

GuardSplat: Efficient and Robust Watermarking for 3D Gaussian Splatting

Zixuan Chen¹ Guangcong Wang² Jiahao Zhu¹ Jianhuang Lai^{1,3,4} Xiaohua Xie^{1,3,4*}

¹School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China

²School of Computing and Information Technology, Great Bay University, Dongguan, China

³Guangdong Province Key Laboratory of Information Security Technology, Guangzhou, China

⁴Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, China

chenzx3@mail2.sysu.edu.cn, wanggc3@gmail.com, zhuhj59@mail2.sysu.edu.cn, {stsljh, xiexiaoh6}@mail.sysu.edu.cn

Project Page: <https://narcissusex.github.io/GuardSplat>

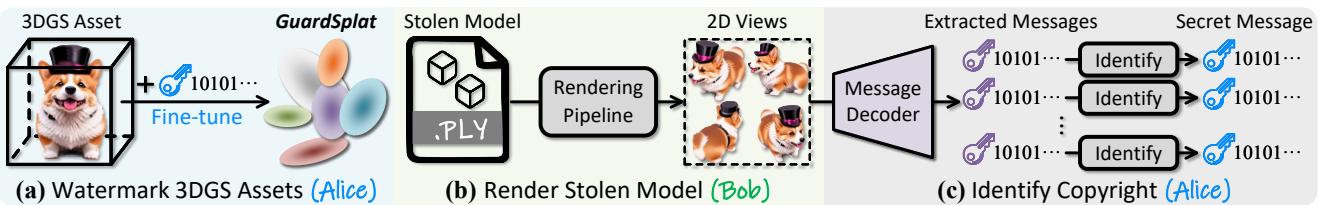


Figure 1. Application scenarios of **GuardSplat**. To protect the copyright of 3D Gaussian Splatting (3DGS) [17] assets, (a) the owners (*Alice*) can use our **GuardSplat** to embed the secret message (blue key) into these models. (b) If malicious users (*Bob*) render views for unauthorized uses, (c) *Alice* can use the private message decoder to extract messages (purple key) for copyright identification.

Abstract

3D Gaussian Splatting (3DGS) has recently created impressive assets for various applications. However, the copyright of these assets is not well protected as existing watermarking methods are not suited for 3DGS considering security, capacity, and invisibility. Besides, these methods often require hours or even days for optimization, limiting the application scenarios. In this paper, we propose **GuardSplat**, an innovative and efficient framework that effectively protects the copyright of 3DGS assets. Specifically, 1) We first propose a CLIP-guided Message Decoupling Optimization module for training the message decoder, leveraging CLIP's aligning capability and rich representations to achieve a high extraction accuracy with minimal optimization costs, presenting exceptional **capability** and **efficiency**. 2) Then, we propose a Spherical-harmonic-aware (SH-aware) Message Embedding module tailored for 3DGS, which employs a set of SH offsets to seamlessly embed the message into the SH features of each 3D Gaussian while maintaining the original 3D structure. It enables the 3DGS assets to be watermarked with minimal fidelity trade-offs and prevents malicious users from removing the messages from the model files, meeting the demands for **invisibility** and **security**. 3) We further propose an Anti-distortion Message Extraction module to improve **robustness** against various visual distor-

tions. Extensive experiments demonstrate that **GuardSplat** outperforms the state-of-the-art methods and achieves fast optimization speed. The code will be publicly available.

1. Introduction

3D representation is a cutting-edge technique in computer vision and graphics, playing a vital role in various domains such as film production, game development, virtual reality, and autonomous driving. One of the most promising approaches in this field is 3D Gaussian Splatting (3DGS) [17]. 3DGS revolutionizes 3D representation techniques by offering high fidelity, rapid optimization capabilities, and real-time rendering speed, which enables the creation of impressive 3D assets [10, 21, 44, 51, 55, 56] in the real world. However, the risk of valuable 3DGS assets being stolen by unauthorized users poses significant losses to creators. This situation raises an urgent question: *How can we design a method tailored for 3DGS to protect copyright?*

One effective strategy for copyright protection is embedding secret messages into 3DGS assets. However, it has several challenging requirements: **1) Security:** A secure watermark should be difficult to detect and cannot be removed from the model. **2) Invisibility:** Any watermarked view rendered from a 3D asset should visually maintain consistency with the corresponding view rendered from original models, avoiding disruption of normal use. **3) Capacity:**

*Corresponding author.

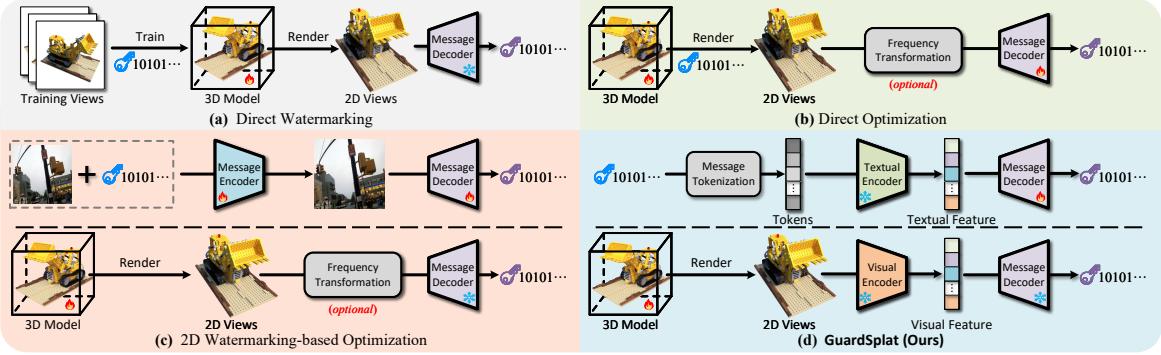


Figure 2. **Comparisons of four 3D watermarking frameworks.** These frameworks differ in how to embed messages and how to train message decoders. **(a)** Directly training 3D models on the watermarked images. **(b)** Directly embedding messages and training a message decoder for extraction. **(c)** Employing the message decoder from a 2D watermarking network for optimization. **(d)** **GuardSplat** first trains a message decoder to extract messages from CLIP [35] textual features. This message decoder then can be applied to the CLIP visual features for watermarking 3D models via optimization.

Large-capacity messages can be effectively embedded into a 3DGS model and accurately extracted from 2D rendered views. **4) Efficiency:** Fast optimization speed is essential to meet real-world demands. When considering all of these four important requirements, it is challenging to design a good 3D watermarking method for 3DGS.

Existing watermarking methods [13, 24, 59, 60] partially meet the four requirements above and have improved a lot for the watermark of 2D or 3D digital assets. However, they are inadequate for the 3DGS framework, which can be categorized into four groups. **First**, an intuitive method is directly applying 2D watermarking methods for 2D training or rendered views. For instance, employing HiDDeN [60] to watermark the 2D rendered views cannot protect the original model files. Besides, directly training 3D models on the watermarked images shown as Figure 2 (a) exhibits low bit accuracy in message extraction, because this pipeline does not guarantee novel views contain a consistent watermark. **Second**, some methods directly embed messages into 3D models during optimization (*e.g.*, [24, 59], Figure 2 (b)). Although this pipeline guarantees novel views that contain a consistent watermark, it needs to optimize a new message decoder per scene during watermarking 3D models, requiring expensive optimization costs. **Third**, to avoid per-scene optimization, one might consider pre-training a general-purpose message decoder (*e.g.*, [12–14], Figure 2 (c)). Given an image and a message, they first train an encoder-decoder model before finetuning the 3D representations. This encoder-decoder model aims to reconstruct the image and extract the message, where the encoder tries to keep the strict consistency between the input and output images while the decoder subsequently tries to extract the messages as intact as possible from the output image. As a result, directly using the decoder to watermark 3D models may yield degraded performance due to the fidelity-capacity trade-off. Moreover, simultaneously optimizing both en-

coder and decoder is time-consuming.

