GAUSSIAN CARRIER: MODEL-LEVEL WATERMARKING IN FEED-FORWARD 3D GAUSSIAN SPLATTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Feed-forward 3D Gaussian Splatting (3DGS) models are increasingly deployed as reusable pretrained reconstruction engines, yet their editable parameters introduce an underexplored model-level risk: an attacker can encode an attacker-specified invisible message into novel view renderings through lightweight fine-tuning of the Gaussian prediction head, without perturbing the input images or retraining the full model. We investigate this model-level attack surface through adversarial watermark injection. Naively fine-tuning the Gaussian prediction head, however, modifies predicted primitives at all spatial locations, resulting in unnecessary representation changes and degrading the rendering fidelity. To address this, we propose *Gaussian Carrier*, which identifies reliable watermark carriers according to opacity and cross-view observability. We develop a carrier-restricted mechanism that applies the adapted predictions only to selected primitives while retaining the original pretrained predictions. Combined with complementary watermark-success and fidelity-preserving objectives, the predictions are optimized. Experiments on RealEstate10K and ACID datasets with two feed-forward 3DGS models achieve over 99% watermarking accuracy while limiting the PSNR difference to at most 0.31 dB. The watermarking approach also maintains above 80% accuracy under various distortions. These results expose Gaussian adaptation as a model-level attack surface in pretrained feed-forward 3DGS.

1 INTRODUCTION

3D Gaussian Splatting (3DGS) has become a widely adopted representation for photorealistic novel view synthesis by combining explicit scene primitives with real-time rendering (Kerbl et al., 2023). Recent feed-forward approaches further eliminate costly per-scene reconstruction by predicting a complete Gaussian representation from only a few input images (Charatan et al., 2024; Chen et al., 2024a; Xu et al., 2025; Ye et al., 2025a; Zhang et al., 2026a; Long et al., 2026). This transition turns 3D reconstruction models into reusable pretrained engines that can be distributed and applied to previously unseen scenes. However, the editable parameters of these models introduce a new attack surface: an attacker can alter Gaussian generation behavior through model adaptation, without perturbing the input images or retraining the full reconstruction network.

A practical model-level attack should preserve the visual appearance of the original reconstruction while silently encoding an attacker-specified message. Based on this requirement, invisible watermarking has been extensively studied for identifying generated content and protecting 3D assets. Existing 3DGS methods commonly embed ownership information by optimizing an already reconstructed Gaussian scene or modifying particular Gaussian attributes (Chen et al., 2025b; Jang et al., 2025; Tan et al., 2024; Huang et al., 2024; Li et al., 2025a; Huang et al., 2025; Li et al., 2025b). Despite their progress, these approaches predominantly consider an owner-side setting in which watermarking is intentionally incorporated for copyright protection. They do not examine whether an adversary can exploit a small parameter subset of an already pretrained feed-forward 3DGS model to inject a hidden message.

This raises a fundamental security question: *can an attacker implant an invisible watermark into the feed-forward Gaussian generation process through lightweight parameter adaptation while preserving reconstruction fidelity?* Inspired by Stable Signature (Fernandez et al., 2023), which roots watermarks in latent diffusion models through decoder fine-tuning, we investigate this possibility

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

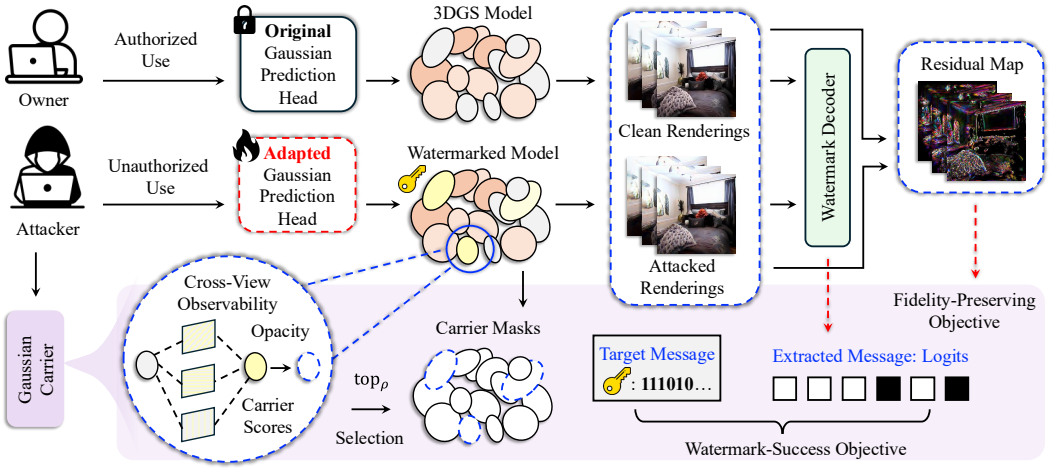


Figure 1: Overview of the proposed method. In the attack setting, the Gaussian prediction head is adapted under unauthorized use, and our *Gaussian Carrier* identifies watermark-carrying primitives according to carrier scores, forming carrier masks that restrict adapted predictions to selected Gaussians. Watermark-success and fidelity-preserving objectives jointly guide the adaptation, enabling reliable watermarking while preserving rendering fidelity.

in feed-forward 3DGS. We consider an attacker with access to the pretrained model who adapts only the Gaussian prediction head, while keeping the input views and all remaining model components frozen. Unlike an image decoder, however, the Gaussian prediction head jointly predicts attributes that control primitive geometry and appearance. Naively fine-tuning it therefore modifies all predicted Gaussians, including primitives with low opacity or limited cross-view observability that provide weak support for watermark recovery. Such unnecessary modifications can degrade reconstruction fidelity and destabilize message recovery across views.

To address this, we propose *Gaussian Carrier*, a model-level invisible watermarking attack against pretrained feed-forward 3DGS. Our key observation is that Gaussian primitives are not equally suitable for carrying an injected message: reliable carriers should be sufficiently opaque and consistently observable across views. As shown in Figure 1, we rank the original Gaussian primitives according to cross-view observability and opacity, and select a subset as carriers. We then apply the adapted predictions only to these selected primitives through masked mixing, while preserving the original predictions elsewhere. This carrier-restricted fine-tuning concentrates representation changes and direct watermark gradients on selected primitives. Combined with watermark-success and fidelity-preserving objectives, our method enables stable watermarking while maintaining rendering quality close to the original reconstruction. Our results reveal lightweight Gaussian head adaptation as an underexplored model-integrity risk in pretrained feed-forward 3DGS.

Our contributions are threefold:

- To our knowledge, we present the first model-level invisible watermarking attack in pretrained feed-forward 3DGS, requiring adaptation of only the Gaussian prediction head.
- We propose *Gaussian Carrier*, which selects reliable primitives based on cross-view observability and opacity and restricts adapted predictions to the selected carriers.
- We validate Gaussian Carrier on two feed-forward 3DGS architectures and two datasets, achieving high watermark recovery and robustness to common distortions.

