GaussianSeal: Rooting Adaptive Watermarks for 3D Gaussian Generation Model

Runyi Li Xuanyu Zhang Chuhan Tong Zhipei Xu Jian Zhang†
School of Electronic and Computer Engineering, Peking University, Shenzhen, China

Abstract

With the advancement of AIGC technologies, the modalities generated by models have expanded from images and videos to 3D objects, leading to an increasing number of works focused on 3D Gaussian Splatting (3DGS) generative models. Existing research on copyright protection for generative models has primarily concentrated on watermarking in image and text modalities, with little exploration into the copyright protection of 3D object generative models. In this paper, we propose the first bit watermarking framework for 3DGS generative models, named GaussianSeal, to enable the decoding of bits as copyright identifiers from the rendered outputs of generated 3DGS. By incorporating adaptive bit modulation modules into the generative model and embedding them into the network blocks in an adaptive way, we achieve high-precision bit decoding with minimal training overhead while maintaining the fidelity of the model's outputs. Experiments demonstrate that our method outperforms post-processing watermarking approaches for 3DGS objects, achieving superior performance of watermark decoding accuracy and preserving the quality of the generated results.

1. Introduction

The advancement of AI Generated Content (AIGC) has marked a significant shift in how we perceive and interact with digital media. Over the years, generative models have evolved dramatically, enhancing their capabilities in producing high-quality images and videos [13, 42, 45, 47]. Recently, AIGC technologies have begun to extend beyond 2D images, venturing into the realm of 3D model generation. This transition not only broadens the applications of AIGC in fields such as gaming, virtual reality, and architecture but also poses new challenges and opportunities for researchers and practitioners in computer vision. As we delve into this emerging domain, recognizing the importance of safety considerations in AIGC models becomes essential for guiding future advancements in both generative modeling and the broader realm of computer vision [5, 34, 38, 43, 69] and AI generation [59, 71].

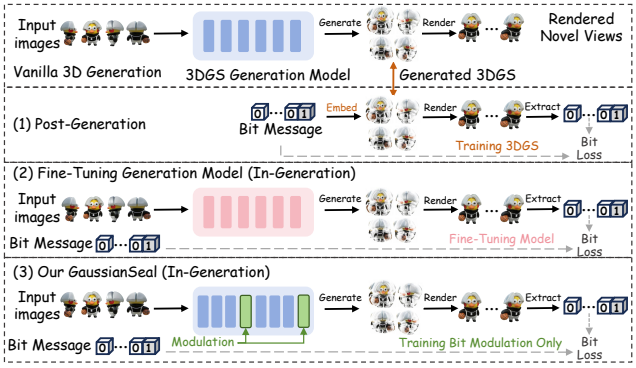

Figure 1. Current approaches for 3D Gaussian watermarking, including (1) post-generation methods, (2) fine-tuning generation model, and (3) our bit modulation into generation model. Our framework is lightweight, making trade-off between accurate bit extraction and keeping generation quality.

3D Gaussian Splatting (3DGS) [20] has emerged as a new generation of representation methods for 3D objects and scenes, providing efficient and high-quality 3D models [8, 26]. Certain approaches utilize images as guiding inputs to generate high-quality 3D Gaussian models in a controllable manner [30, 53, 58, 61]. This versatility makes 3DGS particularly valuable in fields such as game development and interactive applications. As this technology evolves, it holds the potential to enhance the visual fidelity and interactivity of digital environments significantly.

Current model watermarking methods [9, 10, 37, 44], especially those designed for generative models, predominantly focus on text-to-image models, such as Stable Diffusion (SD) [45]. By fine-tuning or adding learnable adapter modules into the generation model, the model itself is able to be watermarked. While these techniques have made strides in embedding watermarks within image outputs, their application to 3DGS remains limited. There are also explorations addressing the protection of individual 3DGS objects [16, 19, 50, 70]. With a fixed watermark decoder and the specially designed 3DGS structure, the 3DGS is trained with a watermark extracted via the decoder, but *one at a time*. Illustrations of these methods and our proposed framework

†Corresponding author.

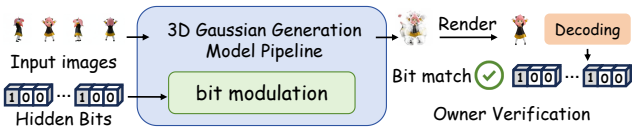

Figure 2. Application and watermarking process of our proposed framework GaussianSeal.

are shown in Fig. 1.

Our motivation is two-folds: ① *Post-generation* watermarking methods (Fig. 1(1)) only watermark one 3D object at a time, which is inefficient and **requires a significant amount of time** for each object; ② For *in-generation* watermarking methods, the approach of fine-tuning generative model (Fig. 1(2)) would **impact the generation quality**, which struggles to achieve precise watermark extraction while maintaining generation quality, and they **require large-scale datasets and substantial GPU memory**. In short, achieving both high-precision watermark decoding and high-quality 3D generation presents is challenging.

Motivated by the insights and limitations mentioned above, we propose the first bit watermarking framework for 3DGS generative models, taking the current state-of-the-art 3D generation model LGM [53] as an example. Specifically, we introduce an adaptive bit modulation mechanism that embeds secret messages directly into the model network (Fig. 1(3)). To ensure that the generative quality of the model remains unaffected, we only fine-tune the modulation module without altering the original model parameters. The added watermark can be extracted from the 3DGS renderings of LGM output, providing a robust method for owner verification and copyright protection in 3D content creation. The application scenarios and the overall watermarking process of our proposed method are shown in Fig. 2. Our contributions are listed as follows:

❑ We introduce the first bit watermarking framework for 3D generative models, enabling precise verification of bit information and effectively safeguarding the copyrights of 3DGS generating models.

❑ We present an adaptive and effective bit watermark modulation mechanism that achieves a balance between message decoding accuracy and the perceptual quality of the generative model, which exhibits strong robustness and security.

❑ Extensive experiments demonstrate that our watermarking framework delivers precise and robust performance in copyright protection for 3DGS generative models with time efficiency, achieving state-of-the-art results compared to existing post-generation methods and possible intuitive in-generation methods.

2. Related Work

2.1. Watermarking Generation Models

With the continuous emergence of AIGC models, the traceability and copyright protection of AI-generated models have gradually attracted research interest. Most current watermarking methods for AI-generated models focus on Text-to-Image (T2I) models [40, 67], especially watermarking for Stable Diffusion [21, 24, 29, 31, 55, 57, 66, 72]. Stable Signature [10] proposed the first watermarking method for Stable Diffusion, which involves pre-training a watermark extractor applicable to both VAE [22] encoders and decoders, then fixing this extractor and fine-tuning the Stable Diffusion decoder to generate images that can reveal bit watermarks through the extractor. This approach has become a common practice for subsequent generative model watermarking. AquaLoRA [9] referenced Stable Signature's method and shifted the fine-tuning focus to diffusion UNet, improving the precision of watermark decoding. However, directly fine-tuning model weights can lead to a significant decrease in generation quality. LaWa [44] considers using a bit modulation module to modulate bit information between diffusion UNet blocks, training the modulation network so that the extractor can decode bit information. Since the original weights were not modified, the generation quality of the model was maintained. WaDiff [37] further explores generation quality, only modulating the watermark in the last layer of the UNet, to minimize the impact on generation quality.

There are also some training-free approaches including Tree-Ring [56], RingID [6], CRoSS [63] and Gaussian-Shading [60] via DDIM Inversion [39]. However, they are only available with diffusion-based models that root watermark in the frequency domain or initial sampling noise, which is kept during the DDIM Inversion process, thus not applicable in 3D Gaussian generation without a diffusion model [12, 28, 49, 52].

2.2. Watermarking for 3D Content

The idea of 3D content watermarking [64] is consistent with the practice of generation model watermarking, which involves leveraging a watermark extractor network first and then fine-tuning the 3D content to decode the bit key correctly [15, 41, 48, 48, 51, 62]. We mainly introduce the watermarking approaches for NeRF [36] and 3DGS [20].

NeRF Watermarking. StegaNeRF [25] is the first exploration into embedding customizable, imperceptible, and recoverable information within NeRF renderings for ownership identification. The method allows for accurate hidden information extractions from images rendered by NeRF while preserving its visual quality through an optimization framework. CopyRNeRF [33] introduces a method to protect the copyright of NeRF models by watermarking, replacing the original color representation in NeRF with a watermarked one. The approach designs a distortion-resistant rendering scheme to ensure robust message extraction in 2D renderings, even when the rendered samples are severely distorted. WateRF [18] presents an innovative watermarking method

applicable to both implicit and explicit NeRF representations by embedding binary messages during the rendering process. The method utilizes the discrete wavelet transform in the NeRF space for watermarking and adopts a deferred back-propagation technique along with a patch-wise loss to improve rendering quality and bit accuracy.

