DreaMark: Rooting Watermark in Score Distillation Sampling Generated Neural Radiance Fields

Xingyu Zhu[1,2], **Xiapu Luo**[2], **Xuetao Wei**[1*]

[1]Department of Computer Science and Engineering, Southern University of Science and Technology, China
[2]Department of Computing, Hong Kong Polytechnic University, Hong Kong
12150086@mail.sustech.edu.cn, csxluo@comp.polyu.edu.hk, weixt@sustech.edu.cn

Abstract

Recent advancements in text-to-3D generation can generate neural radiance fields (NeRFs) with score distillation sampling, enabling 3D asset creation without real-world data capture. With the rapid advancement in NeRF generation quality, protecting the copyright of the generated NeRF has become increasingly important. While prior works can watermark NeRFs in a post-generation way, they suffer from two vulnerabilities. First, a delay lies between NeRF generation and watermarking because the secret message is embedded into the NeRF model post-generation through fine-tuning. Second, generating a non-watermarked NeRF as an intermediate creates a potential vulnerability for theft. To address both issues, we propose DREAMARK to embed a secret message by backdooring the NeRF during NeRF generation. In detail, we first pre-train a watermark decoder. Then, DREAMARK generates backdoored NeRFs in a way that the target secret message can be verified by the pre-trained watermark decoder on an arbitrary trigger viewport. We evaluate the generation quality and watermark robustness against image- and model-level attacks. Extensive experiments show that the watermarking process will not degrade the generation quality, and the watermark achieves 90+% accuracy among both image-level attacks (*e.g.*, Gaussian noise) and model-level attacks (*e.g.*, pruning attack).

Introduction

Digital 3D content has become indispensable in Metaverse and virtual and augmented reality, enabling visualization, comprehension, and interaction with complex scenes that represent our real lives. Recent progress in 3D content generation (Poole et al. 2022; Lin et al. 2023; Wang et al. 2024; Liang et al. 2024) can generate high-quality 3D assets that need a lot of time, computational resources, and skilled expertise. Therefore, protecting the ownership of generated 3D content has become more critical.

We focus on Text-to-3D generation (Poole et al. 2022; Lin et al. 2023; Wang et al. 2024; Liang et al. 2024) and the neural radiance field (NeRF) (Mildenhall et al. 2021; Müller et al. 2022), which have emerged into the spotlight in 3D content modeling. Current trending 3D generation algorithms generate 3D representations such as meshes and NeRFs. This paper focuses on NeRF generation since NeRF can represent 3D models more compactly. Given a textual description, recent text-to-3D methods generate NeRFs by distilling pre-trained diffusion models, such as Stable Diffusion (Rombach et al. 2022). This remarkable progress is grounded in the use of Score Distillation Sampling (SDS). With SDS, NeRF training can be conducted without realistic images. Thus, the research question we address in this paper is: *how to protect the score distillation sampling generated neural radiance fields?*

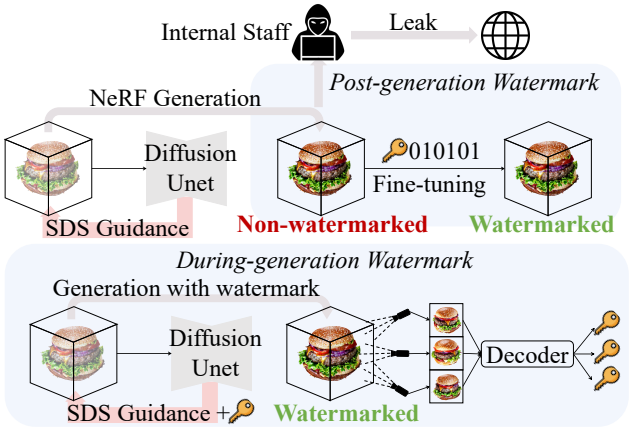

Figure 1: Attack scenario. If company-generated content is considered company property, internal staff could steal non-watermarked intermediates in the post-generation pipeline (top row). However, such intermediates do not exist in the during-generation pipeline (bottom row).

One way to protect the generated NeRF is to apply post-generation watermarking methods, such as CopyRNeRF (Luo et al. 2023) and WateNeRF (Jang et al. 2024), to watermark NeRF after it is generated. However, these methods exhibit two problems. First, post-generation methods pose a risk of data leakage. As shown in Figure 1, since non-watermarked intermediates are generated in the post-generation pipeline, a malicious user could leak the non-watermarked version of the generated content. Second, CopyRNeRF increases the watermarking expense since it requires an additional message feature field in the NeRF structure. Integrating CopyRNeRF with an arbitrary text-

*Corresponding Author.

to-NeRF pipeline requires additional modifications to the NeRF structure. Recognizing these limitations of previous work, can we conduct a *during-generation* watermarking without modifying the NeRF structure?

We propose DREAMARK, the first *during-generation* text-to-3D watermarking method, which is gracefully combined with score distillation sampling to generate high-quality and watermarked NeRF. Different from *post-generation* NeRF watermarking method, DREAMARK directly generates watermarked NeRF without changing NeRF architecture, increasing the flexibility for future development on 3D generation. Our method is inspired by black-box model watermarking methods (Adi et al. 2018; Zhang et al. 2018; Jia et al. 2021; Le Merrer, Perez, and Trédan 2020; Chen, Rouhani, and Koushanfar 2019; Szyller et al. 2021) which watermark a deep neural network by injecting backdoors. To inject backdoors in NeRF during generation, we first generate a trigger view set dependent only on the given secret message. Then, we conduct score distillation sampling in a way that the secret message can be extracted from images rendered from arbitrary trigger viewports. To extract the secret message from the rendered image, we use a pre-trained watermark decoder from HiDDeN (Zhu et al. 2018). All NeRF generated by DREAMARK can be verified as watermarked by such a unique decoder.