2 Related Work

3DGS Watermarking. The growing adoption of 3DGS has motivated increasing attention to its security and provenance (Lou et al., 2025; Chen et al., 2026c; Hong et al., 2025; Zhang et al., 2025b; Meng et al., 2025). More broadly, watermarking has been extensively studied for content

authentication and intellectual-property protection, including robust and efficient image watermarking (Fang et al., 2025; Qiu et al., 2025; 2026), watermarking for generative models (Yang et al., 2024; 2026), and deep watermarking of explicit 3D assets (Wang et al., 2022). For radiance-field representations, NeRF Signature (Luo et al., 2025) embeds model-level signatures into NeRF, while MantleMark (Luo et al., 2026) migrates watermarks from multi-view images into reconstructed NeRF and 3DGS representations. 3DGS-specific methods such as GuardSplat (Chen et al., 2025b), 3D-GSW (Jang et al., 2025), Water-GS (Tan et al., 2024), and GaussianMarker (Huang et al., 2024) primarily embed ownership information into Gaussian representations through training, fine-tuning, or scene-specific optimization. Recent works further improve embedding efficiency and generalization: GS-Marker (Li et al., 2025a) supports single-pass watermarking, MarkSplatter (Huang et al., 2025) enables arbitrary message embedding through a Splatter Image representation, and W2M (Li et al., 2026a) adaptively determines where and how strongly watermarks should be embedded.

Feed-Forward 3DGS. 3D Gaussian Splatting (3DGS) has emerged as an efficient representation for high-quality novel view synthesis (Jiang et al., 2025; Chen et al., 2024b; Liu et al., 2025b; Huang & Mikolajczyk, 2025; Zhao et al., 2026; Chen et al., 2026b; Ye et al., 2025b;a). While conventional 3DGS methods rely on scene-specific optimization to recover Gaussian representations (Kerbl et al., 2023; Lu et al., 2024; Chen et al., 2025a; Ren et al., 2026; Yu et al., 2024), recent research has increasingly shifted toward feed-forward reconstruction, where Gaussian scenes are directly predicted from sparse observations without per-scene optimization (Zhang et al., 2025a; 2026b;a; Jeong et al., 2026; Chen et al., 2026a; Li et al., 2026b). Representative methods such as pixelSplat (Charatan et al., 2024) and MVSplat (Chen et al., 2024a) demonstrate that pixel-aligned Gaussian prediction can achieve competitive rendering quality while substantially reducing reconstruction time. Subsequent approaches further improve generalization and applicability by incorporating geometric priors, relaxing camera-pose requirements, or redesigning the reconstruction pipeline, as exemplified by DepthSplat (Xu et al., 2025), PF3plat (Hong et al., 2024), MonoSplat (Liu et al., 2025a), SelfSplat (Kang et al., 2025), and IDESplat (Long et al., 2026).

Motivation and Positioning of Our Method. Despite these advances, existing 3DGS watermarking approaches predominantly focus on owner-side watermarking for provenance protection, whereas we study an attacker-side model-integrity threat that injects an invisible message by adapting pretrained feed-forward 3DGS models. As these models become increasingly reusable across scenes, their model parameters also constitute an important yet underexplored security surface, motivating the study of attacks that alter Gaussian generation behavior through lightweight model adaptation, revealing a distinct attack surface that is not considered by existing owner-oriented 3DGS watermarking approaches.

3 METHODOLOGY

3.1 PRELIMINARY

Given I sparse context views, a feed-forward 3DGS model ingests $\mathcal{X} = \{(X_i,\, P_i)\}_{i=1}^{I}$ as input, where $X_i \in \mathbb{R}^{H\times W\times 3}$ denotes the i-th context image and P_i contains the corresponding camera intrinsics, rotation, and translation. The model learns a mapping $\mathcal{E}_\theta$ from the posed context images to a set of 3D Gaussian parameters:

$$\mathcal{E}_\theta : \{(X_i,\, P_i)\}_{i=1}^{I} \mapsto \{\mathcal{G} = (\mu_j,\, \alpha_j,\, \Sigma_j,\, c_j)\}_{j=1}^{H\times W\times I}, \tag{1}$$

where θ denotes the model parameters learned during training. For each pixel-aligned Gaussian, the network predicts its position μ_j, opacity α_j, covariance Σ_j, and color c_j. At test time, the pretrained parameters θ are frozen. Given the context views $\mathcal{X}$, the model directly infers the Gaussian scene $\mathcal{G}$ without per-scene optimization. A differentiable renderer $\mathcal{R}$ then rasterizes $\mathcal{G}$ from a set of target cameras $\mathcal{P} = \{P_k\}_{k=1}^{K}$ to novel views $\mathcal{Y} = \{Y_k\}_{k=1}^{K}$. Accordingly, the clean feed-forward inference pipeline is formulated as:

$$\mathcal{G} = \mathcal{E}_\theta(\mathcal{X}) = \mathcal{E}_\theta\left(\{(X_i,\, P_i)\}_{i=1}^{I}\right), \quad Y_k = \mathcal{R}(\mathcal{G},\, P_k), \tag{2}$$

where K novel views are rendered from the Gaussian scene inferred from I context views.

3.2 Problem Formulation

We consider access to a pretrained feed-forward 3DGS model $\mathcal{E}_\theta$, where the attacker is allowed to perform **lightweight model adaptation** while keeping **the input context views unchanged** at test time. We formalize the attack through two definitions: Definition 1 introduces the parameter subset available to the attacker, and Definition 2 specifies the attack success criterion.

Definition 1 (**Model-Level Attack Scope**). *Let θ be the parameters of a feed-forward 3DGS, where θ_h is adapted in the Gaussian prediction head, and θ_f is frozen in the remaining modules. The attacker can modify only the head from θ_h to θ_h^{adv}. Given clean context views $\mathcal{X}$, the frozen modules produce intermediate features t, depth d, and opacity α, whereas the adapted module predicts the remaining Gaussian attributes, mapping intermediate features to the attributes of Gaussian primitives:*

$$\mathcal{E}_{\theta_f}(\mathcal{X}) \to (t,\, d,\, \alpha), \quad \mathcal{E}_{\theta_h \to \theta_h^{adv}}(t) \to (\Delta p^{adv},\, \Sigma^{adv},\, c^{adv}). \tag{3}$$

Here, Δp^{adv} denotes the image-plane offset to obtain the Gaussian mean, while Σ^{adv} and c^{adv} denote covariance and color attributes, respectively. The adapted model produces an attacked Gaussian representation, rendering the attacked novel views $\mathcal{Y}^{adv} = \{Y_k^{adv}\}_{k=1}^K$:

$$\mathcal{G}^{adv} = \mathcal{E}_{\{\theta_f,\, \theta_h^{adv}\}}(\mathcal{X}) = \mathcal{E}_{\{\theta_f,\, \theta_h^{adv}\}}\left(\{(X_i,\, P_i)\}_{i=1}^I\right), \quad Y_k^{adv} = \mathcal{R}(\mathcal{G}^{adv},\, P_k). \tag{4}$$

Definition 2 (**Model-Level Attack Success Criterion**). *Given a binary message bit $\in \{0,1\}^M$ as a target invisible watermark, a frozen watermark decoder $\mathcal{W}$ maps the k-th attacked rendering:*

$$z_k = \mathcal{W}\big(\mathrm{norm}(Y_k^{adv})\big) \in \mathbb{R}^M, \tag{5}$$

where z_k is an M-dimensional logit vector and norm *denotes normalization. A model-level watermark attack is considered successful if the novel view renderings Y_k^{adv} satisfy two requirements: 1) the invisible watermark of Y_k^{adv} recovered by $\mathcal{W}$ approximates bit, and 2) Y_k^{adv} remains visually close to the corresponding clean rendering Y_k generated by the original pretrained model:*

$$\mathcal{W}\big(\mathrm{norm}(Y_k^{adv})\big) \approx \textit{bit} \quad \textit{and} \quad Y_k^{adv} \approx Y_k, \quad k = 1, \ldots, K. \tag{6}$$

Therefore, the adapted Gaussian prediction head encodes the target watermark and generates watermarked Gaussian representations. This fine-tuning adaptation substantially reduces the attack scope, thereby restricting the attack to a lightweight submodule rather than modifying the full reconstruction pipeline inside the feed-forward 3DGS.