3DGS Watermarking. GS-Hider [70] is a steganography framework designed for 3DGS, capable of embedding 3D scenes and images invisibly into original GS point clouds and accurately extracting hidden messages. The framework replaces the spherical harmonics coefficients of the original 3DGS with a coupled secure feature attribute, using a scene decoder and a message decoder to disentangle the original RGB scene from the hidden message. GaussianStego [27] is a novel method for embedding images into generated 3DGS assets. It employs an optimization framework that allows for the extraction of hidden information from rendered images while ensuring minimal impact on rendered content quality. 3D-GSW [19] introduces a novel watermarking method that embeds binary messages into 3DGS by fine-tuning 3DGS models aiming to protect copyrights. The method utilizes Discrete Fourier Transform (DFT) to split 3DGS into high-frequency areas, achieving high-capacity and imperceptible watermarking. GaussianMarker [16] introduces an uncertainty-aware digital watermarking method for protecting the copyright of 3DGS models. It proposes embedding invisible watermarks by adding perturbations to 3D Gaussian parameters with high uncertainty, ensuring both invisibility and robustness against various distortions.

3. Preliminaries for 3DGS Generation

3D Gaussian Splatting (3DGS) is a novel technique in the field of computer graphics and vision that provides an explicit scene representation and enables novel view synthesis without relying on neural networks, unlike NeRF [3, 4, 14, 20, 32, 65]. This method represents scenes using anisotropic 3D Gaussians, which are unstructured spatial distributions. The key to 3DGS is the use of a fast, differentiable GPU-based rendering method to optimize the number, position, and intrinsic properties of the Gaussians, thereby enhancing the quality of scene representation.

The mathematical representation of a 3D Gaussian is given by the formula:

$$G(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right) \quad (1)$$

where $\mathbf{x}$ is the point position, the mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ determine spatial distribution. $\boldsymbol{\Sigma}$ is further calculated via scale matrix $\mathbf{S}$ and rotation matrix $\mathbf{R}$:

$$\boldsymbol{\Sigma} = \mathbf{R}\mathbf{S}^\top\mathbf{S}\mathbf{R}^\top \quad (2)$$

Futher, 3D Gaussian can be projected into image space by :

$$\boldsymbol{\mu}^{'} = \mathbf{P}\mathbf{W}\boldsymbol{\mu}, \boldsymbol{\Sigma}^{'} = \mathbf{J}\mathbf{W}\boldsymbol{\Sigma}\mathbf{W}^\top\mathbf{J}^\top \quad (3)$$

where $\boldsymbol{\Sigma}^{'}$ is the covariance matrix in camera space, $\mathbf{W}$ is the viewing transformation matrix, and $\mathbf{J}$ is the Jacobian matrix used to approximate the projective transformation $\mathbf{P}$. For detailed rendering, color $\mathbf{c}$ of specified pixel $\mathbf{p}$ is:

$$\mathbf{c}[\mathbf{p}] = \sum_{i=1}^{N} c_i \sigma_i \prod_{j=1}^{i-1}(1-\sigma_j) \quad (4)$$

$$\sigma_i = \alpha_i \exp\left(-\frac{1}{2}(\mathbf{p}-\hat{\boldsymbol{\mu}})^\top \hat{\boldsymbol{\Sigma}}^{-1}(\mathbf{p}-\hat{\boldsymbol{\mu}})\right) \quad (5)$$

where N denotes the number of sample Gaussian points that overlap pixel $\mathbf{p}$. c_i and α_i denote the color and opacity of the i-th Gaussian, respectively.

Besides the aforementioned representation, it is also possible to present the 3DGS in tensors, which is commonly used in 3D Gaussian generation models [11, 53, 58, 61]. Given that one 3DGS can be represented using a 3D-Gaussian distribution, with its mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\Sigma}$, opacity α, scale $\mathbf{S}$, rotation $\mathbf{R}$, and RGB value $\mathbf{c}$[†], we can embed all these information into one tensor $\mathbf{g}$, as the output target of generation models. In LGM, the output tensor has 14 channels, 3 for point position, 1 for opacity, 3 for scale, 4 for rotation, and 3 for RGB value. The dimension of the tensor under each channel is splatting size $\times$ splatting size.

4. Method

4.1. Task Settings & Potential Intuitive Approaches

Task Settings. Our watermarking framework is designed to protect the copyright of 3DGS generative models and their outputs. The copyright owner of a 3DGS generative model specifies a bit string as the copyright identifier, which must be accurately decoded from the generated results. Since 3DGS structures are typically viewed after rendering as images, we choose to decode the bit copyright identifier from the rendered outputs. In this context, 3DGS models are publicly uploaded to the internet, necessitating a certain level of robustness against common attack types, such as point cloud pruning, and augmentation attacks to rendered images.

Intuitive Approaches. Considering that our work is the first to embed bit watermark in 3DGS generative models, we present some potential alternative approaches to illustrate the effectiveness of our method, both *post-generation* and *in-generation* watermarking methods, discussing why they are not suitable for watermarking 3DGS generative models. Performance comparisons are detailed in the Sec. 5.2.

❑ **3DGS+HiDDeN**: This approach involves training an additional watermark decoder to extract bit information from the rendered outputs of the 3DGS. We choose the HiDDeN [73] network, a commonly used architecture for bit addition and

[†]In standard 3DGS representation, the RGB value is calculated via Spherical Harmonics coefficients. LGM simplifies this by directly storing RGB values in the 3DGS tensor.

decoding, to facilitate bit extraction. However, while this method might be effective in image watermarking, it **struggles to decode correctly** on rendered results of 3DGS, as rendering is a **strong degradation** for 3DGS.

❑ **3DGS+WateRF**: It is also possible to embed watermark into 3DGS objects using the methods applied in WateRF [18], which is proposed to hide watermark into NeRF. However, due to the different natures of 3DGS and NeRF, approaches available for NeRF watermarking might lead to **inferior performance** for 3DGS watermarking.

❑ **Fine-tuning UNet**: Besides adding modulation modules to the fixed generation model, another possible idea is to directly fine-tune the weights of the generation model, making the model output the 3DGS while also decoding bits correctly. This approach works well in diffusion-based image generation models as its UNet is trained through a denoising process, making latent code robust to noise, thus minor weight changes do not significantly affect the generation quality. In contrast, the latent codes in the UNet of LGM are more fragile, thus weight modifications from fine-tuning can drastically alter the attributes of the generated 3DGS point clouds, thereby **hard to converge** and **impacting the generation quality**.

❑ **Extract from 3DGS directly**: As discussed in Sec. 3, the 3DGS contains dedicated channels representing RGB values, which can be reshaped to match the image size for bit decoding. However, the visualization results of the RGB values in the 3DGS tensor show a substantial **domain gap from natural images**, making it challenging for a pre-trained bit decoder to extract the embedded information accurately.

4.2. Overview

Our proposed 3DGS generative model watermarking framework is based on the current commonly used 3DGS generation state-of-the-art method LGM [53]. By inputting an image $\mathbf{I}$ and a user-specified binary string m as the watermark, our framework will output a high-quality 3DGS object visually consistent with the input image, and the watermark can be decoded in rendered results $\mathbf{R}$. For processing input data, we first use a pre-trained MVDream [46] pipeline to generate 4 multi-view images $\mathbf{x}$ of input image $\mathbf{I}$ to provide detailed multi-view visual information, which is further transformed as features via Conv2D layer $\mathcal{C}_{in}$. The input watermark m is transformed into binary tensor $\mathbf{m}$. The images feature $\mathbf{z}$ are then sent into generation UNet $\mathcal{U}$. The UNet has 5 encoder blocks $\mathcal{U}_e$, and 3 decoder blocks (with 1 middle block) $\mathcal{U}_d$.

To effectively embed bits into generation UNet, we implement an adaptive bit modulation module, denoted as $\mathcal{B}$, allowing for seamless integration of the watermark within the generative process. The modulation modules are located separately at the image input b_{in}, the output of the encoder blocks b_{mid}, and the output of UNet b_{out}. We observe that the value range of 3DGS is consistent with the binary 0-1 values of bits, so directly combining the bit tensor with the block output would affect the generation quality. Therefore, we designed learnable adaptive coefficients α, β and γ for each bit embeddings to avoid this issue. A detailed modulation process is shown in Algo. 1.

The tensor output by the LGM, denoted as $\mathbf{z}_{out}$, is transformed into the shape of the 3DGS tensor (mentioned in Sec. 3) via a Conv2D layer $\mathcal{C}_{out}$, denoted as $\mathbf{g}$. It is further rendered into images of 180 different views $\mathbf{r}$. From these rendered images, we utilize the output low-pass subband of Discrete Wavelet Transform (DWT) [1], denoted as $\mathbf{d}$, to extract the bit results. For the decoding network, we draw inspiration from the task setup utilized in Stable Signature [10], employing a pre-trained watermark extractor network $\mathcal{D}(\cdot)$ from HiDDeN [73] to enhance the decoding accuracy and efficiency. An overall illustration of our proposed framework is shown in Fig. 3.