Two critical evaluation metrics for watermarking algorithms are invisibility and robustness. For robustness, we evaluate bit accuracy under multiple image transformations, such as Gaussian noise, before images rendered from trigger viewports are fed into the watermark decoder. For invisibility, there is no such the "original NeRF" since we root watermarks during a generation task, so DREAMARK cannot be evaluated by Peak Signal-to-Noise Ratio (PSNR) as is done in *post-generation* methods. However, we can still evaluate the invisibility by evaluating the generation quality as previous 2D watermarking tasks (Wen et al. 2024; Yang et al. 2024), where they use CLIP Score (Radford et al. 2021) to show the generation quality. Extensive experiments show that DREAMARK successfully embeds the watermark in a *during-generation* way and maintains robustness under multiple image transformations without degrading the generation quality. In summary, our contributions are as follows:

- To the best of our knowledge, we propose DREAMARK, the first during-generation 3D watermarking method, which eliminates the delay between NeRF generation and watermarking, ensuring that no non-watermarked version of the NeRF is ever produced, thereby preventing NeRF theft.
- The key novelty of our DREAMARK is that it watermarks NeRF by injecting backdoors during score distillation sampling, such that the secret message can be extracted from images rendered from arbitrary trigger viewport.
- Extensive experiments show that the embedded watermark achieves 90+% bit accuracy against multiple image transformations, and the watermarking process does not degrade the generation quality.

Related Work

Text-to-3D Content Generation

One category of text-to-3D generation starts from DreamField (Jain et al. 2022), which trains NeRF with CLIP (Radford et al. 2021) guidance to achieve text-to-3D distillation. However, the generated content is unsatisfactory due to the weak supervision from CLIP loss. Hence, our work will not consider watermarking CLIP-guided 3D content generation. Another category starts from Dreamfusion (Poole et al. 2022), which pioneerly introduces Score Distillation Sampling (SDS) to optimize NeRF by distilling a pre-trained text-to-image diffusion model. This motivates a great number of following works to propose critical incremental. These works improve the quality of generation in various ways. For example, Fantasia3D (Chen et al. 2023), Magic3D (Lin et al. 2023), Latent-nerf (Metzer et al. 2023), DreamGaussian (Tang et al. 2023) and GaussianDreamer (Yi et al. 2023) improve the visual quality of generated content by changing 3D representations or improving NeRF structure. MVDream (Shi et al. 2023) focuses on addressing Janus problems by fine-tuning the pre-trained diffusion model to make it 3D aware. However, SDS guidance still suffers from over-saturation problems, as is shown in Magic3D (Lin et al. 2023), Dreamfusion (Poole et al. 2022), and AvatarVerse (Zhang et al. 2024). The other, like ProlificDreamer (Wang et al. 2024) and LucidDreamer (Liang et al. 2024), focus on improving SDS itself. For example, LucidDreamer uncovers the reason for the overly-smoothed problem that SDS guides the generation process towards an averaged pseudo-ground-truth and proposes ISM to relieve such a problem. ProlificDreamer proposes VSD guidance instead of SDS guidance and shows that SDS is just a special case of VSD. Although extensive research has been proposed to improve text-to-3D generation, these works still require a much longer training stage, which makes it necessary to protect the copyright of generated content.

Digital Watermarking

Digital watermarking hides watermarks into multimedia for copyright protection or leakage source tracing. Various research works have been proposed to protect traditional multimedia content like 2D images and 3D meshes. Early works watermark images and meshes by embedding a secret message in either the least significant bits (Van Schyndel, Tirkel, and Osborne 1994) or the most significant bits (Tsai 2020; Jiang et al. 2017; Tsai and Liu 2022; Peng, Liao, and Long 2022; Peng, Long, and Long 2021) of image pixels and vertex coordinates. HiDDeN (Zhu et al. 2018) and Deep3DMark (Zhu et al. 2024) have made substantial improvements using deep learning networks.

Recently, several watermarking methods have emerged in the NeRF domain. StegaNeRF (Li et al. 2023) designed a steganography algorithm that hides natural images in 3D scene representation. CopyRNeRF (Luo et al. 2023) protects the copyright of NeRF by verifying the secret message extracted from images rendered from the protected NeRF. WateNeRF (Jang et al. 2024) further improves NeRF watermarking by hiding secret messages into the frequency

domain of rendered images, increasing the robustness of the watermark. However, CopyRNeRF and WateNeRF are two *post-generation* watermarking methods, *i.e*. they watermark by fine-tuning a pre-trained NeRF. This poses a delay between the NeRF generation and watermarking. A malicious user could obtain the pre-trained NeRF before it is watermarked. Besides, CopyRNeRF requires additional changes in NeRF architecture. We would like the watermarking method to be architecture agnostic due to the fact that some text-to-3D generation methods, like Magic3D, Fantasia3D, and Latent-nerf, require specific NeRF architecture for visual quality improvement. To address these issues, we design an architecture-agnostic method that watermarks NeRF during generation.

Preliminaries

NeRF. NeRF (Mildenhall et al. 2021) uses multilayer perceptrons (MLPs) f_σ and $f_\mathbf{c}$ to map the 3D location $\mathbf{x} \in \mathbb{R}^3$ and viewing direction $\mathbf{d} \in \mathbb{R}^2$ to a color value $\mathbf{c} \in \mathbb{R}^3$ and a geometric value $\sigma \in \mathbb{R}^+$:

$$\sigma, \mathbf{z} = f_\sigma(\gamma_\mathbf{x}(\mathbf{x})), \tag{1}$$

$$\mathbf{c} = f_\mathbf{c}(\mathbf{z}, \gamma_\mathbf{d}(\mathbf{d})), \tag{2}$$

where $\gamma_\mathbf{x}, \gamma_\mathbf{d}$ are fixed encoding functions for location and viewing direction, respectively. The intermediate variable $\mathbf{z}$ is a feature output by the first MLP f_σ. To render a $H \times W$ image with the given viewport $\mathbf{p}$, the rendering process casts rays $\{r_i\}_{i=1}^{H\times W}$ from pixels and computes the weighted sum of the color $\mathbf{c}_j$ of the sampling points along each ray to composite the color of each pixel:

$$\hat{\mathbf{C}}(r_i) = \sum_j T_j(1 - \exp(-\sigma_j\delta_j))\mathbf{c}_j, \tag{3}$$

where $T_j = \prod_k^{j-1} \exp(-\sigma_k\delta_k)$, and δ_k is the distance between adjacent sample points. In later chapters, we use $\mathbf{g}(\theta, \mathbf{p}) \in [0,1]^{H\times W\times 3}$ to represent the above rendering process, where θ represents parameters of a NeRF, and $\mathbf{g}$ takes viewport $\mathbf{p}$ as input and outputs a normalized image.