Accordingly, the goal of the attack is to encode a target invisible watermark while preserving the rendering fidelity. However, a straightforward approach does not guarantee this trade-off: unrestricted fine-tuning of the Gaussian prediction head distributes watermark-induced changes across primitives that contribute little to stable watermark recovery. We address this limitation by proposing Gaussian Carrier, identifying a subset of suitable primitives and restricting the adaptation as follows.

3.3 Gaussian Carrier

Different Gaussian primitives contribute unevenly to the novel view rendering, making unrestricted fine-tuning of the Gaussian prediction head unnecessarily aggressive. Primitives with low opacity or limited cross-view observability provide weak support for multi-view watermark decoding, yet modifying them can still introduce avoidable rendering distortion. To address this, we introduce an observability-aware **Carrier Selection** strategy that identifies stable and repeatedly observable Gaussians based on cross-view coverage and opacity. We then employ **Restricted Fine-tuning**, where the adapted head predictions are applied only to the selected carriers, while unselected primitives retain the outputs of the original frozen head. This localizes watermark-induced representation changes and gradients to more reliable carriers. Finally, we develop complementary **Optimization Objectives** on the selected carriers, encouraging consistent message recovery while maintaining rendering quality.

Carrier Selection. We construct the carrier mask from the Gaussian representation $\mathcal{G}$ predicted by the original pretrained model in Eq. (1), where μ_j and α_j denote the position and opacity of the j-th Gaussian, respectively. For each target camera $P_k \in \mathcal{P}$, we define a binary observability indicator:

$$q_{k,j} = \mathbf{1}\left[\mu_j \text{ lies in front of } P_k,\ d_{k,j} \geq 0,\ \mathrm{u}_{k,j} \in [0,\, 1]^2\right], \tag{7}$$

Algorithm 1: Overall Optimization of Gaussian Carrier

Input: Context views $\mathcal{X}$, Pretrained feed-forward 3DGS $\mathcal{E}_\theta$, Target message bit, Decoder $\mathcal{W}$
Output: Adapted Gaussian prediction head θ_h^{adv}, Attacked renderings $\{Y_k^{adv}\}_{k=1}^K$
Predict the original Gaussian $\mathcal{G}$ from $\mathcal{X}$ and render original views $\{Y_k\}_{k=1}^K$ in Eq. (2)
Identify eligible carriers by scores r_j in Eq. (8) and construct masks $\mathcal{M}$ in Eq. (9)
Decompose model parameters $\theta = \{\theta_f, \theta_h\}$ in $\mathcal{E}_\theta$, freeze θ_f, and initialize $\theta_h^{adv} \leftarrow \theta_h$
for step $= 1$ to overall fine-tuning steps **do**
    Extract intermediate feature t to predict original $\mathcal{E}_{\theta_h}(t)$ and adapted $\mathcal{E}_{\theta_h^{adv}}(t)$ in Eq. (3)
    Initialize unrestricted attributes $\mathcal{E}^* = \mathcal{E}_{\theta_h^{adv}}(t)$
    if Carrier-Restricted Fine-tuning **then**
        Construct carrier-restricted attributes $\mathcal{E}^* = \mathcal{M} \odot \mathcal{E}_{\theta_h^{adv}}(t) + (1 - \mathcal{M}) \odot \mathcal{E}_{\theta_h}(t)$ in Eq. (10)
    end if
    Construct attacked Gaussian representation $\mathcal{G}^{adv}$ and render novel views Y_k^{adv} in Eq. (4)
    Extract watermark logits z_k and evaluate attack success through bit in Definition 2
    Compute $\mathcal{L} = \mathcal{L}_{bit} + \mathcal{L}_{sp}$ with $\mathcal{L}_{bit}$ in Eq. (11) and $\mathcal{L}_{sp}$ in Eq. (12)
    if Carrier-Restricted Fine-tuning **then**
        $\mathcal{L} = \mathcal{L} + \mathcal{L}_{margin}$ with $\mathcal{L}_{margin}$ in Eq. (11)
    end if
    Update only the θ_h^{adv} using Adam with $\nabla_{\theta_h^{adv}} \mathcal{L}$
end for
return θ_h^{adv} and $\{Y_k^{adv}\}_{k=1}^K$

where $d_{k,j}$ and $\mathrm{u}_{k,j}$ denote the depth and normalized coordinate of Gaussian j under target camera P_k. The value of $q_{k,j}$ as 1 indicates that Gaussian j projects to a valid location in the k-th target view, verifying that the Gaussian lies in front of the camera and projects inside the image boundary.

We measure cross-view observability n_j of each Gaussian by counting the number of novel views from which it is observable. To jointly consider their potential rendering influence, we define the carrier score r_j as:

$$r_j = \alpha_j \frac{n_j}{K} \quad \text{with} \quad n_j = \sum_{k=1}^{K} q_{k,j}, \tag{8}$$

where a larger r_j favors primitives with higher opacity and broader cross-view coverage, making them more reliable candidates for watermark carrying. To identify primitives eligible for carrier selection, we set σ and τ as the minimum observability and opacity requirements, respectively.

Carrier-Restricted Fine-Tuning. Among eligible Gaussians, we select the top_ρ fraction of indices ranked in descending order by r_j as watermark carriers, where $\rho \in (0, 1]$ is the carrier ratio. We accordingly define binary carrier masks m_j as:

$$m_j = \mathbf{1}[j \in \mathcal{S}], \quad \mathcal{S} = \mathrm{top}_\rho(\mathcal{C}; r_j), \quad \mathcal{C} = \{j \mid n_j \geq \sigma,\ \alpha_j \geq \tau\}, \tag{9}$$

where m_j is in a complete set $\mathcal{M} = \{m_j\}_{j=1}^{H \times W \times I}$.