In this paper, we present *GuardSplat*, a novel framework that can effectively protect the copyright of 3DGS assets. As shown in Figure 2 (d), compared to conventional 2D watermarking methods that optimize both the message encoder and decoder simultaneously, we propose a message decoupling optimization module guided by Contrastive Language-Image Pre-training (CLIP) [35], which can only train the decoder for message extraction (top row), significantly reducing the optimization costs. Thanks to the text-image aligning capability and rich representations of CLIP, this message decoder can embed large-capacity messages into 3DGS assets (bottom row) with minimal optimization costs (see Figure 3), demonstrating superior **capacity** and **efficiency**. Subsequently, we tailor a watermark embedding module for 3DGS, employing a set of spherical harmonics (SH) offsets to learn the watermarked SH features during optimization. It can seamlessly embed messages into the SH features of each 3D Gaussian while maintaining the original 3D structure, watermarking 3DGS assets with minimal trade-offs in fidelity. It also prevents malicious users from removing the watermarks from the model files, meeting the demands for **invisibility** and **security**. We further propose an anti-distortion message extraction module, which employs a differentiable distortion layer to simulate the randomly-distorted views during optimization, enabling the watermarked SH features to have strong **robustness** against various types of distortions. Extensive experiments on Blender [27] and LLFF [26] datasets demonstrate that our *GuardSplat* outperforms the state-of-the-art methods. *GuardSplat* achieves fast training speed, which takes 5 and 20 minutes to train the decoder and watermark a 3DGS asset on a single RTX 3090 GPU, respectively. Overall, the main contributions of this paper are summarized as follows:

- We propose *GuardSplat*, a new 3D watermarking approach that can effectively protect the copyright of 3DGS

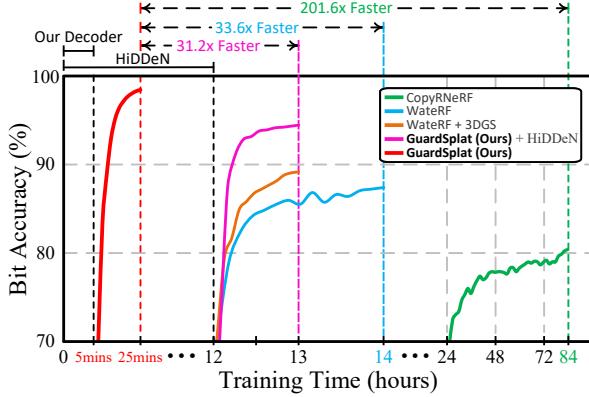


Figure 3. **Training accuracy curves with $N_L = 32$ bits on Blender [27] dataset.** Compared to existing methods that require hours or even days for optimization, our **GuardSplat** achieves much better efficiency, which only takes 5 and 20 minutes to train the message decoder and watermark a 3DGS asset, respectively.

assets in the real world.

- We propose a CLIP-guided Message Decoupling Optimization module to train the message decoder, achieving superior **capacity** and **efficiency**. We tailor a watermark embedding method for 3DGS to meet the **invisibility** and **security** demands. We further propose an anti-distortion message extraction for good **robustness**.
- Experimental results demonstrate that our *GuardSplat* significantly outperforms the state-of-the-art methods and achieves fast optimization speed for training message decoder and watermarking 3DGS assets.

2. Related Works

3D Representations. Neural radiance field (NeRF) [27] is a compelling solution for 3D representations, which is based on the standard volumetric rendering [16] and alpha compositing techniques [33], building the implicit representations using the multi-layer perceptron (MLP). Follow-up works adapt NeRF to various domains, such as sparse-view reconstruction [8, 47, 54], acceleration [28], generative modeling [29, 40], text-to-3D generation [32, 49], anti-aliasing [3, 4], medical image super resolution [7], and RGB-D scene synthesis [1]. 3D Gaussian Splatting (3DGS) [17] has become a mainstream approach for 3D representations with its fast optimization and rendering speed. Unlike NeRF, 3DGS explicitly represents the scene using a set of 3D Gaussians and renders views through splatting [19]. It has been applied to various scenarios and created numerous assets, including text-to-3D generation [21, 45, 52], avatar generation [41, 56], single-view generation [44, 61], anti-aliasing [55], and SLAM [25, 51].

Digital Watermarking. Early studies [2, 20, 36, 46] proposed to embed the watermarks within frequency domains.



Figure 4. **Visual examples of CopyRNeRF [24], WaterRF [13], WaterRF + 3DGS, and our **GuardSplat**.** Heatmaps at the bottom show the differences ($\times 10$) between the watermarked and original views. Red text indicates the best performance.

With the advent of deep learning, Zhu *et al.* [60] proposed the first end-to-end deep watermarking framework – HiDDeN, while the subsequent advances investigate to improve the robustness [22], and extend the application scenarios [23, 57]. Recent methods [9, 50] proposed diffusion-based watermarking to protect the contents yielded from diffusion models [11, 38, 39, 42]. To protect 3D assets, most methods [30, 34] focused on embedding and detecting watermarks within meshes and point clouds. Yoo *et al.* [53] provided a novel perspective, which embedded the invisible watermarks into 3D models through differentiable rendering pipelines, allowing the watermarks to be extracted from rendered views. Inspired by [53], CopyRNeRF [24] and WaterRF [13] aim to insert watermarks into NeRF [27]. Specifically, CopyRNeRF [24] replaces the color representations, while WaterRF [13] embeds the message into model weights via optimization. Recently, Song *et al.* [43] propose to prevent using Triplane Gaussian Splatting (TGS) [61] for unauthorized 3D reconstruction from copyrighted images. Zhang *et al.* [59] introduced a steganography model for 3DGS that employs secured features to replace SH features, and trains scene and message decoders to extract views and hidden messages, respectively. Current works [12, 14] aim to directly embed messages into 3DGS models via a HiDDeN decoder. However, since HiDDeN has an inherent trade-off between fidelity and capacity, using its decoder for optimization may result in sub-optimal capacity. Moreover, [12, 14] may significantly alter the 3D structure during watermarking, leading to low-fidelity results.

3. Preliminary

3D Gaussian Splatting. 3DGS [17] is a recent ground-breaking method for novel view synthesis. Given the center position $\mu \in \mathbb{R}^3$ and covariance $\Sigma \in \mathbb{R}^7$, a 3D Gaussian at position x can be queried as follows:

$$\mathcal{G}(x : \mu, \Sigma) = \exp\left(-\frac{1}{2}(x - \mu)^\top \Sigma^{-1}(x - \mu)\right). \quad (1)$$