Diffusion models. A diffusion model (Song et al. 2020; Ho, Jain, and Abbeel 2020; Song, Meng, and Ermon 2020) involves a forward process $\{q_t\}_{t\in[0,1]}$ to gradually add noise to a data point $\mathbf{x}_0 \sim q_0(\mathbf{x}_0)$ and a reverse process $\{p_t\}_{t\in[0,1]}$ to denoise/generate data. The forward process is defined by $q_t(\mathbf{x}_t|\mathbf{x}_0) := \mathcal{N}(\alpha_t\mathbf{x}_0, \sigma_t^2\mathbf{I})$ and $q_t(\mathbf{x}_t) := \int q_t(\mathbf{x}_t|\mathbf{x}_0)q_0(\mathbf{x}_0)d\mathbf{x}_0$, where $\alpha_t, \sigma_t > 0$ are hyperparamaters; and the reverse process is defined by denoising from $p_1(\mathbf{x}_1) := \mathcal{N}(\mathbf{0}, \mathbf{I})$ with a parameterized noise prediction network $\epsilon_\phi(\mathbf{x}_t, t)$ to predict the noise added to a clean data $\mathbf{x}_0$, which is trained by minimizing:

$$\mathcal{L}_{\text{Diff}}(\phi) = \mathbb{E}_{\mathbf{x}_0,t,\epsilon}\left[w(t)\|\epsilon_\phi(\alpha_t\mathbf{x}_0 + \sigma_t\epsilon) - \epsilon\|_2^2\right], \tag{4}$$

where $w(t)$ is a time-dependent weighting function. After training, we have $p_t \approx q_t$; thus, we can draw samples from $p_0 \approx q_0$. One of the most important applications is text-to-image generation (Rombach et al. 2022; Ramesh et al. 2022), where the noise prediction model $\epsilon_\phi(\mathbf{x}_t, t, y)$ is conditioned on a text prompt y.

Text-to-3D generation by score distillation sampling (SDS) (Poole et al. 2022). SDS is widely used in text-to-3D generation (Lin et al. 2023; Wang et al. 2024; Liang et al. 2024), which lift 2D information upto 3D NeRF by distilling pre-trained diffusion models. Given a pre-trained text-to-image diffusion model $p_t(\mathbf{x}_t|y)$ with the noise prediction network $\epsilon_\phi(\mathbf{x}_t, t, y)$, SDS optimizes a single NeRF with parameter θ. Given a camera viewport $\mathbf{p}$, a prompt y and a differentiable rendering mapping $\mathbf{g}(\theta, \mathbf{p})$, SDS optimize the NeRF θ by minimizing:

$$\mathcal{L}_{\text{SDS}}(\theta) = \mathbb{E}_{t,\mathbf{p}}\left[\frac{\sigma_t}{\alpha_t}w(t)D_{\text{KL}}(q_t^\theta(\mathbf{x}_t|c)\|p_t(\mathbf{x}_t|y^c))\right], \tag{5}$$

where $\mathbf{x}_t = \alpha_t\mathbf{g}(\theta, \mathbf{p}) + \sigma_t\epsilon$. Its gradients are approximated by:

$$\nabla_\theta\mathcal{L}_{\text{SDS}} = \mathbb{E}_{t,\mathbf{p}}\left[w(t)\left(\epsilon_\phi(\mathbf{x}_t, t, y) - \epsilon\right)\frac{\partial\mathbf{g}(\theta, \mathbf{p})}{\partial\theta}\right]. \tag{6}$$

Proposed Method

DREAMARK watermarks generation process of neural radiance fields (NeRF) by injecting backdoors during score distillation sampling (SDS). The message can be extracted from the rendered image of trigger viewports through a fixed watermark decoder. Our method is conducted in two phases. First, we pre-train the watermark decoder W_D. Then, we inject backdoors into a high-resolution NeRF during SDS optimization, such that images rendered from the trigger viewports yield a secret message.

Pre-Train the Watermark Decoder

Different from CopyRNeRF (Luo et al. 2023), which trains a separate watermark decoder for each watermarked NeRF, DREAMARK employs a unique watermark decoder. This allows the NeRFs generated by our method to be verified using this unique decoder.

Building watermark decoder training pipeline. For simplicity, we use HiDDeN (Zhu et al. 2018) as our W_D architecture, a well-established image watermarking pipeline. It optimizes watermark encoder W_E and watermark decoder W_D for signature embedding and extraction. The encoder W_E takes a cover image $x_o \in \mathbb{R}^{H\times W\times 3}$ and a k-bit message $m \in \{0,1\}^k$ as input and outputs a watermarked image $x_w \in \mathbb{R}^{H\times W\times 3}$. The decoder takes watermarked image x_w as input and outputs a predicted secret message m'. The extracted message m' is restricted to $[0,1]$ by utilizing a sigmoid function. The message loss is calculated with Binary Cross Entropy (BCE) between m and sigmoid $sg(m')$:

$$\mathcal{L}_{msg} = -\sum_{i=0}^{L-1} m_i \cdot \log sg(m'_i) + (1-m_i)\cdot\log(1-sg(m'_i)). \tag{7}$$

The W_E is discarded in the later phase since only W_D serves our purpose.