Given the intermediate feature t as defined in Eq. (3), let $\mathcal{E}_{\theta_h}(t)$ and $\mathcal{E}_{\theta_h^{adv}}(t)$ denote the Gaussian attributes predicted by the original and adapted Gaussian prediction heads, respectively. Using the carrier mask $\mathcal{M}$ obtained above, we construct the Gaussian attributes for rendering as:

$$\mathcal{E}^* = \mathcal{M} \odot \mathcal{E}_{\theta_h^{adv}}(t) + (1 - \mathcal{M}) \odot \mathcal{E}_{\theta_h}(t), \tag{10}$$

where selected carriers with a mask of 1 use the adapted prediction, and unselected primitives with a mask of 0 retain the original pretrained prediction. The mask is broadcast along the attribute dimension and is treated as a non-differentiable selection during adaptation. This carrier-restricted fine-tuning localizes watermark-induced representation changes and direct gradient contributions to the selected carriers, while preserving the remaining Gaussian attributes. Detailed gradient analyses are explained in Appendix A.

Optimization Objectives We first leverage a multi-view binary cross-entropy (bce) objective to encourage all K rendered views to recover the target message. Since the attack is realized through

model-level fine-tuning, the gradients are propagated and update only the trainable Gaussian prediction head. Yet, this objective continues to increase the confidence of bits even after they have already been decoded correctly. Under lightweight model adaptation, such redundant updates may unnecessarily enlarge the deviation of the adapted Gaussian parameters and degrade rendering fidelity. To alleviate this issue, we further introduce a bit-level margin objective that focuses optimization on insufficiently confident bits. Let $z_{k,m}$ be the m-th component of the logit vector z_k in Eq. (5), and let $bit_m \in \{0, 1\}$ denote the corresponding target bit for m within M. We define the signed logit as $s_{k,m} = z_{k,m}(2bit_m - 1)$, where a larger $s_{k,m}$ indicates stronger agreement between the predicted and target bits. Thus, the watermark-success objectives are defined as:

$$\mathcal{L}_{bit} = \frac{1}{K}\sum_{k=1}^{K}\left(\mathrm{bce}(z_k,\, bit)\right), \quad \mathcal{L}_{margin} = \lambda\frac{1}{KM}\sum_{k=1}^{K}\sum_{m=1}^{M}\max(\kappa - s_{k,m},\, 0), \tag{11}$$

where λ controls the contribution of the objective and κ specifies the desired confidence margin. Once $s_{k,m}$ reaches κ, the corresponding bit no longer contributes to $\mathcal{L}_{margin}$. In this way, the bce objective drives global message recovery, whereas the bit-level margin objective prevents excessive optimization on correctly recovered bits, thereby aligning more naturally with the Gaussian-head fine-tuning paradigm.

Reliable watermark decoding alone is insufficient for a successful model-level attack, since adapting the Gaussian prediction head introduces visible deviations in the rendered novel views. We therefore constrain the attack from the renderings, encouraging the adapted model to carry the target message while remaining close to the original pretrained model. Specifically, we introduce a fidelity-preserving objective that penalizes both pixel-level and low-frequency discrepancies between the attacked rendering Y_k^{adv} and its clean counterpart Y_k at the same target view:

$$\mathcal{L}_{sp} = \frac{1}{K}\sum_{k=1}^{K}\left(||Y_k^{adv} - Y_k||_1 + \left\|\phi\big(Y_k^{adv}\big) - \phi(Y_k)\right\|_1\right), \tag{12}$$

where ϕ denotes a low-pass operator implemented using average pooling. The first term directly suppresses pixel-wise rendering deviations, while the second preserves the low-frequency structure and overall appearance of the scene. The overall optimizations of unrestricted fine-tuning and carrier-restricted fine-tuning methods are shown in Algorithm 1. By concentrating fine-tuning on sufficiently opaque and repeatedly observable primitives, Gaussian Carrier reduces unnecessary modification of weakly contributing Gaussians and improves the balanced trade-off.

4 Experiments

Datasets & Metrics. We use RealEstate10K (Zhou et al., 2018) and ACID (Liu et al., 2021) as datasets, containing indoor and outdoor scenes. We use PSNR, SSIM, and LPIPS as quality metrics. For watermark recovery, Bit Accuracy (Acc) is averaged over rendered target views and scenes.

Baselines & Implementation Details. We conduct the evaluation on two pretrained models: MVSplat (Chen et al., 2024a) and IDESplat (Long et al., 2026). We compare three configurations: 1) *Original Reconstruction* uses the pretrained model without watermarking and quantifies both the reconstruction ceiling and the chance-level response of the watermark decoder; 2) *Unrestricted Fine-tuning* denotes our initial attempt with unrestricted updates of the Gaussian prediction head with basic objectives; 3) *Carrier-Restricted Fine-tuning* is our full method with carrier selection and a bit-level margin objective. In addition, we use the frozen 48-bit decoder (Fernandez et al., 2023) to extract an invisible watermark. Following the feed-forward reconstruction protocol, each scene provides two clean context views and three target views. The optimization step is 150 with learning rate as 10^{-4}. We set the observability requirement σ and opacity threshold τ as 2 and 0.05, with a carrier ratio ρ of 0.7. The weight λ is 20 with a bit margin κ of 0, and average pooling with a stride of 8 of the low-pass operator.

4.1 Main Results

Watermark Recovery and Rendering Fidelity. As shown in Table 1, unrestricted fine-tuning substantially improves watermark recovery over the original feed-forward models, but at the cost of

Table 1: Main results on RealEstate10K and ACID. We evaluate our method with bit accuracy (Bit Acc ↑) and rendering quality (PSNR ↑, SSIM ↑, and LPIPS ↓). GP denotes the Gaussian prediction.

Comparison Methods	Updated Module	RealEstate10K				ACID			
		Bit Acc	PSNR	SSIM	LPIPS	Bit Acc	PSNR	SSIM	LPIPS
MVSplat (Chen et al., 2024a)	None	51.82	26.36	0.868	0.129	51.37	28.21	0.842	0.145
Unrestricted Fine-tuning	GP Head	87.07	25.56	0.845	0.170	79.10	27.84	0.835	0.163
Carrier-Restricted Fine-tuning	GP Head	99.88	26.20	0.864	0.137	99.63	28.00	0.837	0.158
IDESplat (Long et al., 2026)	None	52.37	27.84	0.893	0.108	51.90	29.09	0.859	0.130
Unrestricted Fine-tuning	GP Head	89.08	27.29	0.880	0.131	88.47	28.34	0.840	0.164
Carrier-Restricted Fine-tuning	GP Head	99.91	27.66	0.888	0.116	99.89	28.78	0.851	0.146

Table 2: Robustness to image distortions on RealEstate10K and ACID. We evaluate bit accuracy (Bit Acc ↑) under carrier-restricted fine-tuning, where No Dist. means no image distortion.