4.3. Adaptive Bit Modulation

The bit message input by the user is a 0-1 binary string, and we first convert it into a binary tensor $\mathbf{m}$. Next, we employ bit modulation module $\mathcal{B}$ to embed this tensor into outputs of various blocks in the UNet $\mathcal{U}$, and weights of embedding networks are initialized as 0. Compared to directly fine-tuning the UNet, this approach significantly reduces GPU memory consumption, as we only need to train specific modulation layers. Additionally, the difference between the UNet inference results before and after modulation is minimal via such approach, which effectively preserves the generative quality of the 3DGS outputs.

Given the large range of possible values for the 0-1 bits, which is close to the value range of the tensors output by the LGM, directly embedding this binary tensor into the intermediate results of the UNet could lead to a degradation in quality. Furthermore, the impact of different blocks within the UNet on the generated results varies; thus, directly embedding the 0-1 bit tensor into the intermediate results is not a reasonable approach. Based on this analysis and observation, we utilize a set of learnable adaptive coefficients α, β, γ to constrain the degree of modulation.

The specific network architecture is structured as follows: the bit modulation modules $\mathcal{B}$ are located at three stages within the UNet of the LGM, specifically (1) in the input image, named b_{in}, (2) in the middle of UNet blocks b_{mid}, and (3) in the output of UNet b_{out}. The embedding network for the input 0-1 bits consists of a linear layer followed by a SiLU [54] activation function, ultimately producing an output of the appropriate size. This result is then processed through a 2D convolutional layer, multiplied by an adaptive coefficient denoted as α, and added to the input image of the UNet. This modulation process is formulated as follows:

$$\mathbf{m}_{in} = \alpha\, \mathrm{b}_{in}(\mathbf{m}),\ \mathbf{z}_{in} = \mathbf{z} + \mathbf{m}_{in} \tag{6}$$

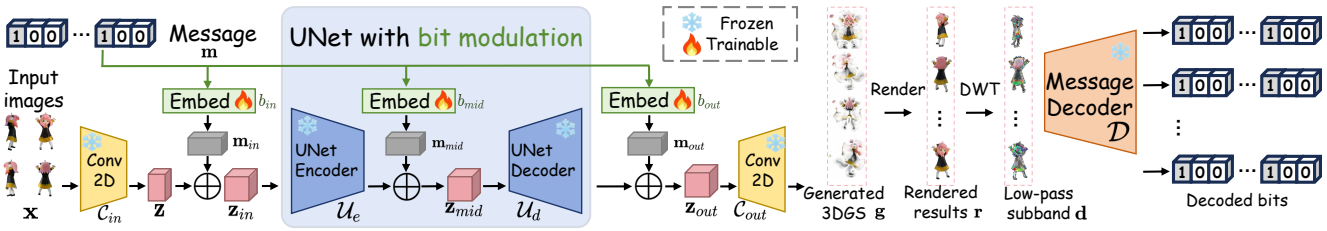

Figure 3. An overview architecture of our proposed 3D generation model watermarking framework, with a detailed illustration of bit modulation embedding and bit decoding process. The input images are transformed into features and processed by the UNet, whose output is converted to 3DGS tensor. The input bit is modulated, multiplied by adaptive coefficients, and embedded at the input, middle, and output of the UNet. This watermark can be extracted from the 3DGS rendering results after DWT.

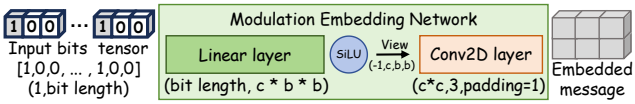

Figure 4. Illustration of bit embedding network. "c" denotes channel size, and "b" denotes block size.

For the modulation modules positioned in the output of UNet, the process is similar. In both cases, the 0-1 bits are embedded into tensors using a linear layer, the SiLU activation function, and 2D convolutional layers to extract further features. The resulting tensor is then multiplied by adaptive coefficients, referred to as β and γ, and added to the intermediate results of the corresponding blocks. The processes are formulated as:

$$\mathbf{m}_{mid} = \beta\, \mathrm{b}_{mid}(\mathbf{m}),\ \mathbf{z}_{mid} = \mathcal{U}_e(\mathbf{z}_{in}) + \mathbf{m}_{mid} \quad (7)$$

$$\mathbf{m}_{out} = \gamma\, \mathrm{b}_{out}(\mathbf{m}),\ \mathbf{z}_{out} = \mathcal{U}_d(\mathbf{z}_{mid}) + \mathbf{m}_{out} \quad (8)$$

This structured approach ensures effective integration of the watermarking information while maintaining the overall generative quality of the model. The specific network module is shown in Fig. 4.

Algorithm 1: GaussianSeal training process

Input: Input multi-view images $\mathbf{x}$, secret message $\mathbf{m}$, Conv2D layers $\mathcal{C}_{in}$ and $\mathcal{C}_{out}$, UNet $\mathcal{U} = \{\mathcal{U}_e, \mathcal{U}_d\}$, bit modulation blocks $\mathcal{B} = \{\mathrm{b}_{in}, \mathrm{b}_{mid}, \mathrm{b}_{out}\}$, adaptive coefficients α, β, γ, watermark decoder $\mathcal{D}$

Output: Bit modulation blocks and adaptive coefficients

for *epoch* $= 1$ **to** *total epochs* **do**
- $\mathbf{z} = \mathcal{C}_{in}(\mathbf{x})$
- Get $\mathbf{z}_{in}$ via bit embedding as Eq. (6).
- Get $\mathbf{z}_{mid}$ via bit embedding as Eq. (7).
- Get $\mathbf{z}_{up}$ via bit embedding as Eq. (8).
- $\mathbf{g} = \mathcal{C}_{out}(\mathbf{z}_{up})$
- $\mathbf{r} = $ 3DGS Render$(\mathbf{z}_{up})$
- $\mathbf{d} = $ DWT$(\mathbf{r})$
- Calculate loss via Eq. (11).
- $\mathcal{B} = \mathcal{B}$.update
- $\alpha, \beta, \gamma = \alpha$.update, β.update, γ.update

end

return Bit modulation blocks $\mathcal{B}$, adaptive coefficients α, β, γ

4.4. Bit Message Decoding

After the generation model gets embedded with the bits, we anticipate that they can be accurately decoded to validate the copyright of both the model and the generated outputs. To achieve this, we explored several common decoding targets, including the 3DGS tensor generated, and the rendered images of the 3DGS.

In our investigations, we focus on the bit extraction performance of each target. We align our approach with the established watermarking methodology, emphasizing robustness and fidelity, which led us to choose the DWT-transformed outputs of the 3DGS renderings as the primary source for decoding the bit results. The DWT is particularly effective for capturing the essential features of the rendered images while preserving the watermark information, and this claim is supported in ablation study in Sec. 5.3. For the watermark extracting network, we leverage HiDDeN [73]'s watermark decoder and pre-train this network on Objaverse [7] corresponding dataset, as there is a severe domain gap between datasets used in pre-training of HiDDeN and Objaverse.

4.5. Training Details

Our training approach employs an end-to-end strategy, where the loss function comprises two components: the bit information loss and the consistency loss between the 3D generation results before and after modulation and the rendered outputs. For the bit loss, we utilize binary cross-entropy (BCE) loss to effectively capture the discrepancies between ground-truth message and decoded result from rendered images, as Eq. (9).

$$\mathcal{L}_{msg} = \mathrm{BCE}(\mathbf{m}, \mathcal{D}(\mathbf{d})) \quad (9)$$

Regarding the consistency loss of the generated results, we simultaneously consider the consistency of the tensor and the consistency of the rendered outputs using mean-squared error (MSE) loss, as Eq. (10):

$$\mathcal{L}_{gs} = \mathrm{MSE}(\mathbf{g}, \mathbf{g}_{clean}),\ \mathcal{L}_{rgb} = \mathrm{MSE}(\mathbf{R}, \mathbf{R}_{clean}) \quad (10)$$

where $\mathbf{g}_{clean}$ and $\mathbf{R}_{clean}$ denote 3DGS tensor and rendered results from original model. Overall training loss is as follows:

$$\mathcal{L} = \mathcal{L}_{msg} + \lambda_{gs}\, \mathcal{L}_{gs} + \lambda_{rgb}\, \mathcal{L}_{rgb} \quad (11)$$

The weights among the various loss components, λ_{gs} and λ_{rgb}, are determined through a grid search process, ensuring