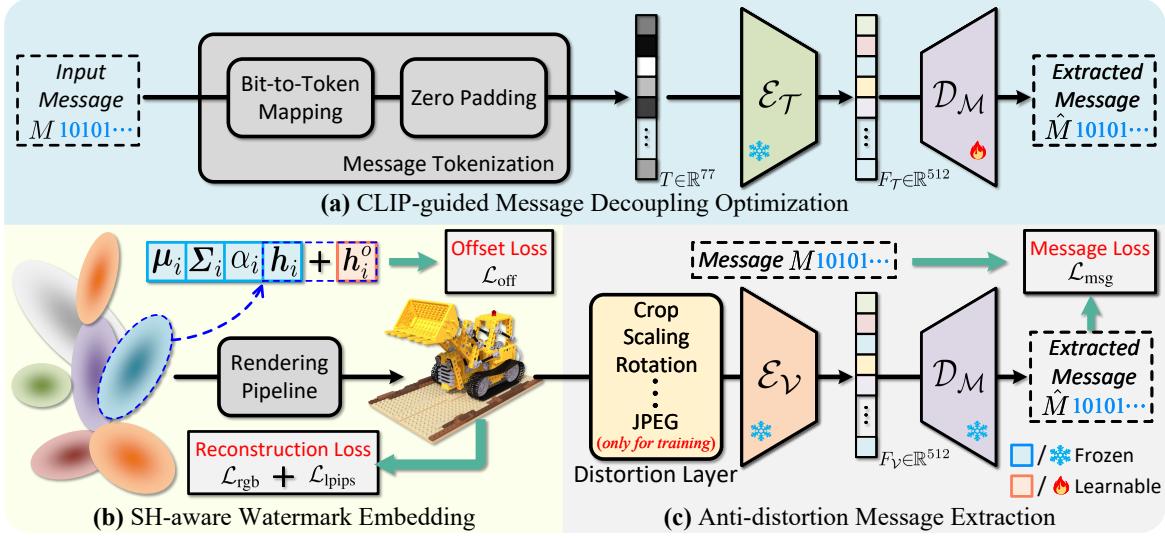


Figure 5. **Overview of GuardSplat.** (a) Given a binary message $M \in \{0, 1\}_{i=1}^L$, we first transform it into CLIP tokens T using the proposed message tokenization. We then employ CLIP’s textual encoder \mathcal{E}_T to map T to the textual feature F_T . Finally, we feed F_T into message decoder \mathcal{D}_M to extract the message $\hat{M} \in \{0, 1\}_{i=1}^L$ for optimization. (b) For each 3D Gaussian, we freeze all the attributes and build a learnable spherical harmonic (SH) offset h_i^o as the watermarked SH feature, which can be added to the original SH features as $h_i + h_i^o$ to render the watermarked views. (c) We first feed the 2D rendered views to CLIP’s visual encoder \mathcal{E}_V to acquire the visual feature F_V and then employ the pre-trained message decoder to extract the message \hat{M} . A differentiable distortion layer is used to simulate various visual distortions during optimization. \mathcal{D}_M and h_i^o are optimized by Eq. (7) and Eq. (10), respectively.

Then, given the projective transformation \mathbf{P} , viewing transformation \mathbf{W} , and Jacobian \mathbf{J} of the affine approximation of \mathbf{P} , the corresponding 2D mean position $\hat{\mu}$ and covariance $\hat{\Sigma}$ of the projected 3D Gaussian can be calculated as:

$$\hat{\mu} = \mathbf{P}\mathbf{W}\mu, \quad \hat{\Sigma} = \mathbf{J}\mathbf{W}\Sigma\mathbf{W}^\top\mathbf{J}^\top. \quad (2)$$

Let $\hat{C} \in \mathbb{R}^{W \times H \times 3}$ denote a $W \times H$ RGB view rendered by 3DGS, the color of each pixel (x, y) can be generated as:

$$\hat{C}_{x,y} = \sum_{i=1}^N \mathbf{c}_i \sigma_i \prod_{j=1}^{i-1} (1 - \sigma_j), \quad \sigma_i = \alpha_i \mathcal{G}((x, y); \hat{\mu}, \hat{\Sigma}), \quad (3)$$

where N represents the number of Gaussians overlapping the pixel (x, y) . $\mathbf{c}_i \in \mathbb{R}^3$ and $\alpha_i \in \mathbb{R}^1$ denote the color transformed from k -ordered spherical harmonic (SH) coefficients $\mathbf{h}_i \in \mathbb{R}^{3 \times (k+1)^2}$ and opacity of the i -th Gaussian, respectively. Since the rendering pipeline is differentiable, 3DGS models can be optimized by the loss function as:

$$\mathcal{L}_{\text{rgb}} = \lambda_{\text{ssim}} \mathcal{L}_{\text{ssim}}(\hat{C}, C) + (1 - \lambda_{\text{ssim}}) \mathcal{L}_1(\hat{C}, C), \quad (4)$$

where C is a groundtruth image and λ_{ssim} is set to 0.2.

Contrastive Language-Image Pre-training. Contrastive Language-Image Pre-training (CLIP) [35] is pre-trained to match images with their corresponding natural language descriptions on 400 million training image-text pairs collected from the internet. It consists of two independent encoders:

a textual encoder \mathcal{E}_T and a visual encoder \mathcal{E}_V . Given a batch of images and texts, CLIP extracts textual features $F_T \in \mathbb{R}^{512}$ and visual features $F_V \in \mathbb{R}^{512}$ through the textual and visual encoders, respectively. These encoders are trained to learn the aligning capability of text-image pairs by maximizing the similarity between textual and visual features via a contrastive loss.

4. Method

In this section, we propose *GuardSplat* to effectively protect the copyright of 3D Gaussian Splatting (3DGS) [17] assets. The overview of the proposed method is depicted in Figure 5. Specifically, we first propose a message decoupling optimization module guided by Contrastive Language-Image Pre-training (CLIP) [35] to train the message decoder \mathcal{D}_M . By analyzing 3DGS rendering pipeline, we then present a spherical-harmonic-aware (SH-aware) watermark embedding approach to integrate the watermarked SH features for a pre-trained 3DGS model. Furthermore, we design a strategy for anti-distortion message extraction. As a result, *GuardSplat* can not only embed large-capacity messages with minimal optimization costs, but also achieves superior fidelity and strong robustness, suggesting that *GuardSplat* can protect the copyright of 3DGS models without the affection of normal use.

4.1. CLIP-guided Message Decoupling Optimization

As discussed in recent works [13, 24, 53], one of the representative approaches to watermark 3D assets is embedding the messages into a 3D representation model via optimization. However, these methods encounter limitations in efficiency. Specifically, one group of methods aims to directly embed messages into 3D models for optimization, as shown in Figure 2 (b). They optimize a message decoder per scene, which is time-consuming. To learn a general-purpose message decoder, the other group of methods first trains an encoder and decoder such that it can reconstruct the image and extract the message given an image and message as input. The message decoder is then used for 3D watermarking as shown in Figure 2 (c). Since conventional methods optimize both the message encoder and decoder simultaneously, it also takes much time for optimization.

To address this issue, we propose a CLIP-guided Message Decoupling Optimization module that optimizes a general-purpose message decoder and a 3DGS model, as shown in Figure 3. The key to the success of this module is that the Contrastive Language-Image Pre-training (CLIP) [35] builds a bridge between the texts and images. As discussed in Section 3, CLIP consists of a good textual encoder and a good visual encoder that is trained on a dataset of 400 million text-image pairs, providing rich text and image representations. As shown in Figure 5 (a), given a binary message $M \in \{0, 1\}_{i=1}^L$, we first transform it into CLIP tokens T using the proposed bit-to-token mapping as:

$$T = \{t_S\} \cup \left\{ \bigcup_{i=1}^L \Phi(M_i, i) \right\} \cup \{t_E\}, \quad (5)$$

where $t_S = 49406$ and $t_E = 49407$ denote the start and ending points of CLIP text token, respectively. $\Phi(\cdot, \cdot)$ is a function that uniformly maps the i -th bit to an integer number within the range [1, 49405]. To match the format of CLIP tokens, T is then zero-padded to a size of 77. Finally, we feed the tokens into CLIP textual encoder \mathcal{E}_T and employ a message decoder \mathcal{D}_M built by multi-layer perception (MLP) with 3 fully-connected (FC) layers to extract the messages \hat{M} from the output textual features $F_T \in \mathbb{R}^{512}$ as:

$$\hat{M} = \mathcal{D}_M(\mathcal{E}_V(T)). \quad (6)$$

This message decoder can be optimized by minimizing the message loss \mathcal{L}_{msg} as:

$$\mathcal{L}_{\text{msg}} = - \sum_{i=1}^L M_i \log \hat{M}_i + (1 - M_i) \log(1 - \hat{M}_i). \quad (7)$$

As a result, the message decoder can be directly optimized without being constrained by the invisibility, and thereby

achieves superior capacity and training efficiency. Furthermore, CLIP’s rich representation also enables it to achieve better performance with minimal optimization costs.

4.2. Spherical-harmonic-aware Watermark Embedding

Unlike neural radiance fields (NeRF) [27], 3D Gaussian Splatting (3DGS) [17] explicitly represents the scene through a set of 3D Gaussian as Eq. (1) and renders the views using Eq. (3). Given the i -th 3D Gaussian, it consists of 4 attributes: center position μ_i , covariance Σ_i , opacity α_i , and spherical harmonic (SH) features \mathbf{h}_i , where the former three attributes denote the 3D structure and while the latter is related to the color representation. An intuitive solution to watermark 3DGS assets is directly updating all the attributes of 3D Gaussians during optimization. However, it may significantly alter the 3D structure (i.e., μ_i , Σ_i , α_i), which leads to sub-optimal fidelity while concealing large-capacity messages (see “Offset_{all}” in Table 3).