Carrier-Restricted Fine-tuning	RealEstate10K (Zhou et al., 2018)					ACID (Liu et al., 2021)				
	No Dist.	Noise $\mu = 0.1$	Scaling 75%	Blur $\sigma = 0.1$	Brightness $0.5 \sim 1.5$	No Dist.	Noise $\mu = 0.1$	Scaling 75%	Blur $\sigma = 0.1$	Brightness $0.5 \sim 1.5$
ours + MVSplat	99.88	93.23	82.70	99.14	89.85	99.63	89.79	80.37	98.23	86.91
ours + IDESplat	99.91	92.93	81.66	99.43	89.56	99.89	92.77	80.82	99.76	89.60

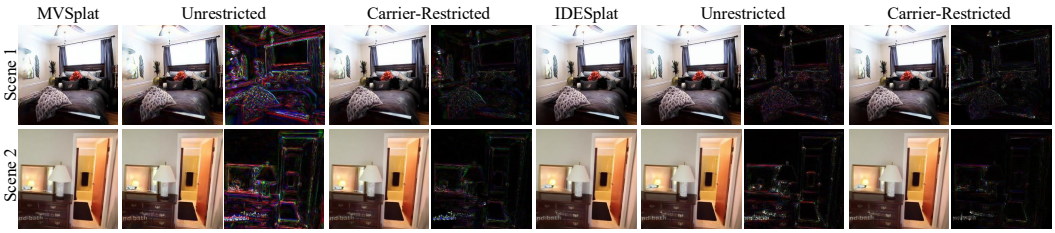


Figure 2: Visual comparison on RealEstate10K. We compare the original MVSplat and IDESplat renderings, unrestricted fine-tuning, and carrier-restricted fine-tuning. Pixel-wise residual maps are $\times 10$ for better view, where our carrier-restricted fine-tuning exhibits weaker residual responses.

noticeable rendering degradation. In contrast, carrier-restricted fine-tuning consistently achieves near-perfect bit accuracy on both RealEstate10K and ACID while preserving rendering quality much closer to the original models. For MVSplat, our method improves Bit Acc to 99.88% and 99.63% on RealEstate10K and ACID, respectively, while maintaining PSNR/SSIM/LPIPS close to the clean baseline. Similar trends are observed on IDESplat, where carrier-restricted fine-tuning reaches 99.91% and 99.89% Bit Acc with substantially smaller fidelity degradation than unrestricted fine-tuning. These results demonstrate that restricting adaptation to selected carriers provides a more favorable watermark–fidelity trade-off across different feed-forward 3DGS architectures and datasets.

Robustness to Image Distortions. We further evaluate the robustness of the injected watermark under common image distortions, including additive noise, scaling, blur, and brightness adjustment. As reported in Table 2, the watermark maintains high decoding accuracy under additive noise, blur, and brightness variations on both RealEstate10K and ACID. Among the evaluated transformations, scaling causes degradation, indicating that geometric resizing is more challenging for the current watermark representation. Nevertheless, the watermark remains reliably recoverable across a broad range of practical distortions, demonstrating robustness to common distortions. These results show that the injected watermark remains recoverable after various operations, rather than relying on exact pixel values of the original rendered views.

Visualization Results. Figure 2 presents qualitative results on RealEstate10K. Across different scenes, unrestricted fine-tuning introduces noticeable changes throughout the rendered images, which are clearly reflected by the stronger responses in the magnified residual maps. In contrast, our carrier-restricted fine-tuning method preserves the visual appearance of the original reconstruction

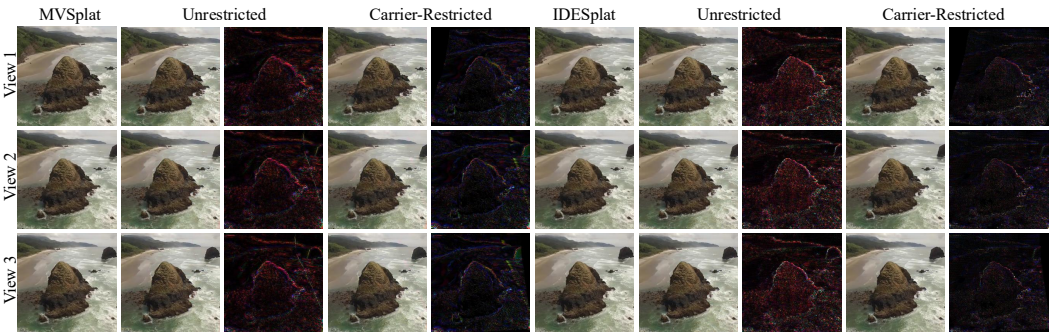


Figure 3: Visual comparison on ACID dataset. We visualize multiple novel views from the same scene. For each backbone, we compare unrestricted and carrier-restricted fine-tunings, and report the corresponding pixel-wise residual maps. Our carrier-restricted fine-tuning method consistently produces smaller visual deviations from the original renderings across different views, indicating improved fidelity.

Table 3: Effectiveness of the design components. We evaluate observability-guided carrier selection and the bit-level margin objective on top of the same basic unrestricted fine-tuning baseline.

Method	$\theta_h \to \theta_h^{adv}$	$\mathcal{L}_{sp}$	$\mathcal{L}_{bit}$	$\mathcal{L}_{margin}$	$\mathcal{M}$	Bit Acc ↑	PSNR ↑	SSIM ↑	LPIPS ↓
Unrestricted Fine-tuning	✓	✓	✓	✗	✗	87.07	25.56	0.845	0.170
+ Bit-level Margin Objective	✓	✓	✓	✓	✗	97.26	26.04	0.859	0.145
+ Carrier Selection	✓	✓	✓	✗	✓	94.35	26.11	0.860	0.143
Carrier-restricted Fine-tuning	✓	✓	✓	✓	✓	99.88	26.20	0.864	0.137

more effectively, with only minor localized differences. This trend is consistently observed for both MVSplat and IDESplat, demonstrating that our carrier-restricted fine-tuning can maintain rendering fidelity across diverse scenes while embedding the target watermark. Figure 3 further evaluates rendering consistency across multiple novel views of the same scene from ACID dataset. Although unrestricted fine-tuning preserves the overall scene structure, its residual maps reveal widespread appearance changes, where the carrier-restricted fine-tuning substantially suppresses these deviations and produces consistently weaker residual responses. The similar observations suggest that the selected carriers support watermark embedding while maintaining stable rendering quality.

4.2 Ablation Studies

Ablation studies are conducted on MVSplat with RealEstate10K for controlled comparisons. The same default hyperparameters are used across all datasets and backbones, and we further verified the selected settings on IDESplat and ACID, observing consistent trends. Additional analyses of more ablations, qualitative results, and computational complexity are provided in the Appendix B.

Effectiveness of Model Design. As shown in Table 3, both the bit-level margin objective and carrier selection contribute consistently to the performance of carrier-restricted fine-tuning. The margin objective primarily improves watermark recovery, increasing Bit Acc from 87.07% to 97.26%, while carrier selection improves both recovery and rendering fidelity, reaching 94.35% Bit Acc with a PSNR of 26.11. Combining the two components yields the strongest result, with 99.88% Bit Acc, 26.20 PSNR, 0.864 SSIM, and 0.137 LPIPS. This confirms that the two components play complementary roles: the margin objective promotes reliable bit decoding, while carrier selection limits adaptation to stable Gaussian primitives and preserves reconstruction quality.