Original HiDDeN architecture combines message loss and perceptual loss to optimize both W_E and W_D. However, since W_E is discarded and the perceptual loss is not needed, we follow the tradition (Fernandez et al. 2023; Jang

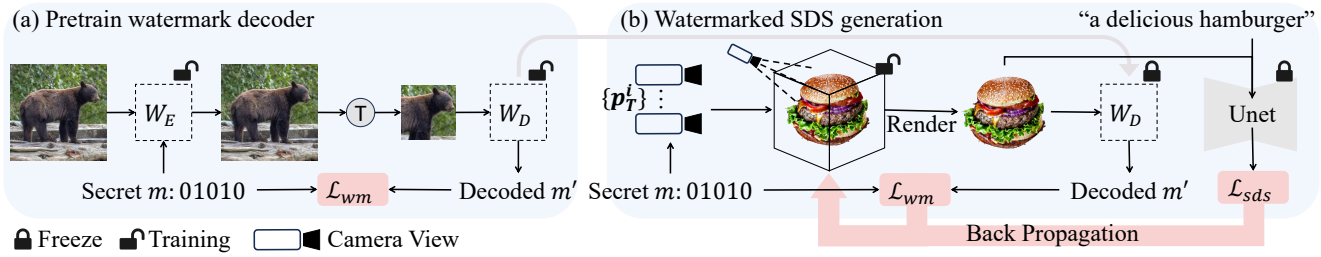

Figure 2: Overview. (a) We first pre-train a watermark encoder W_E to embed a watermark into images and a watermark decoder W_D to decode a watermark from images. (b) We generate trigger viewports $\{\mathbf{p}_T^i\}$ from the given secret message m and optimize a NeRF such that the secret message can be decoded from images rendered from arbitrary trigger viewport $\mathbf{p}_T^i$.

et al. 2024) to optimize W_E and W_D by message loss only. We find that when the decoder receives a vanilla-rendered image, there is a bias between the extracted message bits. Thus, after training the decoder, we conduct PCA whitening to a linear decoder layer to remove the bias without reducing the extraction ability.

Transformation layers. For robustness consideration, a transformation layer is added between W_E and W_D, which applies additional distortions to x_w, such as Gaussian blur, to make the decoder W_D robust to multiple attacks. During training, it takes in a watermarked image x_w produced by image encoder W_E and outputs a noised version of the watermarked image, which will be further fed to the decoder W_D. This transformation layer is made of cropping, resizing, rotation and identity. Within each iteration, one type of transformation is chosen randomly for image editing. In detail, we add random cropping with parameters 0.3 and 0.7, resizing with parameters 0.3 and 0.7 and rotation with a degree range from $-\pi/6$ to $\pi/6$.

DreaMark

Inspired from existing black-box model watermarking (Adi et al. 2018; Zhang et al. 2018; Jia et al. 2021; Le Merrer, Perez, and Trédan 2020; Chen, Rouhani, and Koushanfar 2019; Szyller et al. 2021), where they root backdoors in a deep neural network to achieve DNN watermarking, we watermark NeRF by rooting backdoors during SDS optimization. Formally, given a NeRF with parameter θ, a prompt y, a secret message m, we aim to optimize the NeRF such that the message m can be decoded by W_D from the image $\mathbf{g}(\theta, \mathbf{p}_T)$ rendered from arbitrary trigger viewport $\mathbf{p}_T$.

Generate Trigger Viewports. We wish to generate a set of trigger viewports $\{\mathbf{p}_T^i\}_{i=1}^N$ from the secret message m such that the watermark verifier does not need to keep a replica of the trigger viewport set. Besides, different messages should generate different viewports because a constant trigger viewport set is easy to predict, leading to potential vulnerability. We use a pseudo-random number generator (PRNG) to generate the m-dependent trigger viewport set $\{\mathbf{p}_T^i\}_{i=1}^N$ as shown in Algorithm 1.

Choosing Trigger Embedding Media. After generating m-dependent trigger viewport set $\{\mathbf{p}_T^i\}_{i=1}^N$, the question is how to choose suitable cover media to hide secret message m, such that m can be decoded from the image rendered from arbitrary trigger viewport $\mathbf{p}_T^i$. In the text-to-NeRF generation (Wang et al. 2024; Liang et al. 2024; Lin et al. 2023), NeRF sample a set of points and obtains their colors $\mathbf{c}$ and geometry values σ through two MLPs: $f_\sigma, f_\mathbf{c}$ (Eq. (1), (2)). For brevity, we can view the way NeRF computes point colors as $\mathbf{c} = \mathrm{MLP}(x, d)$. However, CopyRNeRF must learn a separate message feature field to get the point color $\mathbf{c} = \mathrm{MLP}(x, d, m)$. When integrating CopyRNeRF with an arbitrary text-to-NeRF pipeline, modifications to the NeRF structure are necessary to accommodate CopyRNeRF.

Algorithm 1: Trigger Viewport Generation

Input: Secret message m
Output: m-dependent Trigger viewport $\{\mathbf{p}_T^i\}_{i=1}^N$

$seed \leftarrow \mathrm{SHA256}(m)$ ▷ SHA-256 hash of the message
Initialize random generator with $seed$
$\{\mathbf{p}_T^i\}_{i=1}^N \leftarrow$ Generate N viewports with initialized generator
return $\{\mathbf{p}_T^i\}_{i=1}^N$

If we expect no additional changes to the NeRF structure, there are two options to hide triggers in the geometric mapping f_σ or the color mapping $f_\mathbf{c}$. In practice, we find that the generated NeRF has low rendering quality once we incorporate geometric mapping f_σ into backdooring. This may be because changing the geometric density of a sampled point will affect its color rendered from all viewing directions (Eq.(2)). Therefore, we decide to backdoor in color mapping $f_\mathbf{c}$.

Two-stage Trigger Embedding. To backdoor in color mapping $f_\mathbf{c}$ only, we divide SDS optimization into two stages. In the first stage, we optimize a high-resolution NeRF (*e.g.*, 512) by SDS (Eq.(5)) with joint optimization of both $f_\mathbf{c}$ and f_σ. The aim of the first stage is to generate scenes with complex geometry. In the second stage, we freeze f_σ to fine-tune $f_\mathbf{c}$ by the following combined loss to conceal watermarks in trigger viewports p_T:

$$\mathcal{L}_{\text{comb}}(\theta) = \mathcal{L}_{\text{sds}} + \mathbb{E}_{\mathbf{p}_T^i}\left[\mathrm{BCE}(W_D(\mathbf{g}(\theta, \mathbf{p}_T^i)), m)\right]. \quad (8)$$

Note that equation 8 optimizes θ by SDS across arbitrary viewports $\mathbf{p}$, and BCE across trigger viewports $\mathbf{p}_T^i$.