Based on the above observations, we argue that it is required to maintain the original 3D structure during optimization. To achieve this, we propose an SH-aware Watermark Embedding module, a simple yet efficient approach tailored for 3DGS to watermark pre-trained models with minimal losses in fidelity. As shown in Figure 5 (b), for each 3D Gaussian, we freeze all the attributes and create a learnable SH offset $\mathbf{h}_i^o \in \mathbb{R}^{48}$ for watermarking. The reason behind this is the fact that SH parameters represent view-dependent effects like glossy or specular highlights, which only exist in a few regions of a scene. Thus, embedding the secret message into SH features with minimal constraints can preserve the fidelity of the 3D asset. Specifically, we first add each SH offset \mathbf{h}_i^o to the corresponding SH coefficient \mathbf{h}_i as the watermarked SH feature, and then feed it into the 3DGS rendering pipeline to render the watermarked views. To further alleviate the decline in fidelity brought by excessive offset, we employ an offset loss \mathcal{L}_{off} to constrain its magnitude by:

$$\mathcal{L}_{\text{off}} = -\frac{1}{N} \sum_{i=1}^N \|\mathbf{h}_i^o\|_2^2, \quad (8)$$

where N denotes the number of 3D Gaussians. As a result, the secret message can be seamlessly embedded into the learnable SH offset of each 3D Gaussian via optimization without altering the original 3D structure, which exhibits high fidelity and prevents malicious users from removing the watermarks from the model files, achieving superior invisibility and security.

4.3. Anti-distortion Message Extraction

Given CLIP’s visual encoder \mathcal{E}_V and the message decoder \mathcal{D}_M pre-trained in Section 4.1, we can directly extract the

Table 1. Comparisons of the start-of-the-art methods on Blender [27] and LLFF [26] datasets for bit accuracy and reconstruction qualities w.r.t various message lengths. **Bold** and underline texts indicate the best and second-best performance.

Methods	16 bits				32 bits				48 bits			
	Bit Acc	PSNR	SSIM	LPIPS	Bit Acc	PSNR	SSIM	LPIPS	Bit Acc	PSNR	SSIM	LPIPS
CopyRNeRF [24]	91.16	26.29	0.9100	0.0380	78.08	26.13	0.8960	0.0410	60.06	27.56	0.8950	0.0660
WateRF [13]	95.67	32.79	0.9480	0.0330	88.58	31.19	0.9360	0.0400	85.82	30.86	0.9300	0.0400
WateRF [13] + 3DGS	92.89	31.01	0.9678	0.0475	90.15	29.56	0.9611	0.0492	87.30	29.13	0.9562	0.0534
HiDDeN [60] + 3DGS [17]	63.07	31.59	0.9890	0.0171	52.46	34.51	0.9882	0.0209	53.08	33.42	0.9787	0.0299
GuardSplat (Ours) + HiDDeN [60]	<u>98.75</u>	<u>40.48</u>	<u>0.9909</u>	<u>0.0025</u>	<u>95.58</u>	<u>38.32</u>	<u>0.9897</u>	<u>0.0025</u>	<u>93.29</u>	<u>38.56</u>	<u>0.9886</u>	<u>0.0032</u>
GuardSplat (Ours)	99.64	41.55	0.9957	0.0017	99.04	39.40	0.9939	0.0022	98.29	38.90	0.9923	0.0028

hidden messages \hat{M} from the watermarked views \hat{C} based on CLIP’s aligning capability by:

$$\hat{M} = \mathcal{D}_{\mathcal{M}}(\mathcal{E}_{\mathcal{V}}(\hat{C})), \quad (9)$$

Thanks to the generalization capability of CLIP, the watermarked color features have strong robustness against the visual distortions of Gaussian blur, and Gaussian noise. However, it struggles to deal with other visual distortions such as rotation and JPEG compression. To robustly extract messages from the watermarked views across various types of distortions, we further propose an anti-distortion message extraction module. In Figure 5 (c), we employ a differentiable distortion layer to randomly simulate some visual distortions on the rendered views during optimization. These distortions include cropping, scaling, rotation, JPEG compression, and brightness jittering, enabling the watermarked SH features to learn the anti-distortion ability against most visual distortions via optimization. The distortion layer is only used during training. After optimization, *GuardSplat* can deal with most visual distortions, achieving superior extraction accuracy under challenging conditions.

4.4. Full Objective

For optimization, we first freeze CLIP’s visual and textual encoders and train the message decoder by minimizing the message loss \mathcal{L}_{msg} in Eq. (7) between the input and extracted messages. After pre-training the CLIP-guided message decoder, we then employ it to watermark the pre-trained 3DGS models. We freeze the message decoder and utilize it to extract the message from the rendered views, and the secret message can be embedded into 3DGS models by minimizing the following loss as:

$$\mathcal{L} = \lambda_{\text{recon}}(\mathcal{L}_{\text{rgb}} + \mathcal{L}_{\text{lpipl}}) + \lambda_{\text{msg}}\mathcal{L}_{\text{msg}} + \lambda_{\text{off}}\mathcal{L}_{\text{off}}, \quad (10)$$

where λ_{recon} , λ_{msg} , and λ_{off} are the hyper-parameters to balance the corresponding terms. Since \mathcal{L}_{rgb} and $\mathcal{L}_{\text{lpipl}}$ separately denote the RGB loss in Eq. (4) and LPIPS loss [15], optimizing the reconstruction term $\mathcal{L}_{\text{recon}} = \mathcal{L}_{\text{rgb}} + \mathcal{L}_{\text{lpipl}}$ can improve the visual similarity between the watermarked and original views. For extraction, given a 3DGS model watermarked by our *GuardSplat*, we can render the views from

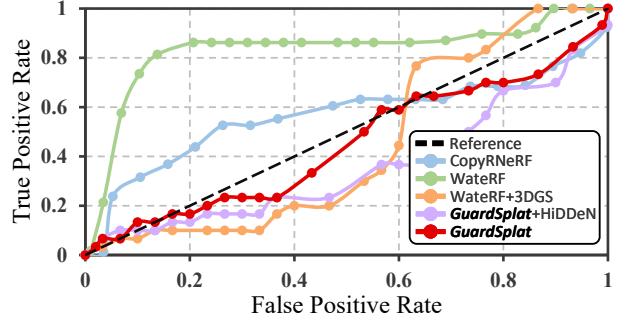


Figure 6. ROC curves produced by varying thresholds in StegExpose [5] on different methods. The closer the curve is to the “reference”, the more effective the method is regarding security.

arbitrary viewpoints using the official rendering pipeline, while the hidden messages can be directly extracted from the rendered views by Eq. (9) for copyright identification.

5. Experiments

Datasets. Following the experimental settings in [13, 24], we choose 2 commonly-used datasets: the Blender [27] and LLFF [26] datasets, for evaluation. Specifically, the Blender dataset consists of 8 synthetic bounded scenes, while the LLFF dataset consists of handheld forward-facing captures of 8 real scenes. For each scene, we employ 200 test views to evaluate the visual quality and bit accuracy. We report average values across all testing views in our experiments.

Baselines. We compare *GuardSplat* with 5 baselines to guarantee a fair comparison: 1) CopyRNeRF [24], 2) WaterRF [13], 3) HiDDeN [60] + 3DGS [17], 4) WaterRF + 3DGS, and 5) *GuardSplat* + HiDDeN. Specifically, CopyRNeRF and WaterRF are NeRF [27] watermarking methods. HiDDeN + 3DGS denotes training 3DGS models on the images watermarked by HiDDeN. WaterRF + 3DGS represents applying WaterRF to 3DGS models. *GuardSplat* + HiDDeN indicates replacing our message decoder with HiDDeN’s.

Implementation Details. *GuardSplat* is implemented on the top of a Pytorch [31] implementation of 3DGS [17]¹.