Effectiveness of Opacity Threshold, Carrier Ratio, and Observability Requirement. We investigate the sensitivity of our method to the opacity threshold τ, which controls the eligibility of Gaussians for carrier selection. As shown in Figure 4, the reconstruction quality is highly stable with marginal variations. Meanwhile, the Bit Accuracy first increases as the threshold grows and reaches its maximum at $\tau = 0.05$, after which it slightly decreases. In particular, this value achieves the highest PSNR and Bit Acc, while its SSIM and LPIPS remain nearly indistinguishable from the best values. This indicates that a moderate opacity threshold effectively filters out less reliable

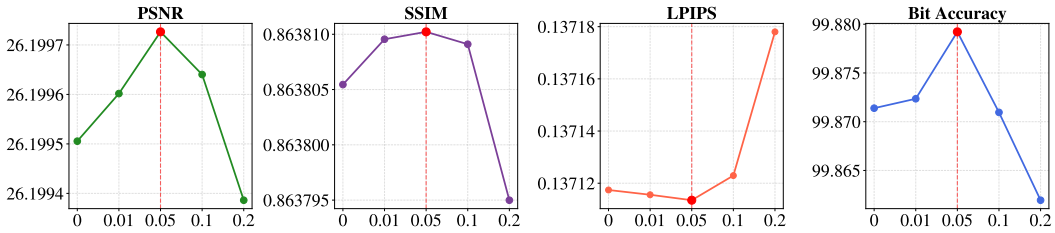


Figure 4: Ablation on the opacity threshold. We report PSNR ↑, SSIM ↑, LPIPS ↓, and Bit Acc ↑.

Table 4: Ablation on the carrier ratio and observability requirement.

ρ	0.1	0.2	0.3	0.4	0.6	0.7	0.8	0.9	1.0	σ	1	2	3
Bit Acc ↑	98.91	99.57	99.74	99.81	99.86	99.88	99.87	99.87	99.87	Bit Acc ↑	99.85	99.88	99.82
PSNR ↑	25.95	26.09	26.15	26.18	26.20	26.20	26.19	26.19	26.19	PSNR ↑	26.19	26.20	26.18
SSIM ↑	0.859	0.861	0.863	0.863	0.864	0.864	0.864	0.864	0.864	SSIM ↑	0.864	0.864	0.863
LPIPS ↓	0.151	0.144	0.140	0.137	0.137	0.137	0.138	0.138	0.138	LPIPS ↓	0.137	0.137	0.138

Table 5: Ablation on loss weight and bit margin.

Weight	1	10	20	30	40	Margin	0	0.5	1.0	1.5	2.0	2.5	3.0
Bit Acc ↑	99.06	99.78	99.88	99.88	99.88	Bit Acc ↑	99.88	96.39	92.71	91.63	91.34	91.25	91.23
PSNR ↑	26.14	26.22	26.20	26.16	26.13	PSNR ↑	26.20	20.77	20.10	20.00	19.99	19.99	19.99
SSIM ↑	0.861	0.864	0.864	0.863	0.863	SSIM ↑	0.864	0.738	0.724	0.722	0.721	0.721	0.721
LPIPS ↓	0.142	0.137	0.137	0.139	0.140	LPIPS ↓	0.137	0.334	0.352	0.356	0.356	0.356	0.356

low-opacity Gaussians without overly restricting the available carriers. Moreover, Table 4 evaluates carrier ratio ρ and minimum observability requirement σ, where only a subset of well-selected Gaussian primitives is sufficient to carry the target watermark. We study the effect of the carrier ratio ρ, which controls the fraction of eligible Gaussians selected for watermark carrying. As ρ increases from 0.1 to 0.7, Bit Acc further improves from 98.91% to 99.88%, while the rendering quality also gradually recovers, with PSNR increasing from 25.95 to 26.20 and LPIPS decreasing from 0.151 to 0.137. This behavior suggests that increasing the carrier ratio distributes the watermark information over more reliable primitives. Beyond 0.7, the performance becomes nearly saturated, with Bit Acc remaining around 99.87%. These results indicate that our method does not require all eligible primitives to achieve reliable decoding, and a moderate carrier ratio provides an effective balance.

Effectiveness of Loss Weight and Bit Margin. Table 5 analyzes the effects of the loss weight λ and bit margin κ. Increasing the loss weight substantially improves Bit Acc from 99.06% to 99.88%, while rendering quality remains relatively stable. We therefore set the loss weight to 20, which reaches the saturated Bit Acc with a favorable fidelity trade-off. For the bit margin, larger values consistently reduce Bit Acc and noticeably degrade rendering quality. We set the margin to 0, which provides the best overall watermark recovery and rendering fidelity. Notably, this value does not disable the margin objective; instead, it reduces the loss to a sign-correctness hinge that penalizes only incorrectly decoded bits and stops updating a bit once its logit crosses the decision boundary.

5 Conclusion

In this work, we investigate an underexplored model-level attack surface in pretrained feed-forward 3DGS and show that adaptation of the Gaussian prediction head can be exploited to inject invisible messages into novel view renderings. To improve the watermark–fidelity trade-off, we introduce *Gaussian Carrier*, which selects reliable Gaussian primitives according to cross-view observability and opacity, and restricts adapted predictions to these carriers. Together with watermark-success and fidelity-preserving objectives, the proposed method enables message recovery while maintaining rendering quality close to the original reconstruction. Experimental results highlight the security risks introduced by editable pretrained 3DGS models and motivate further study of model-integrity protection for feed-forward 3DGS systems.

AI USE STATEMENT

Generative AI tools were used to assist with code implementation, feedback on experimental design and analysis, literature search, and the polishing of manuscript text. The research problem and core method were developed by the authors. All AI-assisted code was reviewed and tested, and all reported results, claims, citations, and figures were manually verified by the authors. Generative AI was not used to generate experimental datasets or determine the final scientific conclusions. We take responsibility for the final content of this work, including text, claims, code, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies a model-level security vulnerability in pretrained feed-forward 3DGS systems. The proposed attack could potentially be misused to modify reconstruction models without authorization; our purpose is to expose this underexplored attack surface and motivate model-integrity verification and defense mechanisms. Experiments are conducted only on publicly available research datasets and pretrained models, without collection of private data or interaction with human participants. We report the methodology for research purposes and encourage responsible use of these findings in the development and evaluation of secure 3D reconstruction systems.

REPRODUCIBILITY STATEMENT

We provide the methodological and experimental details necessary for reproducing our results in the main paper and appendix. These include descriptions of the proposed method, experimental settings, datasets, evaluation protocols, and relevant implementation details.

REFERENCES

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*, pp. 19457–19467, 2024.

Tianyu Chen, Wei Xiang, Kang Han, Yu Lu, Di Wu, Gaowen Liu, and Ramana Rao Kompella. Gifsplat: Generative prior-guided iterative feed-forward 3d gaussian splatting from sparse views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26031–26040, 2026a.

Yang Chen, Shuai Shen, Yueqi Duan, and Yap-Peng Tan. Towards memory-based temporal coherence in pose-free 3d gaussian splatting. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 10067–10071. IEEE, 2026b.

Yang Chen, Yi Yu, Jiaming He, Yueqi Duan, Zheng Zhu, and Yap-Peng Tan. Spectral defense against resource-targeting attack in 3d gaussian splatting. *arXiv preprint arXiv:2603.12796*, 2026c.

Youyu Chen, Junjun Jiang, Kui Jiang, Xiao Tang, Zhihao Li, Xianming Liu, and Yinyu Nie. Dashgaussian: Optimizing 3d gaussian splatting in 200 seconds. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 11146–11155, 2025a.