Watermark Extraction. Given a suspicious NeRF $\mathbf{g}(\theta, \mathbf{p})$, the NeRF creator can first generate the trigger view-

port set $\{\mathbf{p}_T^i\}_{i=1}^N$ following Algorithm 1 based on his secret message m. Then the decoded message m' can be extracted from the image rendered from arbitrary trigger viewport $\mathbf{p}_T \in \{\mathbf{p}_T^i\}_{i=1}^N$. The ownership can be verified by evaluating the bitwise accuracy between m' and m.

Implementation Details

Pretrained Watermark Extractor. We pretrain the watermark encoder W_E and extractor W_D using COCO (Lin et al. 2014) dataset. We build W_E with four-layer MLPs and W_D with eight-layer MLPs, with all intermediate channels set to 64. During pretraining, the input image resolution is set to 256×256, and the output message length is set to 48 to satisfy the capacity requirements of downstream watermarking tasks. We use Lamb (You et al. 2019) and CosineLRScheduler to schedule the learning rate, which decays to 1×10^{-6}. This process is done in 500 epochs.

NeRF. We choose Instant NGP (Müller et al. 2022) for efficient high-resolution (*e.g.*, up to 512) rendering. Given input coordinate $\mathbf{x}$, we use a 16-level progressive grid for input encoding with the coarsest and finest grid resolution set to 16 and 2048, respectively. The encoded input is further fed into $f_\mathbf{c}$ and f_σ, which are both built with one-layer MLPs with 64 channels, to predict the color c_j and density σ_j of input $\mathbf{x}$. We follow the object-centric initialization used in Magic3D (Lin et al. 2023) to initialze density for NeRF as $\sigma_{\text{init}}(\mathbf{x}) = \lambda_\sigma(1 - \frac{\|\mathbf{x}\|_2}{r})$, where $\lambda_\sigma = 10$ is the density strength, $r = 0.5$ is the density radius and $\mathbf{x}$ is the coordinate. We use Adam optimizer with learning rate 10^{-3} to optimize NeRF in both stages. The guidance model is Stable Diffusion (Rombach et al. 2022) with the guidance scale set to 100. During SDS optimization, we sample time $t \sim \mathcal{U}(0.02, 0.98)$ in each iteration. We jointly optimize $f_\mathbf{c}$ and f_σ for 40000 iterations in stage one and fine-tune $f_\mathbf{c}$ only for 30000 iterations in stage two.

Experiment

Experiment Setup

We select 16-bit secret messages and $N = 1000$ trigger viewports in our experiment unless explicitly mentioned. To evaluate DREAMARK, we use 100 different prompts to generate 100 watermarked scenes. Note that the scale of our experiment far exceeds that of prior works where they only evaluate Blender (Mildenhall et al. 2021) and LLFF (Mildenhall et al. 2019) dataset, with each dataset only containing eight scenes. All our experiments are conducted on Ubuntu 22.04 with an Intel Xeon Gold 5318Y CPU and an NVIDIA A100.

Evaluation Metrics. Two key evaluations for watermarking algorithms are invisibility and robustness. We evaluate robustness using bit accuracy under various image distortions such as Gaussian Noise, Rotation, Scaling, Gaussian Blur, Crop, and Brightness. For invisibility evaluation, different from the previous *post-generation* watermarking algorithm, there is no such the "original NeRF". Hence, the typical evaluation metric, the Peak-Signal-to-Noise Ratio (PSNR), is not applicable to evaluate our method. We follow prior 2D *during-generation* watermarking algorithm (Yang et al. 2024; Wen et al. 2024) where they use CLIP-Score (Radford et al. 2021) to evaluate the bias introduced by the watermarking algorithm.

Method	Bit Acc(%)			
	8 bits	16 bits	32 bits	48 bits
Post Generation				
SDS+CopyRNeRF	100.0	91.16	78.08	60.06
SDS+WateRF	100.0	94.24	86.81	70.43
During Generation				
DREAMARK	100.0	98.93	82.59	71.91

Table 1: Bit Acc(%) under different bit length settings.

Method	Bit Acc	CLIP/16	CLIP/32
None	N/A	0.3156	0.2859
Post Generation			
SDS+CopyRNeRF	91.16	0.3152	0.2831
SDS+WateRF	94.24	0.3164	0.2823
During Generation			
DREAMARK	98.93	0.3218	0.2943

Table 2: Bit Acc(%) and CLIP Score comparison with post-generation methods. "None" reports the performance when no watermark is applied, so bit accuracy is not applicable.

Baselines. Since DREAMARK is the first *during-generation* watermarking method to watermark the generated NeRF, and existing NeRF watermarking CopyRNeRF (Luo et al. 2023) and WateNeRF (Jang et al. 2024) can only embed watermark after NeRF generation. We select CopyRNeRF and WateNeRF as two *post-generation* baselines, *i.e.* we first generate NeRF with SDS only, then watermark the generated NeRF with CopyRNeRF and WateNeRF.

Performance of Dreamark

Capacity. We evaluate bit accuracy across 8, 16, 32, and 48-bit secret messages. Table 1 shows that all watermarking methods have a trade-off between bit accuracy and capacity. As a during-generation watermarking method, DREAMARK shows relatively high accuracy on 8, 16, and 48-bit settings compared to post-generation watermarking methods, such as SDS+CopyRNeRF and SDS+WateRF. For example, in 16-bit settings DREAMARK achieves 98.93% accuracy while SDS+CopyRNeRF and SDS+WateRF achieve 91.16% and 94.24% accuracy, respectively.