¹<https://github.com/graphdeco-inria/gaussian-splatting>

Table 2. Comparisons of the start-of-the-art methods on Blender [27] and LLFF [26] datasets for bit accuracy *w.r.t* various distortion types. We show the results on 16-bit messages. **Bold** and underlined texts indicate the best and second-best performance.

Methods	No distortion	Noise ($\mu=0.1$)	Rotation ($\pm\pi/6$)	Scaling ($\leq 25\%$)	Blur ($\sigma=0.1$)	Crop (40%)	Brightness (0.5~1.5)	JPEG (10% quality)	Combined
CopyRNeRF [24]	91.16	90.04	88.13	89.33	90.06	—	—	—	—
WateRF [13] + TensoRF [6]	95.67	95.36	<u>93.13</u>	93.29	<u>95.25</u>	95.40	90.91	86.99	84.12
WateRF [13] + 3DGS [17]	92.07	87.35	88.28	94.33	92.92	95.07	88.71	88.49	86.37
GuardSplat (Ours) + HiDDeN [60]	98.75	<u>96.63</u>	90.02	<u>95.93</u>	94.87	<u>97.25</u>	<u>94.97</u>	<u>90.04</u>	88.70
GuardSplat (Ours)	99.64	99.60	94.56	98.75	99.27	98.71	97.46	94.70	93.38

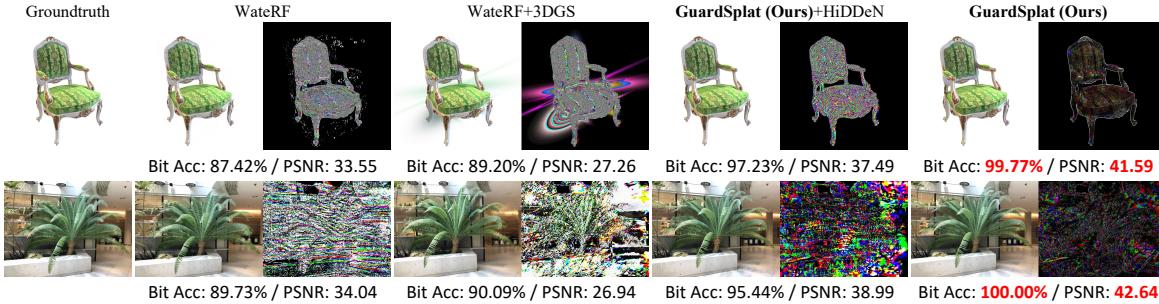


Figure 7. **Visual comparisons** between WateRF [13], WateRF + 3DGS, **GuardSplat** + HiDDeN [60], and our **GuardSplat** on $N_L = 32$ bits. Heatmaps show the differences ($\times 10$) between the watermarked and original views. Red text indicates the best performance.

Our experiments run on a single RTX 3090 GPU with 24G memory. We first train a 3DGS model on a given scene with multi-view photos, and then embed the message into that pre-trained 3DGS model. The size of learnable SH offsets is equal to that of the SH coefficients in the pre-trained models. Our decoder is a multi-layer perceptron (MLP) with 3 fully-connected (FC) layers, where the channels of the first 2 FC layers are separately set to 512 and 256, while the last FC channel is equal to N_L . *For training the message decoder*, we employ Adam [18] as the optimizer with a weight decay of 10^{-6} and a batch size of 64. The maximum epoch is set to 100, and the learning rate is set to 5×10^{-3} as default. It only takes 5 minutes for optimization. *For watermarking 3DGS models*, we employ Adam [18] as the optimizer with a weight decay of 10^{-6} and a batch size of 16. The maximum epoch is set to 100, and the learning rate of the learnable SH offsets is set to 5×10^{-3} as default. The hyper-parameters in Eq. (10) are set as $\lambda_{\text{recon}} = 1$, $\lambda_{\text{msg}} = 0.03$, and $\lambda_{\text{off}} = 10$, respectively. Each 3DGS model takes 20 minutes for watermarking.

Evaluation Methods. We follow the standard of digital watermarking to evaluate *GuardSplat* and the baselines in five aspects: **1) Capacity:** We evaluate the bit accuracy across various message lengths $N_L \in \{16, 32, 48\}$ on the 2D rendered views. **2) Invisibility:** We evaluate the visual similarity of views rendered from the watermarked and original models using Peak Signal-to-Noise Ratio (PSNR), Structured Similarity Index (SSIM) [48], and Learned Perceptual Image Patch Similarity (LPIPS) [58]. **3) Robustness:**

We investigate the extraction accuracy with $N_L = 16$ bits on various visual distortions, including Gaussian Noise, Random Rotation, Random Scaling, Gaussian Blur, Center Crop, Brightness Jitter, 10% JPEG Compression, and a combination of Crop, Brightness, and JPEG. **4) Security:** We perform StegExpose [5], an LSB steganography detection, on the rendered views. The detection set is built by mixing the views rendered from the watermarked and original models with equal proportions. **5) Efficiency:** We analyze the relationship between bit accuracy and training time.

5.1. Experimental Results

Capacity & Invisibility. We report the capacity and invisibility across various message lengths $N_L \in \{16, 32, 48\}$ on the Blender [27] and LLFF [26] datasets in terms of bit accuracy, PSNR, SSIM, and LPIPS in Table 1. As demonstrated, our *GuardSplat* surpasses all the competitors with a consistently superior performance *w.r.t* various message lengths, achieving a significant improvement in extraction accuracy and reconstruction quality. Moreover, *GuardSplat* achieves much higher bit accuracy than *GuardSplat* + HiDDeN on $N_L \geq 32$ bits, indicating that our CLIP-guided message decoder is superior to HiDDeN’s in handling large-capacity messages. Besides, *GuardSplat* + HiDDeN outperforms WateRF + 3DGS as they share the same message decoder, which demonstrates the superiority of our SH-aware watermark embedding module.

Robustness. We report bit accuracy across various visual distortions on $N_L = 16$ bits in Table 2. Visually, our *Guard-*

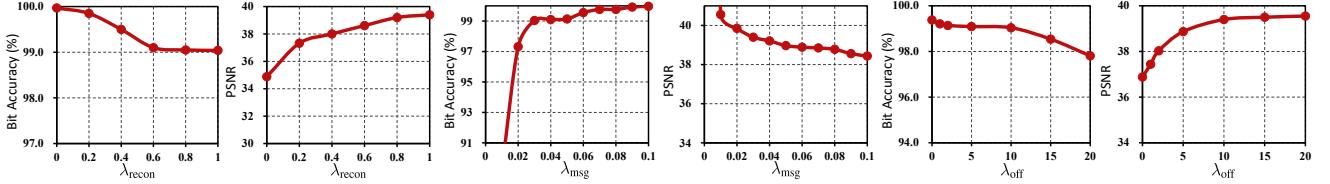


Figure 8. Performance over different hyper-parameter values with $N_L = 32$ bits on the Blender [27] and LLFF [26] datasets.

Splat outperforms all the baselines across various types of distortions, demonstrating that our Anti-distortion Message Extraction module enables the watermarked 3DGS models to learn robust SH features during optimization.

Security. We claim that our *GuardSplat* is secure. Since it adaptively embeds messages by slightly perturbing the SH features of every 3D Gaussians, it is difficult to remove the watermark from the model file. To further verify the security, we employ StegExpose [5] to detect steganographic content in rendered views. Figure 6 depicts the detection results on our *GuardSplat* and the baselines. As shown, our *GuardSplat* achieves superior security to the competitors.

Efficiency. As shown in Figure 3, the efficiency of existing advances is unsatisfactory. Specifically, training a Copy-RNeRF requires 84 hours, while WateRF, and WateRF + 3DGS separately spend 14 and 13 hours for optimization as it takes 12 hours to train the HiDDeN. Compared to these methods, our *GuardSplat* achieves much higher efficiency, which only takes 5 and 20 minutes to train the message decoder and watermark a 3DGS asset, respectively.