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*, pp. 370–386, 2024a. doi: 10.1007/978-3-031-72664-4_21.

Zequn Chen, Jiezhi Yang, and Heng Yang. Pref3r: Pose-free feed-forward 3d gaussian splatting from variable-length image sequence. *arXiv preprint arXiv:2411.16877*, 2024b.

Zixuan Chen, Guangcong Wang, Jiahao Zhu, Jianhuang Lai, and Xiaohua Xie. Guardsplat: Efficient and robust watermarking for 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16325–16335, 2025b.

Han Fang, Kejiang Chen, Zehua Ma, Jiajun Deng, Yicong Li, Weiming Zhang, and Ee-Chien Chang. Syntag: Enhancing the geometric robustness of inversion-based generative image watermarking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15416–15425, 2025.

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*, pp. 22466–22477, 2023.

Jiaxin Hong, Sixu Chen, Shuoyang Sun, Hongyao Yu, Hao Fang, Yuqi Tan, Bin Chen, Shuhan Qi, and Jiawei Li. Gausstrap: Stealthy poisoning attacks on 3d gaussian splatting for targeted scene confusion. *arXiv preprint arXiv:2504.20829*, 2025.

Sunghwan Hong, Jaewoo Jung, Heeseong Shin, Jisang Han, Jiaolong Yang, Chong Luo, and Seungryong Kim. Pf3plat: Pose-free feed-forward 3d gaussian splatting. *arXiv preprint arXiv:2410.22128*, 2024.

Ranran Huang and Krystian Mikolajczyk. No pose at all: Self-supervised pose-free 3d gaussian splatting from sparse views. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 27947–27957, 2025.

Xiufeng Huang, Ruiqi Li, Yiu-ming Cheung, Ka Chun Cheung, Simon See, and Renjie Wan. Gaussianmarker: Uncertainty-aware copyright protection of 3d gaussian splatting. *Advances in Neural Information Processing Systems*, 37:33037–33060, 2024.

Xiufeng Huang, Ziyuan Luo, Qi Song, Ruofei Wang, and Renjie Wan. Marksplatter: Generalizable watermarking for 3d gaussian splatting model via splatter image structure. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 12189–12198, 2025. doi: 10.1145/3746027.3758144.

Youngdong Jang, Hyunje Park, Feng Yang, Heeju Ko, Euijin Choo, and Sangpil Kim. 3d-gsw: 3d gaussian splatting for robust watermarking. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5938–5948, 2025.

Moonyeon Jeong, Seunggi Min, Suhyeon Lee, and Hongje Seong. Viewsplat: View-adaptive dynamic gaussian splatting for feed-forward synthesis. *arXiv preprint arXiv:2603.25265*, 2026.

Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, et al. Anysplat: Feed-forward 3d gaussian splatting from unconstrained views. *ACM Transactions on Graphics*, 44(6):1–16, 2025.

Gyeongjin Kang, Jisang Yoo, Jihyeon Park, Seungtae Nam, Hyeonsoo Im, Sangheon Shin, Sangpil Kim, and Eunbyung Park. Selfsplat: Pose-free and 3d prior-free generalizable 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22012–22022, 2025.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, George Drettakis, et al. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):139–1, 2023.

Lijiang Li, Jinglu Wang, Xiang Ming, and Yan Lu. Gs-marker: Generalizable and robust watermarking for 3d gaussian splatting. *arXiv preprint arXiv:2503.18718*, 2025a.

Nan Li, Yike Zeng, Qian Zhang, Qi Zhang, Zhiyi Pan, Wei Feng, and Liang Wan. Write where it matters: Policy-guided watermarks for 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6580–6590, 2026a.

Runyi Li, Xuanyu Zhang, Chuhan Tong, Zhipei Xu, and Jian Zhang. Gaussianseal: Robust high-fidelity watermark for 3d gaussian splatting. *arXiv preprint arXiv:2503.00531*, 2025b.

Yihui Li, Chengxin Lv, Zichen Tang, Hongyu Yang, and Di Huang. Tokensplat: Token-aligned 3d gaussian splatting for feed-forward pose-free reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 40886–40895, 2026b.

Andrew Liu, Richard Tucker, Varun Jampani, Ameesh Makadia, Noah Snavely, and Angjoo Kanazawa. Infinite nature: Perpetual view generation of natural scenes from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14458–14467, 2021.

Yifan Liu, Keyu Fan, Weihao Yu, Chenxin Li, Hao Lu, and Yixuan Yuan. Monosplat: Generalizable 3d gaussian splatting from monocular depth foundation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21570–21579, 2025a.

Yifan Liu, Shengjun Zhang, Chensheng Dai, Yang Chen, Hao Liu, Chen Li, and Yueqi Duan. Learning efficient and generalizable human representation with human gaussian model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11797–11806, 2025b.

Wei Long, Haifeng Wu, Shiyin Jiang, Jinhua Zhang, Xinchun Ji, and Shuhang Gu. Idesplat: Iterative depth probability estimation for generalizable 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Tianrui Lou, Xiaojun Jia, Siyuan Liang, Jiawei Liang, Ming Zhang, Yanjun Xiao, and Xiaochun Cao. 3d gaussian splatting driven multi-view robust physical adversarial camouflage generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 28752–28762, 2025.

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 Computer Vision and Pattern Recognition Conference*, pp. 20654–20664, 2024.

Ziyuan Luo, Anderson Rocha, Boxin Shi, Qing Guo, Haoliang Li, and Renjie Wan. The nerf signature: Codebook-aided watermarking for neural radiance fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(6):4652–4667, 2025. doi: 10.1109/TPAMI.2025.3550166.

Ziyuan Luo, Jun Liu, Haoliang Li, Anderson Rocha, and Renjie Wan. Mantlemark: Migrating watermarks from multi-view images to radiance fields via frequency modulation. *IEEE Transactions on Information Forensics and Security*, 21:256–271, 2026. doi: 10.1109/TIFS.2025.3636020.

Lingzhuang Meng, Mingwen Shao, Yuanjian Qiao, Wenjie Liu, and Xiang Lv. Gaussian splitting attack: Gaussian splatting-based multi-view 3d adversarial attack. *Pattern Recognition*, pp. 112466, 2025.

Yupeng Qiu, Han Fang, and Ee-Chien Chang. Lightweight-mark: Rethinking deep learning-based watermarking. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 50480–50501. PMLR, 2025.

Yupeng Qiu, Han Fang, and Ee-Chien Chang. Casial: Geometric distortion robust image watermarking. *arXiv preprint arXiv:2607.26729*, 2026.

Shiwei Ren, Tianci Wen, Yongchun Fang, and Biao Lu. Fastgs: Training 3d gaussian splatting in 100 seconds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26094–26103, 2026.

Yuqi Tan, Xiang Liu, Shuzhao Xie, Bin Chen, Shu-Tao Xia, and Zhi Wang. Water-gs: Toward copyright protection for 3d gaussian splatting via universal watermarking. *arXiv preprint arXiv:2412.05695*, 2024.