Generation quality. We report CLIP/16 evaluated by *clip-ViT-B-16* and CLIP/32 evaluated by *clip-ViT-B-32* to indicate the generation quality of the watermarked images. For each scene, its CLIP-Score is averaged among all viewports $\mathbf{p}$, and bit accuracy is averaged among all N trigger viewport $\{\mathbf{p}_T^i\}_{i=1}^N$. All prior watermarking works (Jang et al. 2024; Luo et al. 2023; Zhu et al. 2024, 2018) show a trade-off between bit accuracy and watermarked content quality. However, Table 2 shows DREAMARK achieves superior performance in both bit accuracy and generation qual-

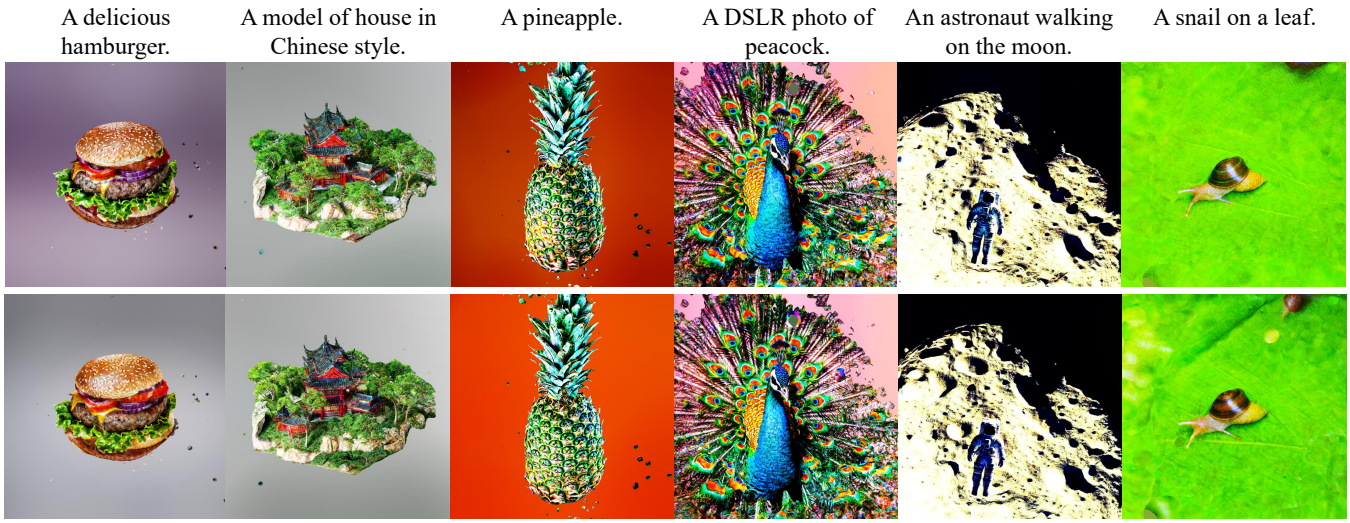

Figure 3: Images rendered from trigger viewports. Top: generated non-watermarked NeRF. Bottom: Watermarked NeRF generated by DREAMARK. We aim to show that generated NeRF has the same visual quality as the non-watermarked NeRF instead of showing they are perceptually the same since there is no such the "original NeRF" in the generation context.

	GN.	Rot.	Sca.	Crop
Without T	0.64	0.57	0.56	0.89
With T	0.93	0.84	0.92	0.91

Table 3: Bit accuracy by removing transformation layer T.

	CLIP/16	CLIP/32	Bit Acc
Both $f_{\mathbf{c}}$ and f_{σ}	0.2308	0.2241	60.43%
$f_{\mathbf{c}}$ only	0.3218	0.2943	98.93%

Table 4: Watermark by optimizing both $f_{\mathbf{c}}$ and f_{σ} or $f_{\mathbf{c}}$ only.

ity when compared to post-generation methods. Notably, the CLIP Score is 0.3156/0.2859 when no watermark is applied, while DREAMARK achieves 0.3218/0.2943 CLIP Score. This indicates that during-generation watermark embedding of DREAMARK does not harm the generation quality. We also provide visual results of generated NeRF in Figure 3.

Size of trigger viewports. We evaluate bit accuracy under different numbers of trigger sizes N. As is shown in Figure 4, we surprisingly find that bit accuracy is not significantly affected by trigger size. For example, bit accuracy reaches 99.88% when $N = 50$ and only drops to 96.27% when N increases to 20000.

Ablation Study

Can we remove the transformation layer T during watermark decoder W_D pre-training? Table 3 shows that the robustness significantly drops when the transformation layer is removed.

Can we hide watermarks by jointly optimizing color mapping $f_{\mathbf{c}}$ and geometric mapping f_{σ}? Table 4 shows that watermark, by optimizing both color mapping and geometric mapping, has lower performance on both bit accuracy and CLIP Score. This may be because, within one iteration, only one single direction of point color is supervised, while changing its point density will affect its color in all directions, significantly increasing the difficulty of convergence.

Attacks on Dreamark's Watermarks

This section aims to examine the robustness of the watermark against various attacks. We first consider image-level attack, which performs arbitrary image transformations and is typical for many NeRF watermarking methods (Luo et al. 2023; Jang et al. 2024). We then consider model-level attacks such as fine-tuning and pruning since the generated NeRF could be made public; in this case, the attacker will have white-box access to the generated NeRF. Besides, model-level attacks are commonly evaluated in model watermarking methods (Adi et al. 2018; Zhang et al. 2018; Jia et al. 2021; Le Merrer, Perez, and Trédan 2020; Chen, Rouhani, and Koushanfar 2019; Szyller et al. 2021).

Robustness Against Image-Level Attacks

We evaluate the robustness of the watermark against different image transformations before rendered images are fed into the watermark decoder. We consider Gaussian Noise (v=0.1), rotation ($\pm\pi/6$), Scaling (25%), Gaussian Blur (deviation=0.1), Crop (40%) and Brightness (2.0). Bit accuracy is averaged on all transformed images rendered from trigger viewports. Table 5 shows DREAMARK is robust against previously mentioned image-level attacks. The bit accuracy is always above 90% except for rotation. Note that the robustness is achieved without the need for transformations during the DREAMARK optimization phase: it is attributed to the watermark decoder. If the watermark decoder is trained to withstand arbitrary transformation, the generated NeRF

	Bit Accuracy (%)						
	No Distortion	Gaussian Noise (v=0.1)	Rotation ($\pm\pi/6$)	Scaling (25%)	Gaussian Blur (deviation=0.1)	Crop (40%)	Brightness (2.0)
Post Generation							
SDS+CopyRNeRF	91.16%	90.04%	88.13%	89.33%	90.06%	N/A	N/A
SDS+WateRF	94.24%	94.06%	85.02%	91.35%	94.12%	95.40%	90.91%
During Generation							
DREAMARK	98.93%	93.75%	84.51%	92.40%	98.93%	91.49%	91.23%

Table 5: Robustness under multiple image transformations compared with post-generation-based methods.

subsequently learns to contain watermarks that maintain robustness throughout the DREAMARK optimization.