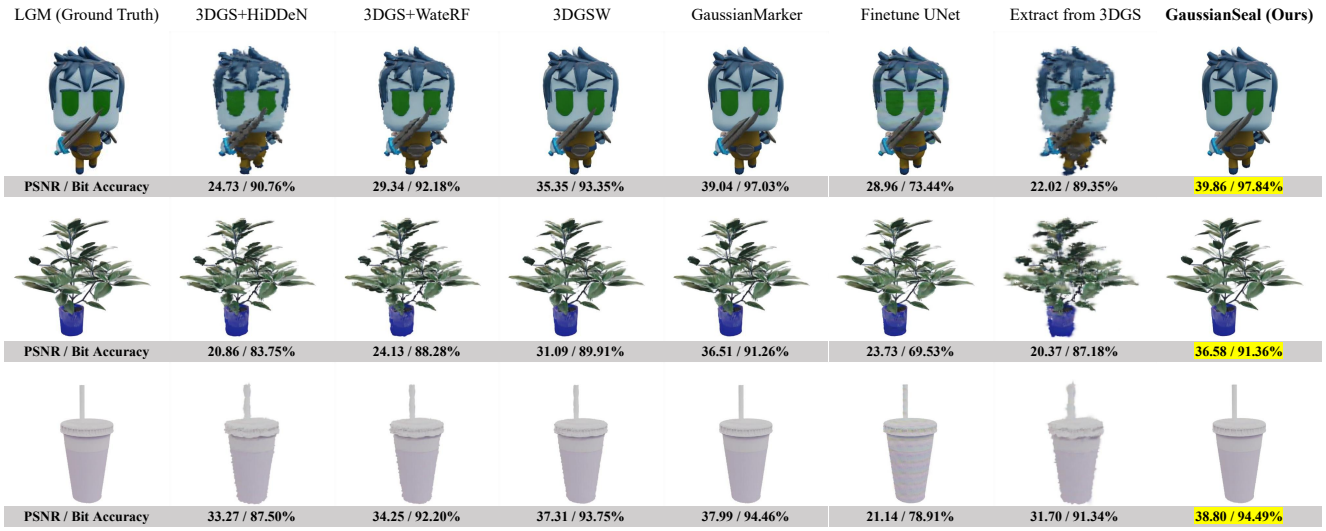

Figure 5. Visualized results of rendered 3D objects watermarked via compared methods, and our GaussianSeal. These results are on 16 bits.

optimal performance. Additionally, the adaptive modulation coefficients discussed in Sec. 4.3 are updated using an auxiliary optimizer, allowing for fine-tuning of the modulation process and enhancing the overall effectiveness of the watermarking framework. The training process of GaussianSeal with detailed modulation is shown in Algo. 1, with parameters of bit modulation network and adaptive coefficients trained through this pipeline.

5. Experiments

5.1. Experimental Settings

Datasets and Pre-training. We select Objaverse dataset [7] as our training and validation dataset, from which we randomly sample 10000 objects for training and 100 objects for validation. Specifically, the data for each object consists of images captured from 38 different viewpoints, along with the corresponding camera intrinsic and extrinsic parameters, viewpoint information, and rotation matrices. All images are standardized to the size of 512×512. For the 3DGS generation model, we use pre-trained weights of LGM [53] without fine-tuning or modification. For the watermark decoder, we pre-train HiDDeN [73] decoder for 16 and 32 bits and fix the parameters during the training process.

Metrics. Our framework aims to achieve higher accuracy in bit decoding while minimizing the impact on the original 3D generative model, thus we evaluate the performance of our proposed watermarking framework in **accuracy** and **visual consistency**. The bit accuracy process is calculated via code from [56]. For visual consistency, the metrics include PSNR, SSIM, and LPIPS [68] between the rendered results of the watermarked model and the original model. We also compare the watermarking time for each 3DGS object, which *in-generation* methods have significant advantages over *post-generation* methods.

Implementation Details. For training bit modulation modules, we choose AdamW [17] as the optimizer with learning rate 1e-4. For adaptive coefficients α, β, and γ, we initialize them to 0.1 and let another AdamW optimizer with a learning rate 1e-3 to update them. For loss balancing weights γ_{gs} and γ_{rgb}, we choose $\gamma_{gs} = 1000$ and $\gamma_{rgb} = 300$. Detailed ablation for these two hyper-parameters is shown in Sec. 5.3. The bit message length is set as 16 and 32 in evaluation (Sec. 5.2), and set as 16 in other experiments. All experiments are done on NVIDIA GTX 3090Ti GPU, and the batch size is set as 2.

5.2. Evaluation of proposed GaussianSeal

Baselines. As we are the first to propose a bit watermarking method for 3D generative models, we select comparison methods to demonstrate the effectiveness of our approach including *post-generation* 3D object watermarking and *in-generation* intuitive watermarking methods mentioned in Sec. 4.1, including: (1) **3DGS+HiDDeN** [73]: train a watermark decoder to extract bits correctly; (2) **3DGS+WateRF** [18]: train a watermarked 3DGS object via WateRF approach; (3) **3D-GSW** [19] current 3DGS watermarking state-of-the-art method via fine-tuning 3DGS representation; (4) **GausssianMarker** [16] current state-of-the-art 3DGS watermarking method via uncertainty estimation; (5) **Fine-tune UNet**: fine-tuning generation UNet to decode bits correctly; (6) **Extract from 3DGS**: use our bit modulation module, bit extract the watermark directly from the tensor representation of the 3DGS.

Quantitative results are shown in Tab. 1, and the visual experimental results are presented in Fig. 5. From the experimental results, we observe that our watermarking framework achieves the best bit decoding accuracy and visual consis-

Method Type	Method	PSNR↑	SSIM↑	LPIPS↓	Bit Accuracy↑	Time (s)↓
Post-Generation	3DGS+HiDDeN [73]	31.8991	0.9067	0.0082	92.83%	786.40
	3DGS+WateRF [18]	32.1800	0.9486	0.0084	94.05%	2184.25
	3D-GSW [19]	33.5937	0.9485	0.0071	94.38%	475.23
	GaussianMarker [16]	37.4583	0.9813	0.0075	97.19%	1228.71
In-Generation	Fine-tune UNet	21.5227	0.9279	0.0819	64.06%	0.18
	Extract from 3DGS	22.6368	0.9525	0.0096	89.58%	**0.12**
	GaussianSeal (Ours)	**38.0228**	**0.9892**	**0.0034**	**97.93%**	0.18

Table 1. Quantitative results of comparison between our proposed framework and other approaches. These results are on 16 bits. Best results are shown in **red**, and second best results are shown in blue.

Method Type	Method	PSNR↑	SSIM↑	LPIPS↓	Bit Accuracy↑	Time (s)↓
Post-Generation	3DGS+WateRF [18]	32.8751	0.9446	0.0088	89.06%	2259.42
	3D-GSW [19]	33.1479	0.9539	0.0079	92.63%	763.54
	GaussianMarker [16]	35.6208	0.9541	0.0056	94.71%	1292.95
In-Generation	GaussianSeal (Ours)	**36.8051**	**0.9583**	**0.0045**	**96.58%**	**0.18**

Table 2. Comparison of bit embedding performance in 32 bits. Best results are shown in **red**, and the second best results are shown in blue.

Decoding Target	PSNR↑	SSIM↑	Bit Accuracy↑
3DGS Tensor	22.6368	0.9525	89.58%
Rendered Images	34.2383	0.9675	96.09%
Rendered Images + DWT	**38.0228**	**0.9892**	**97.93%**

Table 3. Results of performance comparison among different decoding targets. Best results are shown in **red**.

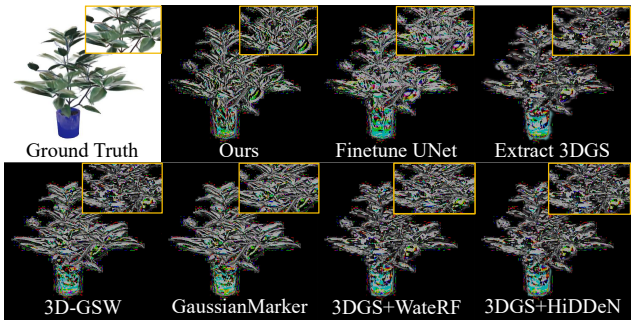

Figure 6. Residual for watermarked rendered images and originally generated rendered results. We put a detailed view in the center of the example plant image. It shows that our watermark keeps more detail of the original image, with the shape of branches and leaves unchanged. Other watermark methods make details broken, resulting in worse quality.

tency. We also provide residual images for watermarked rendered images and vanilla-generated rendered results in Fig. 6. It shows that our watermark preserves the detailed structure of generated images.

5.3. Ablation Study

Here we discuss the key technical choices in our method. **Selection of Watermark Decoding Target.** The choice of decoding target for watermarking is non-trivial. Unlike image generation models that directly decode bits from generated results, we opt to decode from rendered results via DWT. This decision is guided by both the practical insights from WateRF [18] and our own experimental validation. Results are shown in Tab. 3, which shows that the DWT sub-band of rendered results is easier to get decoded bits.

Discussion of Modulation Modules. We incorporate modulation modules at three locations: in the input image feature, in the middle of UNet blocks, and in the output of UNet. What if we add watermarking to only some of these positions? We test with different configurations and concluded that including bit modulation modules at all three locations improves bit decoding performance. Results are presented in Tab. 5. It is shown that all modulation modules function effectively, contributing to precise bit decoding. Without significantly impacting the quality of generation, we opt to add modulation modules at all three locations.