Visual Comparison. We visually compare our *GuardSplat* with baselines on $N_L = 32$ bits in Figure 7. As shown, our results present superior reconstruction quality and bit accuracy to the competitors. Moreover, the fidelity of WateRF + 3DGS is much lower than *GuardSplat* + HiDDeN, which demonstrates that altering all the attributes during optimization may lead to a significant decline in image quality.

5.2. Ablation Study & Sensitivity Analysis

We first conduct ablation experiments to prove the effectiveness of our model designs, including various watermark embedding strategies and different loss combinations. Subsequently, we evaluate the bit accuracy and fidelity of our method across different values of λ_{recon} , λ_{msg} , and λ_{off} to analyze the sensitivity. The results are evaluated on the Blender [27] and LLFF [26] datasets with $N_L = 32$ bits.

Various watermark embedding Strategies. We compare our SH-aware watermark embedding module with three strategies in Table 3, including updating all the attributes: “Offest_{all}”, the DC components of SH features: “Offset_{dc}”, and the residuals of SH features: “Offset_{rest}”. As shown, Offset_{all} achieves unsatisfactory reconstruction quality (row 1) as it alters the original 3D structure to embed messages during optimization. Moreover, Offset_{dc} (row 2) and

Table 3. Comparisons across various watermark embedding strategies w.r.t $N_L = 32$ bits on Blender [27] and LLFF [26] datasets. **Bold** text indicates the best and second-best performance.

	Bit Acc	PSNR	SSIM	LPIPS
Offest _{all}	98.79	36.56	0.9804	0.0123
Offset _{dc}	74.59	36.98	0.9828	0.0146
Offset _{rest}	98.25	38.70	0.9892	0.0077
SH-aware (Ours)	99.04	39.40	0.9939	0.0022

Table 4. Comparisons across different loss combinations w.r.t $N_L = 32$ bits on Blender [27] and LLFF [26] datasets. The first row denotes the original 3DGS model, and $\mathcal{L}_{\text{recon}} = \mathcal{L}_{\text{rgb}} + \mathcal{L}_{\text{lips}}$ indicates the reconstruction loss.

\mathcal{L}_{msg}	$\mathcal{L}_{\text{recon}}$	\mathcal{L}_{off}	Bit Acc	PSNR	SSIM	LPIPS
			53.41	inf	1.0000	0.0000
✓			100.00	31.79	0.9604	0.0379
✓	✓		99.26	36.88	0.9831	0.0101
✓	✓	✓	99.04	39.40	0.9939	0.0022

Offset_{rest} (row 3) are inferior to our SH-aware module (row 4), proving the correctness of our motivation.

Different Loss Combinations. We explore the optimal loss combinations in Table 4. Compared to the original 3DGS model (row 1), only using message loss \mathcal{L}_{msg} (row 2) will significantly reduce the reconstruction quality. Though simultaneously minimizing the message loss \mathcal{L}_{msg} and reconstruction loss $\mathcal{L}_{\text{recon}}$ (row 3) can alleviate the decline in fidelity, it can only achieve sub-optimal results due to some large SH offsets. Thus, we design the offset loss \mathcal{L}_{off} to restrict the deviation of SH offsets, which achieves the optimal reconstruction quality (row 4).

Various Hyper-parameter Values. We analyze the sensitivity of 3 hyper-parameters: λ_{recon} , λ_{msg} , and λ_{off} in Figure 8. For simplification, we only change the value of one hyper-parameter, while keeping the other two at their default values. Visually, as increasing λ_{recon} from 0 to 1, the PSNR rises with a subtle decline in bit accuracy. When $\lambda_{\text{msg}} \in [0, 0.03]$, there is a significant change in performance. However, this effect gradually diminishes as $\lambda_{\text{msg}} \in [0.03, 0.1]$. $\lambda_{\text{off}} = 10$ is a watershed in performance, as values above and below it will influence the trade-off.

Thus, we choose $\lambda_{\text{recon}} = 1$, $\lambda_{\text{msg}} = 0.03$, and $\lambda_{\text{off}} = 10$ for the optimal overall performance.

6. Conclusion

In this paper, we propose *GuardSplat*, a novel framework to effectively protect the copyright of 3DGS assets. Specifically, we build an efficient message decoder via CLIP-guided Message Decoupling Optimization, enabling the watermarked 3DGS models to achieve high capacity and efficiency. Moreover, we propose a SH-aware message embedding module tailored for 3DGS, which can seamlessly embed the messages while maintaining fidelity, meeting the demands for invisibility and security. We further propose an anti-distortion message extraction module to achieve strong robustness against various visual distortions. Experimental results demonstrate that our *GuardSplat* outperforms the baselines and achieves fast training speed.

Acknowledgement

This project is supported by the Project of Guangdong Provincial Key Laboratory of Information Security Technology (Grant No. 2023B1212060026), the National Natural Science Foundation of China (12326618), the National Natural Science Foundation of China (U22A2095), and the Key-Area Research and Development Program of Guangzhou (202206030003).

References

- [1] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgb-d surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6290–6301, 2022. [3](#)
- [2] Mauro Barni, Franco Bartolini, and Alessandro Piva. Improved wavelet-based watermarking through pixel-wise masking. *IEEE TIP*, 10(5):783–791, 2001. [3](#)
- [3] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *ICCV*, 2021. [3](#)
- [4] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5470–5479, 2022. [3](#)
- [5] Benedikt Boehm. Stegexpose-a tool for detecting lsb steganography. *arXiv preprint arXiv:1410.6656*, 2014. [6](#), [7](#), [8](#)
- [6] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *ECCV*, pages 333–350. Springer, 2022. [7](#)
- [7] Zixuan Chen, Jianhuang Lai, Lingxiao Yang, and Xiaohua Xie. Cunerf: Cube-based neural radiance field for zero-shot medical image arbitrary-scale super resolution. In *ICCV*, 2023. [3](#)
- [8] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12882–12891, 2022. [3](#)
- [9] Pierre Fernandez, Guillaume Couairon, Hervé Jégou, Matthijs Douze, and Teddy Furon. The stable signature: Rooting watermarks in latent diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22466–22477, 2023. [3](#)
- [10] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In *CVPR*, pages 5354–5363, 2024. [1](#)
- [11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 33:6840–6851, 2020. [3](#)
- [12] Xiufeng Huang, Ruiqi Li, Yiu-ming Cheung, Ka Chun Cheung, Simon See, and Renjie Wan. Gaussianmarker: Uncertainty-aware copyright protection of 3d gaussian splatting. *NeurIPS*, 2024. [2](#), [3](#)
- [13] Youngdong Jang, Dong In Lee, MinHyuk Jang, Jong Wook Kim, Feng Yang, and Sangpil Kim. Waterf: Robust watermarks in radiance fields for protection of copyrights. In *CVPR*, pages 12087–12097, 2024. [2](#), [3](#), [5](#), [6](#), [7](#), [1](#)
- [14] Youngdong Jang, Hyunje Park, Feng Yang, Heejoo Ko, Eui-jin Choo, and Sangpil Kim. 3d-gsw: 3d gaussian splatting watermark for protecting copyrights in radiance fields. *arXiv preprint arXiv:2409.13222*, 2024. [2](#), [3](#)
- [15] Younghyun Jo, Sejong Yang, and Seon Joo Kim. Investigating loss functions for extreme super-resolution. In *CVPRW*, pages 424–425, 2020. [6](#)
- [16] James T Kajiya and Brian P Von Herzen. Ray tracing volume densities. *ACM SIGGRAPH computer graphics*, 18(3):165–174, 1984. [3](#)
- [17] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM TOG*, 42(4):139–1, 2023. [1](#), [3](#), [4](#), [5](#), [6](#), [7](#)
- [18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. [7](#)
- [19] Georgios Kopanas, Julien Philip, Thomas Leimkühler, and George Drettakis. Point-based neural rendering with per-view optimization. *Comput. Graph. Forum*, 40(4), 2021. [3](#)
- [20] Martin Kutter, Frederic D Jordan, and Frank Bossen. Digital signature of color images using amplitude modulation. In *Storage and Retrieval for Image and Video Databases V*, pages 518–526. SPIE, 1997. [3](#)
- [21] Yixin Liang, Xin Yang, Jiantao Lin, Haodong Li, Xiaogang Xu, and Yingcong Chen. Luciddreamer: Towards high-fidelity text-to-3d generation via interval score matching. In *CVPR*, 2024. [1](#), [3](#)
- [22] Xiyang Luo, Ruohan Zhan, Huiwen Chang, Feng Yang, and Peyman Milanfar. Distortion agnostic deep watermarking. In *CVPR*, pages 13548–13557, 2020. [3](#)

