters can reduce perturbation but still show noticeable distortion. As shown in Fig. 2 (c), our learnable Laplace CDF can adaptatively find an optimized deviation parameter β to adjust the CDF threshold ($\psi = 0.99$) for the selection of cover medium and finally make the perturbation caused by the attached messages imperceptible.

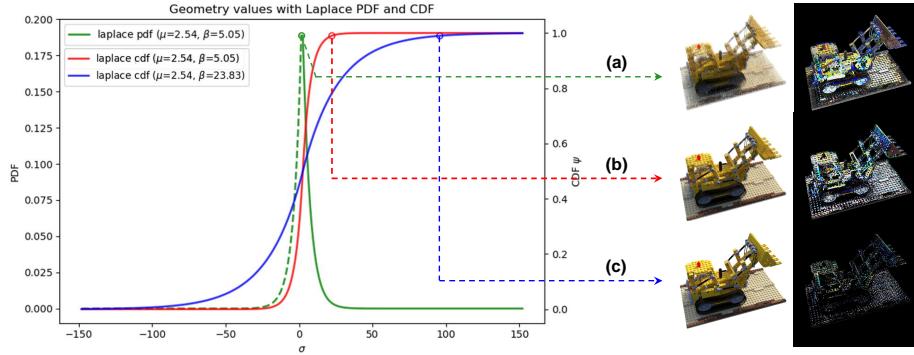


Fig. 2: Message attachment into NeRF geometry values by applying Laplace CDF with different deviation parameters. The geometry values distribution is modeled by a Laplace distribution with the mean μ and deviation β . (a) indicates directly attaching messages on all geometry values can cause obvious distortion. (b) indicates applying Laplace CDF with fixed μ and β can reduce perturbation but still show noticeable distortion. (c) indicates applying Laplace CDF with a learnable deviation parameter β can find an optimized threshold for filtering 3D points and make the distortion imperceptible.

4 More details on recolorization

We select 10 reference colors from the Standard sRGB / Rec.709 color gamut including green, yellow, orange, red, pink, magenta, purple, blue, dodger blue, cyan. We recolorize the NeRF models by using colors' name for the CLIP-based method or assigning RGB values for the palette-based method. We also convert the NeRF renderings into HSV format to recolorize the images by changing the hue channel. As shown in Fig. 3a, the palette-based method can precisely recolorize NeRF by editing the palette's colors to the reference colors. As shown in Fig. 3b, though the CLIP-based method can roughly conduct the recolorization via the text prompts, the results are uncontrollable since the recolorization under the same prompts may have some differences as shown in Fig. 3d. Thus, it is hard to get the same results for an unwatermarked NeRF model and a watermarked NeRF model. As shown in Fig. 3c, color-jittering is an image-level recolorization by converting images into HSV format and shifting the intensity of the hue channels in a scale of $[-0.5, 0.5]$. For a fair comparison across different baselines, we only use color-jittering in our reconstruction quality computation for PSNR/SSIM and LPIPS in the main manuscript Table 1, since CLIP-based recolorization is uncontrollable and palette-based recolorization is not applicable to CopyRNeRF [15] and StegaNeRF [12]. All the testing set images in the main manuscript Section 5.1 are recolored for computing reconstruction quality or message extraction bit accuracies.



Fig. 3: Recolorization results by using different methods: (a) is the Palette-based recolorization. The first column is the reference image with the original color palette, and others are recolorized by assigning reference colors to different base colors in the color palette. (b) is the CLIP-based recolorization. The first column is the reference image, and the others are recolorized by using the colors' name as the text prompt. (c) is the color-jittering recolorization. The first column is the original image and others are recolorized by changing the hue of the original image by shifting the intensities with the range of $[-0.5, 0.5]$ in the hue channel. (d) indicates CLIP-based recolorization with the same text prompt can have different results.

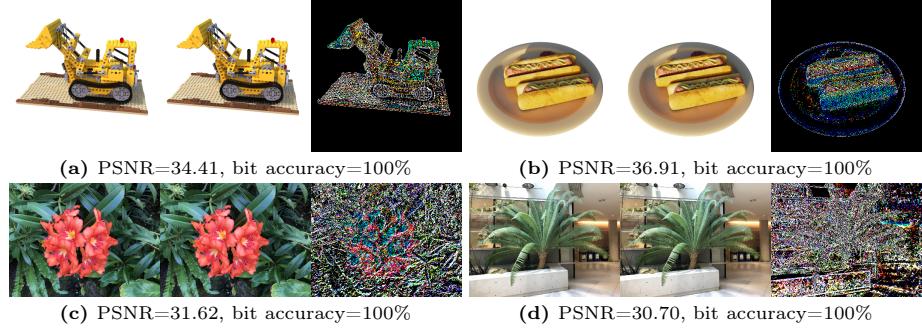


Fig. 4: Additional results for different scenes. The message length is 48 bits. We visualize the residual maps between the unwatermarked renderings and the watermarked renderings. From left to right: unwatermarked, GeometrySticker, residual maps ($\times 10$).