Balancing Loss Weights. We fix the bit message loss weight at 1 and performed a grid search for the Gaussian tensor and RGB rendering loss weights. The results of this grid-search process are shown in Tab. 6. To balance the generation quality (measured by PSNR) and bit decoding accuracy, we choose λ_{gs} as 1000 and λ_{rgb} as 300.

5.4. Method Analysis

Robustness of Our Watermark. Current attacks on 3DGS objects primarily involve pruning, where the Gaussian structure is disrupted by reducing the number of Gaussian points in the point cloud. To assess the robustness of our watermarking framework against this type of attack, we conducted tests with results shown in Tab. 7, and a visualized result of pruned 3DGS point cloud with decoded accuracy is shown in Fig. 7. We also test our watermark framework's robustness on common image augmentation attacks. Results are shown

Method / Attack	Bit Accuracy(%)↑					
	Gaussian Noise ($\mu = 0.1$)	Crop (40%)	Rotation (60°)	Brightness (2.0)	JPEG Compression (Quality 10%)	Gaussian Blur (kernel size = 5)
3DGS+WateRF [18]	95.50	95.73	92.59	92.73	95.42	95.20
3D-GSW [19]	94.93	**95.75**	95.25	94.84	96.12	96.29
GaussianMarker [16]	96.18	95.37	93.94	94.37	90.63	**98.36**
GaussianSeal (Ours)	**96.32**	95.20	**96.34**	**97.26**	**96.81**	97.02

Table 4. Robustness test result on common image augmentation attacks. Best results are shown in **red**, and second best results shown in blue.

Input	Middle of blocks	Output	PSNR	Bit Acc
×	×	×	inf	64.53%
✓	×	×	38.7487	89.20%
×	✓	×	38.7396	84.31%
×	×	✓	38.7589	92.96%
✓	✓	×	38.1323	97.65%
✓	×	✓	38.1967	93.19%
×	✓	✓	38.2432	95.30%
✓	✓	✓	38.0228	97.93%

Table 5. Ablation of bit modulation on different positions in UNet.

λ_{gs}	λ_{rgb}	PSNR	Bit Acc
3000	100	43.1517	86.71%
2000	100	40.1112	91.40%
1000	100	35.0509	92.18%
1000	200	35.2704	95.31%
1000	**300**	**38.0228**	**97.93%**
1000	400	38.1557	96.09%
500	100	32.6613	70.31%

Table 6. Ablation results of different choices on weights of Gaussian tensor loss and RGB loss. Our choice is shown in **bold**.

Ratio	PSNR	SSIM	LPIPS	Bit Acc
5%	27.6489	0.9762	0.0082	96.37%
10%	25.3123	0.9156	0.0097	92.75%
15%	24.3196	0.8978	0.0183	91.96%
25%	23.0801	0.8803	0.0245	89.20%

Table 7. Robustness analysis of proposed watermarking framework under different 3DGS pruning ratios.

in Tab. 4. Results demonstrate that our watermark is robust to common attacks, both in 3D and 2D domain.

Security Analysis of Our Watermark. Additionally, we evaluate the exposure risk of our embedded watermark using the open-source tool StegExpose [2]. Detailed results are shown in Supplementary Materials, indicating that our watermark is sufficiently secure and difficult to detect.

5.5. Limitation and Discussion

The common issue in the current field of model watermarking is the large demand for dataset scale. T2I model watermarking methods such as LaWa [44] and AquaLoRA [9], as well as our GaussianSeal for 3D generative model, all require large amounts of data for training to simultaneously achieve convergence of the watermark loss and maintain generation quality. Our watermarking technique is also limited to protecting the copyright of 3DGS objects generated by AI models. In future work, we aim to expand our approach to include watermarking for point clouds and meshes rendered from generated 3DGS objects. This will enable us to achieve comprehensive watermark coverage across the entire process of 3D generation, thereby enhancing the robustness and versatility of our watermarking framework.

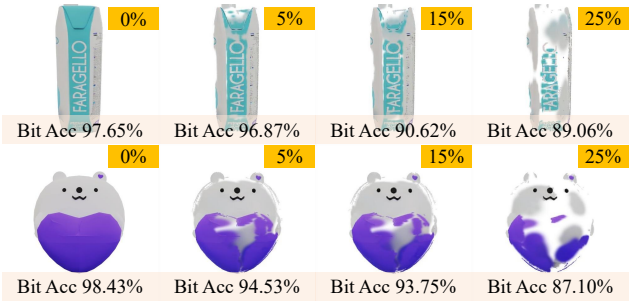

Figure 7. Visualization of pruned 3DGS point clouds, with each decoded accuracy. The ratio on the image denotes the prune ratio.

6. Conclusion

In this paper we propose the first bit watermarking framework for 3D Gaussian generative models, named GaussianSeal, to protect the copyright of 3D generative models and their generated results. We design an adaptive bit watermark modulation module that is able to effectively embed bit watermarks into generative model's blocks, achieving accurate watermark decoding while maintaining the consistency and fidelity of original generation quality for both 3D Gaussian model and the rendered images, which is challenging. In future work, we aim to extend GaussianSeal to generalize across multiple bit messages, enabling both model owner verification and generator tracing. We will also consider expanding the decoding targets of the watermark to rendered results such as point clouds and meshes from 3DGS, thereby achieving versatile watermarking across multiple modalities. In summary, our method effectively protects the copyright of AIGC models and corresponding 3D assets.

GaussianSeal: Rooting Adaptive Watermarks for 3D Gaussian Generation Model

Supplementary Material

In the supplementary materials, we demonstrate additional experimental results, implementation details, discussion, and analysis as follows.

Contents

7. Implementation Details

7.1. Dataset Details

For the Objaverse dataset, each object has 38 views, and each view is 512×512 with an RGB value range from 0 to 1 and set in white background. The camera system loaded in raw dataset is Blender world and OpenCV camera, and we transform it into OpenGL world and OpenGL camera.

For the NeRF Synthetic dataset [35] in additional evaluation, for each object we choose one view that contains most of visual information of the object, and follow the same LGM inference pipeline used in the evaluation section in the main paper.

7.2. PyTorch-like Pseudo Code of Adaptive Bit Modulation

We here provide PyTorch-like pseudo code for adaptive bit modulation modules definition, and how they are applied in LGM inference process.

8. Additional Experiments

8.1. Evaluation on NeRF Synthetic Dataset

To evaluate the generalization ability of our proposed GaussianSeal, we leverage the bit modulation trained on the Objaverse dataset and test on the NeRF Synthetic dataset which is not used during the training process. Details of dataset settings can be referred from Sec. 7.1. Quantitative results are shown in Tab. 8, and qualitative results are shown in Fig. 8.

8.2. Evaluation of Generalization and Scalability

In the main experiments, we randomly select 10,000 and 100 objects separately from Objaverse dataset for training and validation. These two subsets **do not have overlap**. To validate the generalization and scalability of GaussisanSeal, we have done the validation on 10,000 and 100,000 objects from Objaverse dataset, which also have no overlap with training and validation datasets. The results are as below in Tab. 9, which shows that our method has good **scalability and generalizability** on larger unseen datasets.

8.3. Watermark Capacity of GaussianSeal

In the experiments section of the main paper, we hide 16 and 32 bits in the 3D generation model, and here, we try to hide more bits to explore the upper limit of the bit watermarking capacity of our method. We extend message length to 48 and 64, and provide the quantitative results in Tab. 10.

Algorithm 2: Adaptive Bit Modulation

```
import torch.nn as nn
class View(nn.Module):
    def forward(self, x):
        return x.view(*self.shape)
# define modulation modules
# input
noise_block_size = 8
message_len = 16
b_in = nn.Sequential(
    nn.Linear(message_len, 4 * noise_block_size * noise_block_size), nn.SiLU(),
    View(-1, 4 , noise_block_size, noise_block_size), )
b_in_conv = torch.nn.Conv2d(4, 4, 3, padding=1)
# middle
b_mid = nn.Sequential(
    nn.Linear(message_len, 512 * noise_block_size * noise_block_size), nn.SiLU(),
    View(-1, 512 , noise_block_size, noise_block_size), )
b_mid_conv = torch.nn.Conv2d(512,512,3,padding=1)
# output
b_out = nn.Sequential(
    nn.Linear(message_len, 512 * noise_block_size * noise_block_size), nn.SiLU(),
    View(-1, 512 , noise_block_size, noise_block_size), )
b_out_conv = torch.nn.Conv2d(512,512,3,padding=1)
# define adaptive coefficients
watermark_alpha = nn.Parameter(torch.tensor(0.1))
watermark_beta = nn.Parameter(torch.tensor(0.1))
watermark_gamma = nn.Parameter(torch.tensor(0.1))
# forward process for UNet with modulation
def forward_with_watermark(x, m): # x is input image, m is message
    H, W, B, C = x.shape[2], x.shape[3], x.shape[0], x.shape[1]
    # modulation in input stage
    m_in = b_in(m).repeat(B,1,int(H/noise_block_size),int(W/noise_block_size))
    m_in = b_in_conv(m).repeat(1,3,1,1)[:, :9, :, :]
    x = conv_in(x)
    x = x + watermark_alpha * m_in
    # encoder
    x = unet.encoder(x)
    H,W = x.shape[2], x.shape[3]
    # modulation in middle
    m_mid = b_mid(m).repeat(B,1,int(H/noise_block_size),int(W/noise_block_size))
    repeat_times = int(x.shape[1]/m_mid.shape[1])
    m_mid = b_mid_conv(m_mid).repeat(1,repeat_times,1,1)
    x = x + watermark_beta * m_mid
    # decoder
    x = unet.decoder(x)
    H, W = x.shape[2], x.shape[3]
    m_mid = b_out(m).repeat(B,1,int(H/noise_block_size),int(W/noise_block_size))
    m_mid = b_out_conv(m_mid)[:, :x.shape[1], :, :]
    x = x + watermark_gamma * m_mid

    # conv out to get output 3DGS tensor
    x = conv_out(x)
    return x
```