- [23] Xiyang Luo, Yinxiao Li, Huiwen Chang, Ce Liu, Peyman Milanfar, and Feng Yang. Dvmark: a deep multiscale framework for video watermarking. *IEEE TIP*, 2023. 3
- [24] Ziyuan Luo, Qing Guo, Ka Chun Cheung, Simon See, and Renjie Wan. Copyrnerf: Protecting the copyright of neural radiance fields. In *ICCV*, pages 22401–22411, 2023. 2, 3, 5, 6, 7, 1
- [25] Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. Gaussian splatting slam. In *CVPR*, pages 18039–18048, 2024. 3
- [26] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM TOG*, 38(4):1–14, 2019. 2, 6, 7, 8, 1
- [27] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 2, 3, 5, 6, 7, 8, 1
- [28] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM TOG*, 41(4):1–15, 2022. 3
- [29] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *CVPR*, pages 11453–11464, 2021. 3
- [30] 1 Ryutarou Ohbuchi, 1 Akio Mukaiyama, and 2 Shigeo Takahashi. A frequency-domain approach to watermarking 3d shapes. In *Comput. Graph. Forum*, pages 373–382. Wiley Online Library, 2002. 3
- [31] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NeurIPS*, 2017. 6, 1
- [32] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *ICLR*, 2022. 3
- [33] Thomas Porter and Tom Duff. Compositing digital images. In *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, pages 253–259, 1984. 3
- [34] Emil Praun, Hugues Hoppe, and Adam Finkelstein. Robust mesh watermarking. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 49–56, 1999. 3
- [35] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763. PMLR, 2021. 2, 4, 5
- [36] MS Raval and PP Rege. Discrete wavelet transform based multiple watermarking scheme. In *TENCON 2003. Conference on Convergent Technologies for Asia-Pacific Region*, pages 935–938. IEEE, 2003. 3
- [37] Christoph Reich, Biplob Debnath, Deep Patel, and Srimat Chakradhar. Differentiable jpeg: The devil is in the details. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4126–4135, 2024. 1
- [38] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pages 10684–10695, 2022. 3
- [39] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS*, 35:36479–36494, 2022. 3
- [40] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *NeurIPS*, pages 20154–20166. Curran Associates, Inc., 2020. 3
- [41] Zhijing Shao, Zhaolong Wang, Zhuang Li, Duotun Wang, Xiangru Lin, Yu Zhang, Mingming Fan, and Zeyu Wang. Splattingavatar: Realistic real-time human avatars with mesh-embedded gaussian splatting. In *CVPR*, pages 1606–1616, 2024. 3
- [42] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2020. 3
- [43] Qi Song, Ziyuan Luo, Ka Chun Cheung, Simon See, and Renjie Wan. Geometry cloak: Preventing tgs-based 3d reconstruction from copyrighted images. *NeurIPS*, 2024. 3
- [44] Stanislaw Szymanowicz, Chrisitian Rupprecht, and Andrea Vedaldi. Splatter image: Ultra-fast single-view 3d reconstruction. In *CVPR*, pages 10208–10217, 2024. 1, 3
- [45] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. In *ICLR*, 2023. 3
- [46] Peining Tao and Ahmet M Eskicioglu. A robust multiple watermarking scheme in the discrete wavelet transform domain. In *Internet Multimedia Management Systems V*, pages 133–144. SPIE, 2004. 3
- [47] Guangcong Wang, Zhaoxi Chen, Chen Change Loy, and Ziwei Liu. Sparsenerf: Distilling depth ranking for few-shot novel view synthesis. In *ICCV*, 2023. 3
- [48] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE TIP*, 13(4):600–612, 2004. 7
- [49] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. In *NeurIPS*, 2023. 3
- [50] Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-rings watermarks: Invisible fingerprints for diffusion images. *Advances in Neural Information Processing Systems*, 36, 2024. 3
- [51] Chi Yan, Delin Qu, Dan Xu, Bin Zhao, Zhigang Wang, Dong Wang, and Xuelong Li. Gs-slam: Dense visual slam with 3d gaussian splatting. In *CVPR*, pages 19595–19604, 2024. 1, 3
- [52] Taoran Yi, Jiemin Fang, Junjie Wang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. Gaussiandreamer: Fast generation from text to 3d gaussians by bridging 2d and 3d diffusion models. In *CVPR*, 2024. 3

- [53] Innfarm Yoo, Huiwen Chang, Xiyang Luo, Ondrej Stava, Ce Liu, Peyman Milanfar, and Feng Yang. Deep 3d-to-2d watermarking: Embedding messages in 3d meshes and extracting them from 2d renderings. In *CVPR*, pages 10031–10040, 2022. 3, 5
- [54] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4578–4587, 2021. 3
- [55] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. In *CVPR*, pages 19447–19456, 2024. 1, 3
- [56] Ye Yuan, Xuetong Li, Yangyi Huang, Shalini De Mello, Koki Nagano, Jan Kautz, and Umar Iqbal. Gavatar: Animatable 3d gaussian avatars with implicit mesh learning. In *CVPR*, pages 896–905, 2024. 1, 3
- [57] Chaoning Zhang, Philipp Benz, Adil Karjauv, Geng Sun, and In So Kweon. Udh: Universal deep hiding for steganography, watermarking, and light field messaging. *NeurIPS*, 33: 10223–10234, 2020. 3
- [58] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pages 586–595, 2018. 7
- [59] Xuanyu Zhang, Jiarui Meng, Runyi Li, Zhipei Xu, Yongbing Zhang, and Jian Zhang. Gs-hider: Hiding messages into 3d gaussian splatting. In *NeurIPS*, 2024. 2, 3
- [60] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks. In *ECCV*, pages 682–697, 2018. 2, 3, 6, 7, 1
- [61] Zi-Xin Zou, Zhipeng Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Yan-Pei Cao, and Song-Hai Zhang. Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers. In *CVPR*, pages 10324–10335, 2024. 3

GuardSplat: Efficient and Robust Watermarking for 3D Gaussian Splatting

Supplementary Material

A. Overview

In this supplementary material, we further provide more discussions, implementation details, and results as follows:

- Section **B** presents the superior performance of our *GuardSplat* to the state-of-the-art methods.
- Section **C** supplements more implementation details, including the architecture of the message decoder and distortion layers.
- Section **D** conducts an additional evaluation for security, exploring whether the watermarks can be simply removed from model files.
- Section **E** visualizes more experimental results, including visual comparisons of various ablations in Tables **3** and **4** of the main paper, and quantitative results on larger-capacity messages $N_L = \{64, 72\}$.

B. Superiority of our GuardSplat

To prove the superiority of our *GuardSplat*, we show the performance between our *GuardSplat* and state-of-the-art methods in Figure **S1**, where our *GuardSplat* + HiDDeN [60] denotes replacing our CLIP-guided message decoder with HiDDeN’s to optimize our *GuardSplat*. As shown, our *GuardSplat* and *GuardSplat* + HiDDeN achieve superior performance in bit accuracy and reconstruction quality to the compared methods, demonstrating that the proposed watermarking watermark can seamlessly embed the messages into 3DGS assets while maintaining the fidelity. Moreover, unlike the compared methods that require hours and even days for optimization, the training time of our *GuardSplat* is 25 minutes, which is more suitable for real-world applications.

C. More Implementation Details

We supplement more details of our *GuardSplat*.