Robustness Against Model-Level Attacks

This subsection considers the scenario when an attacker gets full access to the generated NeRF model and aims to remove the embedded watermark without degrading its visual quality. We denote x_w as images rendered from watermarked NeRF and x_a as images rendered from attacked NeRF. We use $\text{PSNR}(x_w, x_a) = -10 \cdot \log_{10}(\text{MSE}(x_w, x_a))$, for $x_a, x_w \in [0,1]^{c\times h\times w}$ to evaluate distortion made by attacks.

Model Fine-tuning. Since our method uses a unified watermark decoder W_D to decode the secret message, we consider two scenarios of fine-tuning attack. One assumes that the attacker has full access to the watermark decoder W_D, and the other assumes that the attacker has no access to W_D. Besides, we assume the attacker has no prior knowledge of the secret message m. In this case, the attacker cannot reproduce trigger viewports since trigger viewports are only related to m. For the first scenario, when the attacker has full access to W_D, the attacker can use an adversarial attack to partially remove the watermark by minimizing the BCE loss between the extracted message and a random binary message sampled beforehand:

$$\mathcal{L}_{\text{fine-tune}}(\theta') = \mathbb{E}_{\mathbf{p}}[\|\mathbf{g}(\theta', \mathbf{p}) - \mathbf{g}(\theta, \mathbf{p})\|_2^2 + \text{BCE}(W_D(\mathbf{g}(\theta', \mathbf{p})), m')], \tag{9}$$

where θ is the fixed parameter of watermarked NeRF, θ' is the parameter of NeRF to be fine-tuned, m' is a random binary message different from m. As shown in Fig. 5, even if PSNR drops below 26dB, bit accuracy is still above 90%. For the second scenario, when the attacker has no access to W_D, the attacker cannot produce the adversarial attack to remove the watermark.

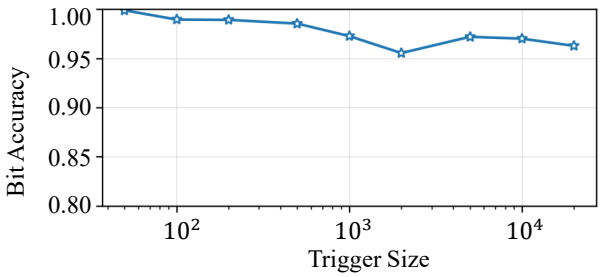

Figure 4: Effect under varied trigger size. Bit accuracy is not significantly affected by trigger size.

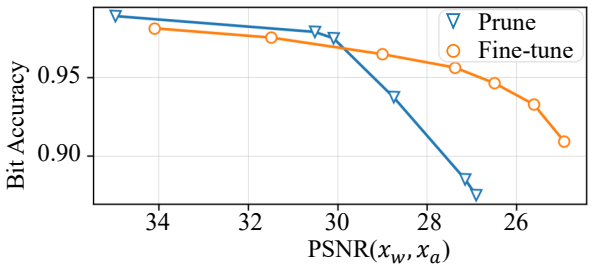

Figure 5: Robustness against model-level attacks. x_w, x_a are images rendered from watermarked and attacked NeRF.

Model Pruning. Model pruning is widely used in model compression since it can reduce the storage and computation cost of DNNs. However, pruning will affect not only the size and operation speed of the model but also the accuracy of the watermark and the visual quality of NeRF. A higher pruning rate gives lower watermark accuracy and lower visual quality. In practice, we vary the pruning rate and, at the same time, evaluate $\text{PSNR}(x_a, x_w)$ and bit accuracy on x_a. The pruning attack with pruning rate $a\%$ means setting the smallest $a\%$ of network parameters to zero, where the size of the network parameter is evaluated by its ℓ_1 norm. Fig. 5 shows that our method is robust against pruning attack since we still have $\sim$88% accuracy when PSNR is below 27dB, while 27dB PSNR means relatively high distortion has been made in image watermarking context (Zhu et al. 2018).

Conclusion

In this work, we propose a *during-generation* text-to-3D watermarking method, DREAMARK, which eliminates the delay between the generation phase and the watermarking phase: the watermark can be verified on the generated NeRF once the generation is finished. Inspired by the black-box model watermarking method, DREAMARK watermarks NeRF by injecting backdoors into NeRF such that a secret message can be extracted from images rendered from arbitrary trigger viewport. Extensive experiments show that our method will not degrade generation quality and maintain robustness against image-level and model-level attacks.

Acknowledgments

This work was supported in part by National Key R&D Program of China under Grant 2021YFF0900300, in part by Guangdong Key Program under Grant 2021QN02X166, and in part by Research Institute of Trustworthy Autonomous Systems under Grant C211153201. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding parties.

References

Adi, Y.; Baum, C.; Cisse, M.; Pinkas, B.; and Keshet, J. 2018. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th USENIX security symposium (USENIX Security 18)*, 1615–1631.

Chen, H.; Rouhani, B. D.; and Koushanfar, F. 2019. Blackmarks: Blackbox multibit watermarking for deep neural networks. *arXiv preprint arXiv:1904.00344*.

Chen, R.; Chen, Y.; Jiao, N.; and Jia, K. 2023. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In *Proceedings of the IEEE/CVF international conference on computer vision*, 22246–22256.

Fernandez, P.; Couairon, G.; Jégou, H.; Douze, M.; and Furon, T. 2023. The stable signature: Rooting watermarks in latent diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 22466–22477.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33: 6840–6851.

Jain, A.; Mildenhall, B.; Barron, J. T.; Abbeel, P.; and Poole, B. 2022. Zero-shot text-guided object generation with dream fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 867–876.

Jang, Y.; Lee, D. I.; Jang, M.; Kim, J. W.; Yang, F.; and Kim, S. 2024. WateRF: Robust Watermarks in Radiance Fields for Protection of Copyrights. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12087–12097.

Jia, H.; Choquette-Choo, C. A.; Chandrasekaran, V.; and Papernot, N. 2021. Entangled watermarks as a defense against model extraction. In *30th USENIX security symposium (USENIX Security 21)*, 1937–1954.