Metrics	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship	Overall
PSNR	35.7522	34.3105	32.3541	34.8191	32.9918	34.4048	35.1800	32.2207	34.0042
SSIM	0.9413	0.9283	0.9161	0.9219	0.9081	0.9440	0.9421	0.9535	0.9319
LPIPS	0.0083	0.0019	0.0016	0.0029	0.0013	0.0076	0.0037	0.0009	0.0035
Bit Acc	95.22%	94.98%	94.21%	91.69%	92.56%	95.65%	90.52%	92.89%	93.46%

Table 8. Additional results of GaussianSeal on NeRF Synthetic dataset.

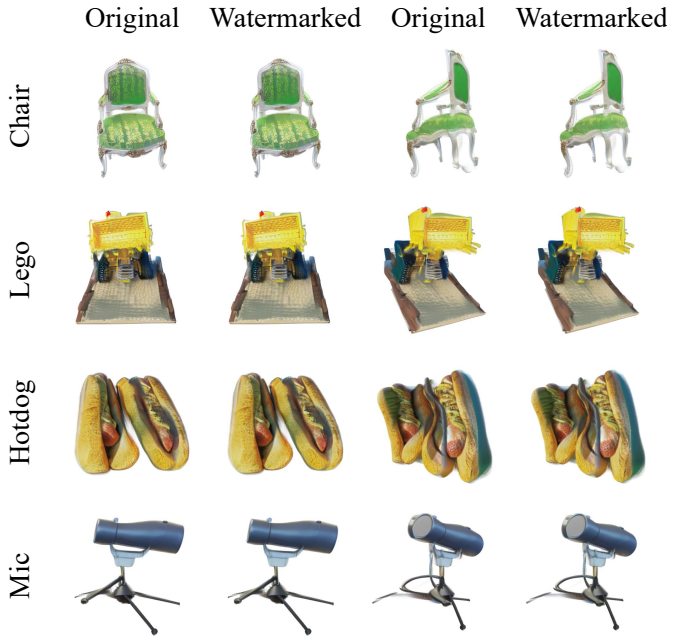

Figure 8. Visualized results of objects from NeRF Synthetic dataset, generated via original LGM model and watermarked LGM model.

Amount	PSNR	SSIM	LPIPS	Bit Acc
10,000	37.4815	0.9705	0.0038	96.41%
100,000	37.0329	0.9639	0.0056	96.27%

Table 9. Results of larger-scale validation.

8.4. Results on novel views

We supply experiments on novel views here. Results are shown in Tab. 11, which show that our method is **generalizable** to novel views.

8.5. Robustness against 3DGS compressing

We adopt C3DGS [23] to the generated 3DGS objects, which are compressed by 25 times. Results shown below in Tab. 12.

8.6. Robutness against fine-tuning attack

In our threat model, we envision a scenario where the model owner provides an API-like service, allowing users and attackers to input images and receive a watermarked 3DGS model, but without access to the model's structure or weights. To address security concerns, we demonstrate the model's robustness against mild fine-tuning attacks in Tab. 13. Such attack involves fine-tuning the LGM network weights using a loss function that excludes bit loss, which reduces the watermark performance of GaussianSeal.

8.7. Results of Security Analysis

To evaluate the security of our GaussianSeal, we conduct anti-steganography detection using StegExpose [2] on container images generated by various methods, including 3D-GSW [19], GaussianMarker [16], 3DGS+HiDDeN [73], 3DGS+WateRF [18], and our GaussianSeal. Each method embeds 16 bits into 3DGS objects. We adjust the detection thresholds across a wide range, from 0.001 to 0.95, within StegExpose [2], and plot the resulting receiver operating characteristic (ROC) curve in Fig. 9. The ideal scenario assumes the detector has a 50% chance of identifying a watermark in a balanced test set, equivalent to random guessing. The results clearly demonstrate that our method significantly outperforms 3DGS watermarking methods in terms of security. This indicates that our approach is much less susceptible to detection by steganography analysis techniques, thereby

Message Length	Method	PSNR	SSIM	LPIPS	Bit Acc
48	GaussianMarker	32.8155	0.9373	0.0088	91.14%
	GaussianSeal (Ours)	33.0690	0.9463	0.0078	91.67%
64	GaussianMarker	30.9979	0.9053	0.0061	90.34%
	GaussianSeal (Ours)	31.3665	0.9267	0.0045	92.77%

Table 10. Watermark capacity exploration of our method, compared to current 3DGS watermarking state-of-the-art method GaussianMarker.

PSNR	SSIM	LPIPS	Bit Acc
32.2195	0.9617	0.0060	96.61%

Table 11. Results of novel views unseen in training.

PSNR	SSIM	LPIPS	Bit Acc
35.6889	0.9458	0.0089	93.51%

Table 12. Results of robustness validation on 3DGS compression.

Steps	Bit Acc
20	96.88%
40	94.53%
60	92.97%
80	91.26%
100	89.84%

Table 13. Results of robutsness against fine-tuning attack.

achieving a robust level of security.

9. Discussion and Analysis

9.1. GaussianSeal with 3D feature

Our GaussianSeal method draws on some watermarking practices used in Text-to-Image (T2I) models and incorporates specialized designs for the representation of 3D objects and scenes, as well as properties unique to 3DGS, to avoid compromising the generation quality of 3D models. Specifically, our considerations and designs are as follows:

❑ We observe that the attributes of 3DGS are highly sensitive to changes in values; even minor parameter adjustments can cause significant degradation in the visual quality of 3DGS point clouds, resulting in issues such as Gaussian point dispersion and irregular deformation. Based on these observations, we (1) embed the watermark into the output of the UNet rather than into the final 3DGS tensor used for representing 3D objects, and (2) introduce an adaptive coefficient for the embedded watermark tensor to minimize its impact on the original model output values. Similarly, we opt for bit modulation of the watermark instead of directly fine-tuning the UNet, as shown in the main paper, where direct fine-tuning of the UNet leads to artifacts in the generated 3DGS results.

❑ For 3DGS generative models, such as LGM, although they are similar to commonly used diffusion models in image generation, directly applying diffusion watermarking methods such as fine-tuning, designing LoRA, or adapter-based methods still results in reduced model quality. This is because the latent code in diffusion models is trained through a noise-adding and denoising process, making the UNet robust to noise in the latent code and capable of producing good outputs even when disturbances occur. However, in 3D generative models like LGM, the training does not involve robustness to noise, so changes in UNet weights and intermediate results between blocks lead to significant changes in the generated output. Therefore, a better approach for embedding bit information is needed. Through experiments, we find that using an additional lightweight modulation module to embed bit information at key locations achieves a better trade-off between precise bit decoding and maintaining generation quality.

❑ Regarding watermark extraction, previous studies on watermarking methods for 3D objects and scenes often choose to decode from the DWT low-low subband of rendered results to improve decoding accuracy. We adopt this mechanism in our watermarking of 3D generative models, and experiments show that such an approach is also effective in 3D generation model watermarking.

9.2. Relationship to other generation model watermarking methods

First, we define the terms *post-generation* and *in-generation* watermark methods mentioned in the main paper. The *post-generation* watermarking method refers to the technique of adding watermarks to the generated results of a 3D Gaussian generation **after** the model produce its output. Since this method is a post-processing step and is independent and decoupled from the specific generation process of the model, it is termed post-generation. This approach requires adding watermarks to each generated result individually, thus necessitating a longer time and greater consumption of computational resources.