Distortion layers. Since the proposed CLIP-guided message decoder can deal with Gaussian Noise and Gaussian Blur, it only considers the visual distortions of cropping, scaling, rotation, brightness jittering, and JPEG compression. The former four visual distortions are differentiably re-implemented by Pytorch [31] built-in functions, while the differentiable JPEG comparison is built by [37]. During training, we employ the differentiable distortion layer to simulate various visual distortions for optimization. In the test, the rendered views are distorted using the visual distortions built by OpenCV for evaluation.

Architecture of Message Decoder. Figure **S2** depicts the architecture of our CLIP-guided Message Decoder. Thanks

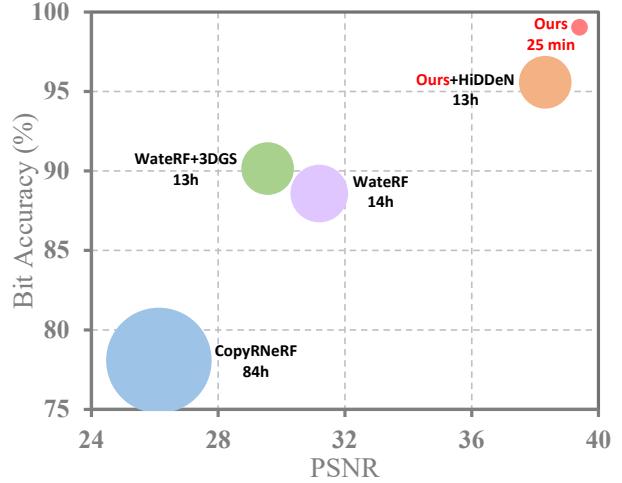


Figure S1. Bit accuracy and PSNR of CopyRNeRF [24], WaterRF [13], WaterRF [13] + 3DGs [17], our **GuardSplat** + HiDDeN [60] and our **GuardSplat** with $N_L = 32$ bits on Blender [27] and LLFF [26] datasets. The radius of circles is proportional to their total training time evaluated on RTX 3090 GPU.

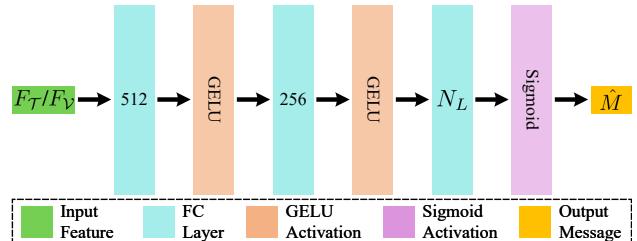


Figure S2. The architecture of our message decoder. Given the output features F_T or F_V , we first pass it through 2 FC layers with GELU activations, where the first and second FC layers have 512 and 256 channels, respectively. Then, we map the output feature to the binary message by using a N_L -channel FC layer and a Sigmoid activation.

to CLIP’s rich representation, our decoder can achieve excellent performance with minimal parameter size.

D. Additional Evaluation for Security

We conduct an additional experiments to evaluate the security of our *GuardSplat* in Table **S1**, investigating whether the malicious users can remove the watermarks from the model file by pruning the $K\%$ of Gaussians, where $K \in \{5, 10, 15, 20, 25\}$. “Bottom K ” denotes pruning K of low-opacity Gaussians, while “random” denotes randomly pruning K of the Gaussians. As demonstrated, our *GuardSplat*

Table S1. Security analysis across various pruning ratios $K\%$. “Bottom K ” denotes removing $K\%$ of the low-opacity Gaussians, while “random” denotes randomly removing $K\%$ of the Gaussians.

K (%)	Bottom K				Random			
	Bit Acc	PSNR	SSIM	LPIPS	Bit Acc	PSNR	SSIM	LPIPS
5	99.04	39.38	0.9939	0.0022	98.59	37.76	0.9916	0.0033
10	99.02	39.06	0.9937	0.0025	96.87	36.35	0.9891	0.0047
15	98.99	38.68	0.9933	0.0031	94.68	35.14	0.9832	0.0063
20	98.94	38.33	0.9928	0.0037	91.98	33.98	0.9779	0.0081
25	98.74	37.87	0.9922	0.0041	88.59	31.50	0.9721	0.0103

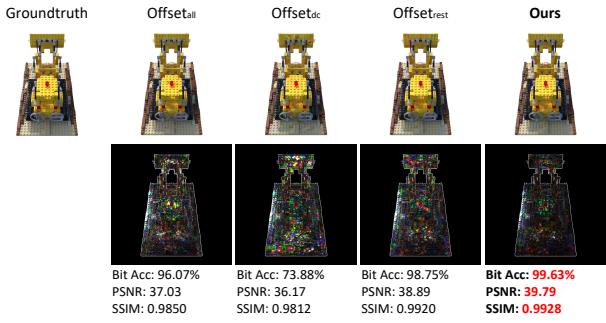


Figure S3. Visual comparisons between various watermarking embedding strategies and our SH-aware module. Heatmaps at the bottom show the differences ($\times 10$) between the watermarked and Groundtruth. **Red** text indicates the best performance.

still achieves a bit accuracy of 98.74% when 25% of the low-opacity Gaussians are removed, indicating that simply removing low-opacity Gaussians does not effectively attack our method. Though randomly removing the Gaussians can lead to a significant decline in bit accuracy, it also greatly affects the reconstruction quality (*i.e.*, PSNR, SSIM, and LPIPS), resulting in low-fidelity rendering. This experimental result demonstrates that the malicious cannot directly remove the watermarks from the model file, verifying the security of our *GuardSplat*.

E. More Results

E.1. Additional Visual Comparisons

Various Watermark Embedding Strategies. In the main paper, we explore the performance under various watermark embedding strategies with $N_L = 32$ bits (see quantitative results in Table 3). For better comparisons, we further visualize the results of various watermark embedding strategies in Figure S3. As shown, the proposed SH-aware module achieves superior bit accuracy and reconstruction quality to the competitors.

Various Loss Combinations. In the main paper, we quantitatively compare the performance across various loss com-

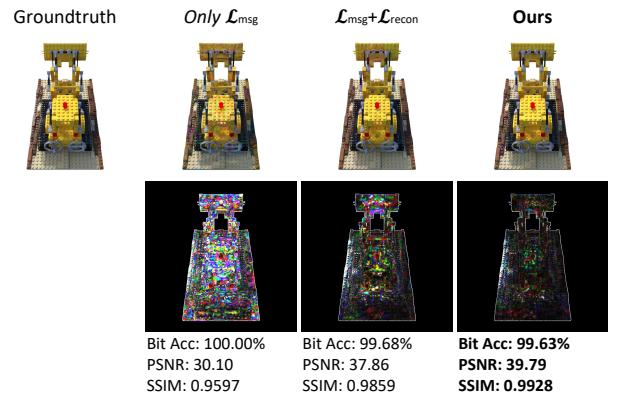


Figure S4. Visual comparisons of various loss combinations. “Ours” denotes the combination of $\mathcal{L}_{\text{msg}} + \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{off}}$. Heatmaps at the bottom show the differences ($\times 10$) between the watermarked and Groundtruth. **Bold** text indicates the best overall performance.

Table S2. Quantitative results of our **GuardSplat** on Blender [27] and LLFF [26] datasets w.r.t $N_L \in \{64, 72\}$ bits.

N_L	Bit Acc	PSNR	SSIM	LPIPS
64	97.41	37.76	0.9899	0.0040
72	96.64	36.47	0.9866	0.0053

bination in Table 4. We also conduct a visual comparison of these ablation variants in Figure S4. As shown, “ $\mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{msg}} + \mathcal{L}_{\text{off}}$ ” achieves the best performance in bit accuracy and reconstruction quality.

E.2. Quantitative Results on Larger-Capacity Messages.

To further investigate the superiority of our *GuardSplat* in capacity, we supplement the results on larger message lengths ($N_L \in \{64, 72\}$) in Table S2. As demonstrated, the bit accuracy and reconstruction quality of our 72-bit results are still higher than the state-of-the-art methods on $N_L \in \{16, 32, 48\}$ bits reported in the main paper (see Ta-

ble 1), significantly improve the capacity of existing baselines.