Jiang, R.; Zhou, H.; Zhang, W.; and Yu, N. 2017. Reversible data hiding in encrypted three-dimensional mesh models. *IEEE Transactions on Multimedia*, 20(1): 55–67.

Le Merrer, E.; Perez, P.; and Trédan, G. 2020. Adversarial frontier stitching for remote neural network watermarking. *Neural Computing and Applications*, 32(13): 9233–9244.

Li, C.; Feng, B. Y.; Fan, Z.; Pan, P.; and Wang, Z. 2023. Steganerf: Embedding invisible information within neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 441–453.

Liang, Y.; Yang, X.; Lin, J.; Li, H.; Xu, X.; and Chen, Y. 2024. Luciddreamer: Towards high-fidelity text-to-3d generation via interval score matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6517–6526.

Lin, C.-H.; Gao, J.; Tang, L.; Takikawa, T.; Zeng, X.; Huang, X.; Kreis, K.; Fidler, S.; Liu, M.-Y.; and Lin, T.-Y. 2023. Magic3d: High-resolution text-to-3d content creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 300–309.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, 740–755. Springer.

Luo, Z.; Guo, Q.; Cheung, K. C.; See, S.; and Wan, R. 2023. Copyrnerf: Protecting the copyright of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 22401–22411.

Metzer, G.; Richardson, E.; Patashnik, O.; Giryes, R.; and Cohen-Or, D. 2023. Latent-nerf for shape-guided generation of 3d shapes and textures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12663–12673.

Mildenhall, B.; Srinivasan, P. P.; Ortiz-Cayon, R.; Kalantari, N. K.; Ramamoorthi, R.; Ng, R.; and Kar, A. 2019. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (ToG)*, 38(4): 1–14.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1): 99–106.

Müller, T.; Evans, A.; Schied, C.; and Keller, A. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4): 1–15.

Peng, F.; Liao, T.; and Long, M. 2022. A semi-fragile reversible watermarking for authenticating 3D models in dual domains based on variable direction double modulation. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(12): 8394–8408.

Peng, F.; Long, B.; and Long, M. 2021. A general region nesting-based semi-fragile reversible watermarking for authenticating 3D mesh models. *IEEE transactions on circuits and systems for video technology*, 31(11): 4538–4553.

Poole, B.; Jain, A.; Barron, J. T.; and Mildenhall, B. 2022. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, 8748–8763. PMLR.

Ramesh, A.; Dhariwal, P.; Nichol, A.; Chu, C.; and Chen, M. 2022. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2): 3.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-resolution image synthesis with latent

diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10684–10695.

Shi, Y.; Wang, P.; Ye, J.; Long, M.; Li, K.; and Yang, X. 2023. Mvdream: Multi-view diffusion for 3d generation. *arXiv preprint arXiv:2308.16512*.

Song, J.; Meng, C.; and Ermon, S. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.

Song, Y.; Sohl-Dickstein, J.; Kingma, D. P.; Kumar, A.; Ermon, S.; and Poole, B. 2020. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*.

Szyller, S.; Atli, B. G.; Marchal, S.; and Asokan, N. 2021. Dawn: Dynamic adversarial watermarking of neural networks. In *Proceedings of the 29th ACM International Conference on Multimedia*, 4417–4425.

Tang, J.; Ren, J.; Zhou, H.; Liu, Z.; and Zeng, G. 2023. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv preprint arXiv:2309.16653*.

Tsai, Y.-Y. 2020. Separable reversible data hiding for encrypted three-dimensional models based on spatial subdivision and space encoding. *IEEE transactions on multimedia*, 23: 2286–2296.

Tsai, Y.-Y.; and Liu, H.-L. 2022. Integrating coordinate transformation and random sampling into high-capacity reversible data hiding in encrypted polygonal models. *IEEE Transactions on Dependable and Secure Computing*, 20(4): 3508–3519.

Van Schyndel, R. G.; Tirkel, A. Z.; and Osborne, C. F. 1994. A digital watermark. In *Proceedings of 1st international conference on image processing*, volume 2, 86–90. IEEE.

Wang, Z.; Lu, C.; Wang, Y.; Bao, F.; Li, C.; Su, H.; and Zhu, J. 2024. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *Advances in Neural Information Processing Systems*, 36.

Wen, Y.; Kirchenbauer, J.; Geiping, J.; and Goldstein, T. 2024. Tree-rings watermarks: Invisible fingerprints for diffusion images. *Advances in Neural Information Processing Systems*, 36.

Yang, Z.; Zeng, K.; Chen, K.; Fang, H.; Zhang, W.; and Yu, N. 2024. Gaussian Shading: Provable Performance-Lossless Image Watermarking for Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12162–12171.

Yi, T.; Fang, J.; Wu, G.; Xie, L.; Zhang, X.; Liu, W.; Tian, Q.; and Wang, X. 2023. Gaussiandreamer: Fast generation from text to 3d gaussian splatting with point cloud priors. *arXiv preprint arXiv:2310.08529*.

You, Y.; Li, J.; Reddi, S.; Hseu, J.; Kumar, S.; Bhojanapalli, S.; Song, X.; Demmel, J.; Keutzer, K.; and Hsieh, C.-J. 2019. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*.

Zhang, H.; Chen, B.; Yang, H.; Qu, L.; Wang, X.; Chen, L.; Long, C.; Zhu, F.; Du, D.; and Zheng, M. 2024. Avatarverse: High-quality & stable 3d avatar creation from text and pose. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 7124–7132.

Zhang, J.; Gu, Z.; Jang, J.; Wu, H.; Stoecklin, M. P.; Huang, H.; and Molloy, I. 2018. Protecting intellectual property of deep neural networks with watermarking. In *Proceedings of the 2018 on Asia conference on computer and communications security*, 159–172.

Zhu, J.; Kaplan, R.; Johnson, J.; and Fei-Fei, L. 2018. Hidden: Hiding data with deep networks. In *Proceedings of the European conference on computer vision (ECCV)*, 657–672.

Zhu, X.; Ye, G.; Luo, X.; and Wei, X. 2024. Rethinking Mesh Watermark: Towards Highly Robust and Adaptable Deep 3D Mesh Watermarking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 7784–7792.