The *in-generation* watermarking method refers to the process where watermarks are embedded **during** the generation process of the model. This method is closely coupled and tightly integrated with the generation model, such that the results produced by the model already contain the watermarks,

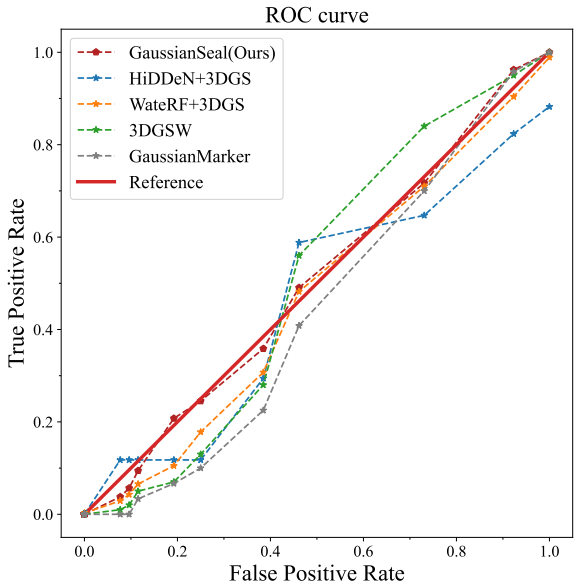

Figure 9. The ROC curve of various methods under a steganography detector. A curve closer to the reference central axis (indicating random guessing) signifies better security for the corresponding method.

eliminating the need for any post-processing. The advantage of this method is that with a single training session, one can directly obtain the target with the watermarks embedded.

We compare our GaussianSeal with existing in-generation T2I generation model watermarking methods which are similar to ours, highlighting the differences in approach and implementation between the proposed GaussianSeal and the methods mentioned below.

❑ **Compared to AquaLoRA**: AquaLoRA [9] fine-tunes the UNet of Stable Diffusion (SD) using LoRA to enable the extraction of bit information from the generated results. This method requires substantial computational resources and extensive training periods—our previous attempts indicated that it necessitates over 72 hours of training on an A100 GPU—and it can adversely affect the quality of the model generation. This approach is markedly different from our proposed solution.

❑ **Compared to LaWa**: LaWa [44] achieves precise bit decoding by adding a bit embedding module to the decoder of SD and training it. In contrast, our method opts to incorporate a bit modulation mechanism within the UNet, and considering the sensitivity of 3DGS to weight variations, we have introduced adaptive value range compression to prevent the generation of poor-quality results.

❑ **Compared to WaDiff**: WaDiff [37] decodes bits by fine-tuning the first layer of SD UNet and training an additional bit embedding layer. Our approach, on the other hand, completely freezes the original generation model, which is a consideration aimed at addressing the fragility of LGM model weights.

In short, our method exhibits distinct differences in implementation and conceptual approach compared to previous watermarking techniques for generative models. These differences stem from our observations and considerations of the properties of 3DGS and its generative models.

10. More Visualized Results

Here we provide more visualized results of 3DGS objects watermarked by GaussianSeal in Fig. 10, including original generated object, watermarked object, and their residual image.

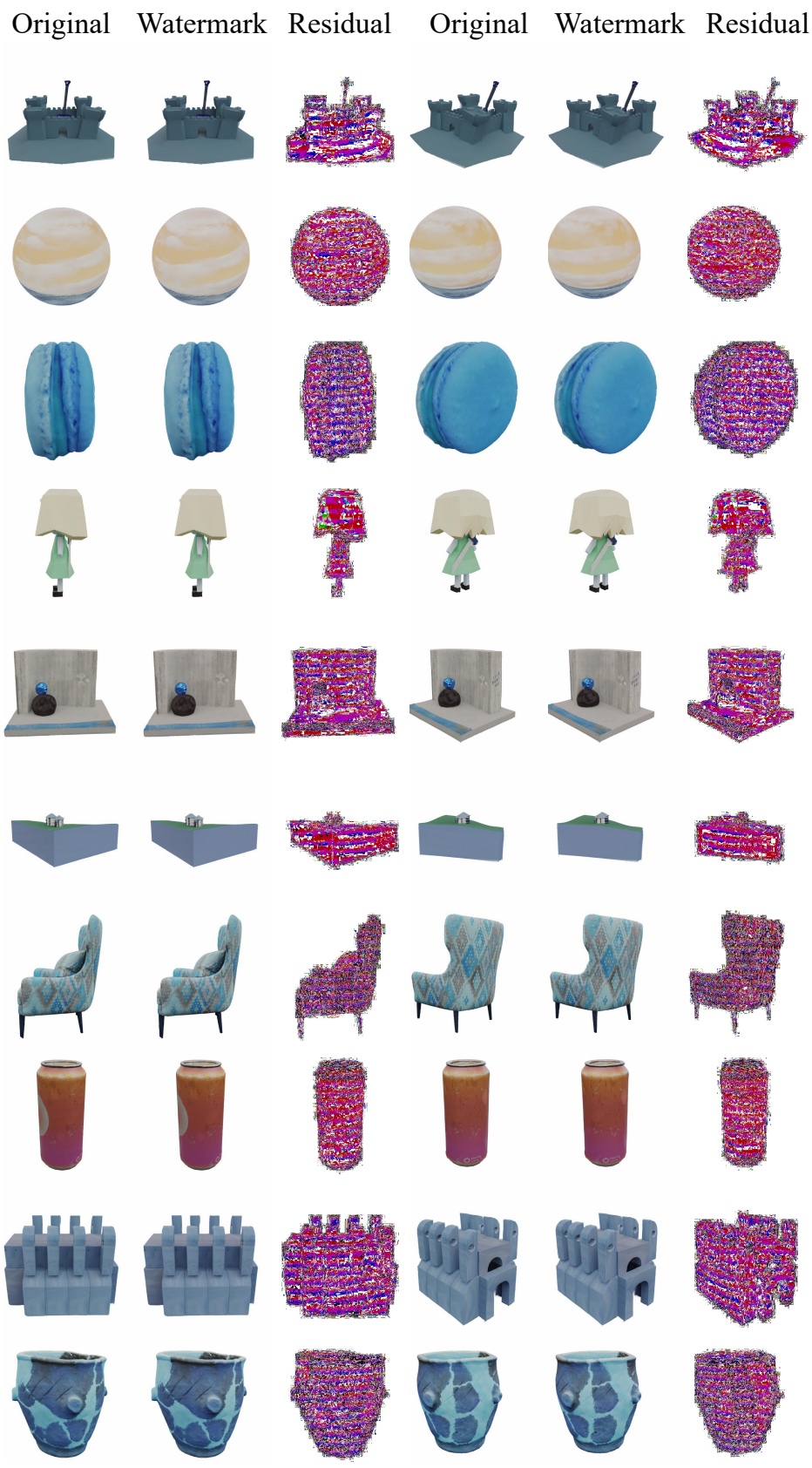

Figure 10. More visualized results of GaussianSeal, each object with two views.

References

[1] Ali Naci Akansu, Richard A Haddad, and Hakan Caglar. Perfect reconstruction binomial qmf-wavelet transform. In *Visual Communications and Image Processing'90: Fifth in a Series*, pages 609–618. SPIE, 1990. 4

[2] Benedikt Boehm. Stegexpose-a tool for detecting lsb steganography. *arXiv preprint arXiv:1410.6656*, 2014. 8, 11

[3] David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19457–19467, 2024. 3

[4] Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images. In *European Conference on Computer Vision*, pages 370–386. Springer, 2025. 3

[5] Hai Ci, Yiren Song, Pei Yang, Jinheng Xie, and Mike Zheng Shou. Wmadapter: Adding watermark control to latent diffusion models. *arXiv preprint arXiv:2406.08337*, 2024. 1

[6] Hai Ci, Pei Yang, Yiren Song, and Mike Zheng Shou. Ringid: Rethinking tree-ring watermarking for enhanced multi-key identification. In *European conference on computer vision (ECCV)*, 2024. 2

[7] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. *arXiv preprint arXiv:2212.08051*, 2022. 5, 6

[8] Ben Fei, Jingyi Xu, Rui Zhang, Qingyuan Zhou, Weidong Yang, and Ying He. 3d gaussian as a new vision era: A survey. *arXiv preprint arXiv:2402.07181*, 2024. 1

[9] Weitao Feng, Wenbo Zhou, Jiyan He, Jie Zhang, Tianyi Wei, Guanlin Li, Tianwei Zhang, Weiming Zhang, and Nenghai Yu. Aqualora: Toward white-box protection for customized stable diffusion models via watermark lora. In *International Conference on Machine Learning (ICML)*, 2024. 1, 2, 8, 13

[10] 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. 1, 2, 4

[11] Xianglong He, Junyi Chen, Sida Peng, Di Huang, Yangguang Li, Xiaoshui Huang, Chun Yuan, Wanli Ouyang, and Tong He. Gvgen: Text-to-3d generation with volumetric representation. In *European Conference on Computer Vision*, pages 463–479, 2024. 3

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[13] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022. 1

[14] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 3

[15] Xiufeng Huang, Ka Chun Cheung, Simon See, and Renjie Wan. Geometrysticker: Enabling ownership claim of recolorized neural radiance fields. In *European Conference on Computer Vision*, pages 438–454, 2024. 2