5 Implementation details

5.1 Network architectures

In our proposed GeometrySticker, the message sticker Θ_m is an MLP layer. In specific, it has 80 input channels, which are a concatenation of the message M in 48 dimensions and positional encoding $\gamma_x(\mathbf{x})$ in 32 dimensions. The message sticker Θ_m has two hidden layers with 64 dimensions and 1-dimensional output for the message embedding m . For the message extractor D_χ , we use the VGG16 network [25] as the backbone feature extractor. An average pooling is then performed, followed by a final linear layer with a fixed output dimension N_b to produce the continuous predicted message \hat{M} . For the watermark classifier C_ϕ , we use a similar architecture with the message extractor D_χ with the VGG16 network [25] as the feature extractor followed by an average pooling layer and a final 1-dimensional layer for classification.

5.2 Training process

The training process consists of two stages. In the first stage, we establish a NeRF scene by optimizing Θ_σ and Θ_c to get the geometry and color values of the scene according to \mathcal{L}_{cont} . In the second stage, we keep the geometry MLP Θ_σ and color MLP Θ_c unchanged and train the message sticker Θ_m and Laplace CDF with the learnable deviation parameter β for message attachment and key points selection. Meanwhile, we train a message extractor D_χ to extract the hidden message from the 2D watermarked renderings. In addition, we also train the watermarking classifier C_ϕ to classify whether the NeRF renderings contain watermarkings or not. The \mathcal{L}_{cont} is measured by the mean squared error between the watermarked rendering images and the ground truth images. The \mathcal{L}_{msg} is a binary cross entropy loss calculated between the embedded messages M and the extracted messages \hat{M} . The \mathcal{L}_{cls} is a binary cross entropy loss calculated between

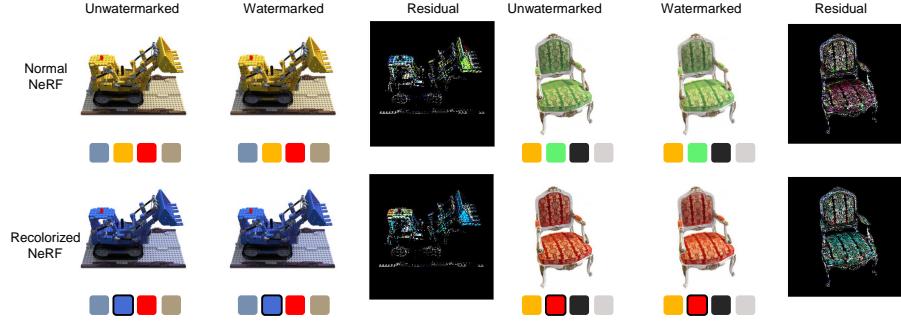


Fig. 5: Residual maps for NeRF renderings before and after palette-based recolorizations. The first row shows the residual maps before palette-based recolorizations. The second row shows the residual maps after palette-based recolorizations. Each residual map shows the differences between the unwatermarked renderings and watermark renderings by GeometrySticker.

the watermarked rendering image \mathbf{I}_w and the unwatermarked rendering images \mathbf{I}_u . \mathcal{L}_{sparse} is the sparsity loss [14] to force the CDF value ψ to be close to either zero or one. The network Θ_m and parameters χ , ϕ and β are optimized with the objective functions \mathcal{L}_{cont} , \mathcal{L}_{msg} , \mathcal{L}_{cls} and \mathcal{L}_{sparse} . In every training loop, we attach the message \mathbf{M} with a random camera pose and apply 2D distortions on the watermarked rendering images.

6 Additional results

We provide additional results to validate the effectiveness of our GeometrySticker. As shown in Fig. 4, we evaluate the qualitative and quantitative results of the reconstruction quality and bit accuracies of our GeometrySticker on the selected scene. The watermarked rendered images have high reconstruction quality with minimal discrepancies compared with the original rendered images. From the residual maps, we can observe that the hidden messages are sparsely embedded into the geometrical structure of the object or scene.

We further validate the consistency of our GeometrySticker under different recolorizations. As shown in Fig. 5, the message perturbation attached by GeometrySticker remains consistent from non-recolored NeRF models to recolored NeRF models. These results show our method successfully embeds secret messages into the geometry representation and disentangles them with the color representation, thus claiming ownership under various NeRF recolorizations.