[16] Xiufeng Huang, Ruiqi Li, Yiu-ming Cheung, Ka Chun Cheung, Simon See, and Renjie Wan. Gaussianmarker: Uncertainty-aware copyright protection of 3d gaussian splatting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 1, 3, 6, 7, 8, 11

[17] Loshchilov Ilya and Hutter Frank. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019. 6

[18] Youngdong Jang, Dong In Lee, MinHyuk Jang, Jong Wook Kim, Feng Yang, and Sangpil Kim. Waterf: Robust watermarks in radiance fields for protection of copyrights. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2024. 2, 4, 6, 7, 8, 11

[19] Youngdong Jang, Hyunje Park, Feng Yang, Heeju Ko, Euijin Choo, and Sangpil Kim. 3d-gsw: 3d gaussian splatting watermark for protecting copyrights in radiance fields. *arXiv preprint arXiv:2409.13222*, 2024. 1, 3, 6, 7, 8, 11

[20] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 2, 3

[21] Changhoon Kim, Kyle Min, Maitreya Patel, Sheng Cheng, and Yezhou Yang. Wouaf: Weight modulation for user attribution and fingerprinting in text-to-image diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8974–8983, 2024. 2

[22] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2013. 2

[23] Joo Chan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. Compact 3d gaussian representation for radiance field. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21719–21728, 2024. 11

[24] Liangqi Lei, Keke Gai, Jing Yu, and Liehuang Zhu. Diffusetrace: A transparent and flexible watermarking scheme for latent diffusion model. *arXiv preprint arXiv:2405.02696*, 2024. 2

[25] Chenxin Li, Brandon Y Feng, Zhiwen Fan, Panwang Pan, and Zhangyang Wang. Steganerf: Embedding invisible information within neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 441–453, 2023. 2

[26] Chenghao Li, Chaoning Zhang, Joseph Cho, Atish Waghwase, Lik-Hang Lee, Francois Rameau, Yang Yang, Sung-Ho Bae, and Choong Seon Hong. Generative ai meets 3d: A survey on text-to-3d in aigc era. *arXiv preprint arXiv:2305.06131*, 2023. 1

[27] Chenxin Li, Hengyu Liu, Zhiwen Fan, Wuyang Li, Yifan Liu, Panwang Pan, and Yixuan Yuan. Gaussianstego: A generalizable stenography pipeline for generative 3d gaussians splatting. *arXiv preprint arXiv:2407.01301*, 2024. 3

[28] Runyi Li, Xuhan Sheng, Weiqi Li, and Jian Zhang. Omnissr: Zero-shot omnidirectional image super-resolution using stable diffusion model. In *European Conference on Computer Vision*, pages 198–216, 2024. 2

[29] Runyi Li, Xuanyu Zhang, Zhipei Xu, Yongbing Zhang, and Jian Zhang. Protect-your-ip: Scalable source-tracing and attribution against personalized generation. *arXiv preprint arXiv:2405.16596*, 2024. 2

[30] Yixun 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2024. 1

[31] Yugeng Liu, Zheng Li, Michael Backes, Yun Shen, and Yang Zhang. Watermarking diffusion model. *arXiv preprint arXiv:2305.12502*, 2023. 2

[32] Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20654–20664, 2024. 3

[33] Ziyuan Luo, Qing Guo, Ka Chun Cheung, Simon See, and Renjie Wan. Copyrnerf: Protecting the copyright of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22401–22411, 2023. 2

[34] Zhiyuan Ma, Guoli Jia, Biqing Qi, and Bowen Zhou. Safe-sd: Safe and traceable stable diffusion with text prompt trigger for invisible generative watermarking. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 7113–7122, 2024. 1

[35] 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. In *European conference on computer vision (ECCV)*, 2020. 9

[36] 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

[37] Rui Min, Sen Li, Hongyang Chen, and Minhao Cheng. A watermark-conditioned diffusion model for ip protection. In *European conference on computer vision (ECCV)*, 2024. 1, 2, 13

[38] Rui Min, Zeyu Qin, Nevin L Zhang, Li Shen, and Minhao Cheng. Uncovering, explaining, and mitigating the superficial safety of backdoor defense. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 1

[39] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[40] Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, and Ying Shan. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 4296–4304, 2024. 2

[41] Ryutarou Ohbuchi, Akio Mukaiyama, and Shigeo Takahashi. A frequency-domain approach to watermarking 3d shapes. In *Computer graphics forum*, pages 373–382, 2002. 2

[42] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023. 1

[43] Sen Peng, Yufei Chen, Cong Wang, and Xiaohua Jia. Intellectual property protection of diffusion models via the watermark diffusion process. *arXiv preprint arXiv:2306.03436*, 2023. 1

[44] Ahmad Rezaei, Mohammad Akbari, Saeed Ranjbar Alvar, Arezou Fatemi, and Yong Zhang. Lawa: Using latent space for in-generation image watermarking. In *European conference on computer vision (ECCV)*, 2024. 1, 2, 8, 13

[45] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1

[46] Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation. In *International Conference on Learning Representations (ICLR)*, 2023. 4

[47] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *International Conference on Learning Representations (ICLR)*, 2023. 1

[48] Gursimran Singh, Tianxi Hu, Mohammad Akbari, Qiang Tang, and Yong Zhang. Towards secure and usable 3d assets: A novel framework for automatic visible watermarking. *arXiv preprint arXiv:2409.00314*, 2024. 2

[49] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2021. 2

[50] Qi Song, Ziyuan Luo, Ka Chun Cheung, Simon See, and Renjie Wan. Geometry cloak: Preventing tgs-based 3d reconstruction from copyrighted images. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 1

[51] Qi Song, Ziyuan Luo, Ka Chun Cheung, Simon See, and Renjie Wan. Protecting nerfs' copyright via plug-and-play watermarking base model. In *European conference on computer vision (ECCV)*, 2024. 2

[52] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021. 2

[53] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *European Conference on Computer Vision*, pages 1–18, 2024. 1, 2, 3, 4, 6

[54] Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 25(7):926–930, 2018. 4

[55] Ruofei Wang, Renjie Wan, Zongyu Guo, Qing Guo, and Rui Huang. Spy-watermark: Robust invisible watermarking for backdoor attack. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2700–2704. IEEE, 2024. 2

[56] 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. 2, 6

[57] Cheng Xiong, Chuan Qin, Guorui Feng, and Xinpeng Zhang. Flexible and secure watermarking for latent diffusion model. In *Proceedings of the 31st ACM International Conference on Multimedia*, page 1668–1676. Association for Computing Machinery, 2023. 2

[58] Yinghao Xu, Zifan Shi, Wang Yifan, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wetzstein. Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. In *European conference on computer vision (ECCV)*, 2024. 1, 3

[59] Zhipei Xu, Xuanyu Zhang, Runyi Li, Zecheng Tang, Qing Huang, and Jian Zhang. Fakeshield: Explainable image forgery detection and localization via multi-modal large language models. In *International Conference on Learning Representations (ICLR)*, 2025. 1

[60] Zijin Yang, Kai Zeng, Kejiang Chen, Han Fang, Weiming Zhang, and Nenghai Yu. Gaussian shading: Provable performance-lossless image watermarking for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12162–12171, 2024. 2

[61] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 1, 3

[62] Innfarn 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10031–10040, 2022. 2

[63] Jiwen Yu, Xuanyu Zhang, Youmin Xu, and Jian Zhang. Cross: Diffusion model makes controllable, robust and secure image steganography. In *Advances in Neural Information Processing Systems*, 2023. 2

[64] Zhiqiang Yu, Horace HS Ip, and LF Kwok. A robust watermarking scheme for 3d triangular mesh models. *Pattern recognition*, 36(11):2603–2614, 2003. 2

[65] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19447–19456, 2024. 3

[66] Zihan Yuan, Li Li, Zichi Wang, and Xinpeng Zhang. Watermarking for stable diffusion models. *IEEE Internet of Things Journal*, 2024. 2

[67] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023. 2

[68] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 6

[69] Xuanyu Zhang, Runyi Li, Jiwen Yu, Youmin Xu, Weiqi Li, and Jian Zhang. Editguard: Versatile image watermarking for tamper localization and copyright protection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1

[70] Xuanyu Zhang, Jiarui Meng, Runyi Li, Zhipei Xu, Yongbing Zhang, and Jian Zhang. Gs-hider: Hiding messages into 3d gaussian splatting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 1, 3

[71] Xuanyu Zhang, Youmin Xu, Runyi Li, Jiwen Yu, Weiqi Li, Zhipei Xu, and Jian Zhang. V2a-mark: Versatile deep visual-audio watermarking for manipulation localization and copyright protection. In *Proceedings of the 31st ACM International Conference on Multimedia*, 2024. 1

[72] Yunqing Zhao, Tianyu Pang, Chao Du, Xiao Yang, Ngai-Man Cheung, and Min Lin. A recipe for watermarking diffusion models. *arXiv preprint arXiv:2303.10137*, 2023. 2

[73] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks. In *European Conference on Computer Vision (ECCV)*, 2018. 3, 4, 5, 6, 7, 11