Feng Wang, Hang Zhou, Han Fang, Weiming Zhang, and Nenghai Yu. Deep 3d mesh watermarking with self-adaptive robustness. *Cybersecurity*, 5(1):24, 2022. doi: 10.1186/s42400-022-00125-w.

Haofei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. Depthsplat: Connecting gaussian splatting and depth. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 16453–16463, 2025.

Jindong Yang, Han Fang, Weiming Zhang, Nenghai Yu, and Kejiang Chen. Fari: Robust one-step inversion for watermarking in diffusion models. In *International Conference on Learning Representations*, 2026.

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*, pp. 12162–12171, 2024. doi: 10.1109/CVPR52733.2024.01156.

Botao Ye, Boqi Chen, Haofei Xu, Daniel Barath, and Marc Pollefeys. Yonosplat: You only need one model for feedforward 3d gaussian splatting. *arXiv preprint arXiv:2511.07321*, 2025a.

Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3d gaussian splats from sparse unposed images. In *International Conference on Learning Representations*, volume 2025, pp. 54009–54033, 2025b.

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*, pp. 19447–19456, 2024.

Shangzhan Zhang, Jianyuan Wang, Yinghao Xu, Nan Xue, Christian Rupprecht, Xiaowei Zhou, Yujun Shen, and Gordon Wetzstein. Flare: Feed-forward geometry, appearance and camera estimation from uncalibrated sparse views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21936–21947, 2025a.

Xiaoxue Zhang, Xiaoxu Zheng, Yixuan Yin, Tiao Zhao, Kaihua Tang, Michael Bi Mi, Zhan Xu, and Dave Zhenyu Chen. Anchorsplat: Feed-forward 3d gaussian splatting with 3d geometric priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18924–18933, 2026a.

Xuanyu Zhang, Jiarui Meng, Zhipei Xu, Shuzhou Yang, Yanmin Wu, Ronggang Wang, and Jian Zhang. Securegs: Boosting the security and fidelity of 3d gaussian splatting steganography. *arXiv preprint arXiv:2503.06118*, 2025b.

Zicheng Zhang, Xiangting Meng, Ke Wu, and Wenchao Ding. Sparsesplat: Towards applicable feed-forward 3d gaussian splatting with pixel-unaligned prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5049–5058, 2026b.

Jia-Chen Zhao, Beiqi Chen, Xinyang Chen, Guangcong Wang, and Liqiang Nie. Structsplat: Generalizable 3d gaussian splatting from uncalibrated sparse views. *arXiv preprint arXiv:2606.28321*, 2026.

Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Transactions on Graphics*, 37(4), 2018.

A Appendix-Additional Discussions

A.1 Gradient Routing

We provide additional details on how carrier selection controls the forward and backward paths of the Gaussian prediction head. Given the intermediate feature t, let $\mathcal{E}_{\theta_h}(t)$ and $\mathcal{E}_{\theta_h^{adv}}(t)$ denote the Gaussian attributes predicted by the original and adapted heads, respectively. The three adaptation regimes can be written in a unified form as:

$$\mathcal{E}^* = \begin{cases} \mathcal{E}_{\theta_h}(t), & \text{Frozen}, \\ \mathcal{E}_{\theta_h^{adv}}(t), & \text{Unrestricted fine-tuning}, \\ \mathcal{M} \odot \mathcal{E}_{\theta_h^{adv}}(t) + (1 - \mathcal{M}) \odot \mathcal{E}_{\theta_h}(t), & \text{Carrier-restricted fine-tuning}. \end{cases} \tag{13}$$

The frozen setting corresponds to the original pretrained model without adaptation. Unrestricted fine-tuning applies the adapted prediction at every Gaussian location, whereas carrier-restricted fine-tuning applies it only to selected carriers. The binary carrier mask $\mathcal{M}$ is broadcast along the Gaussian attribute dimension. Unselected locations therefore retain the original pretrained predictions.

Both $\mathcal{M}$ and $\mathcal{E}_{\theta_h}(t)$ are treated as fixed during adaptation. Consequently, the gradient support with respect to the adapted head prediction is:

$$\frac{\partial \mathcal{E}^*}{\partial \mathcal{E}_{\theta_h^{adv}}(t)} = \begin{cases} 0, & \text{Frozen}, \\ 1, & \text{Unrestricted fine-tuning}, \\ \mathcal{M}, & \text{Carrier-restricted fine-tuning}. \end{cases} \tag{14}$$

Accordingly, the gradient of the optimization objective with respect to the adapted Gaussian-head parameters follows:

$$\nabla_{\theta_h^{adv}} \mathcal{L} = \frac{\partial \mathcal{L}}{\partial \mathcal{E}^*} \frac{\partial \mathcal{E}^*}{\partial \mathcal{E}_{\theta_h^{adv}}(t)} \frac{\partial \mathcal{E}_{\theta_h^{adv}}(t)}{\partial \theta_h^{adv}}. \tag{15}$$

Thus, unrestricted fine-tuning receives direct gradient contributions from all Gaussian locations, whereas carrier-restricted fine-tuning restricts them to the selected carriers. Although the parameters θ_h^{adv} are spatially shared, unselected locations are replaced by the frozen-head prediction before Gaussian construction and therefore do not directly contribute gradients under the masked forward pass. Depth and opacity remain unchanged because they are produced by separate frozen branches.

When $m_j = 1 \in \mathcal{M}$ for all Gaussian locations, Eq. (13) reduces to unrestricted fine-tuning. Hence, carrier-restricted fine-tuning changes only the spatial support of the adapted prediction while retaining the same trainable Gaussian prediction head.

A.2 Detailed Adaptation

We initialize the adapted Gaussian prediction head from the pretrained parameters as:

$$\theta_h^{adv} \leftarrow \theta_h, \tag{16}$$

and optimize only θ_h^{adv} while keeping all remaining components of the feed-forward 3DGS pipeline frozen. The adapted head is optimized with Adam using the watermark-success and fidelity-preserving objectives defined in the main paper.

The clean reference renderings, carrier mask $\mathcal{M}$, and original Gaussian-head predictions are computed before adaptation and remain fixed throughout optimization. At each iteration, the original clean context images are used as input, and the losses are averaged over the target views. Gradients are propagated only to θ_h^{adv}, and no image-space perturbations are required.

After adaptation, the optimized Gaussian prediction head together with the otherwise frozen feed-forward model defines the attacked model for the current scene. Inference uses the original context images and the standard feed-forward rendering pipeline, without requiring the target message as an additional model input and without applying any post-rendering watermarking operation.

For scene-wise evaluation, the original Gaussian-head parameters are restored after the attacked renderings of each scene are recorded. This reset is used only to ensure independent evaluation across scenes and is not part of the message injection or extraction procedure.

Table 6: Efficiency and computational cost of model-level adaptation. We compare unrestricted fine-tuning with carrier-restricted fine-tuning on MVSplat using the RealEstate10K dataset. Adapt. Time is the adaptation time measured per scene for 150 optimization steps, GPU Mem. denotes the peak allocated memory, and FPS measures novel view rendering speed.

Methods	Adapt. Time (s) ↓	FPS ↑	GPU Mem. (MB) ↓	Bit Acc ↑	PSNR ↑
Unrestricted Fine-tuning	7.91	540.28	2054	87.07	25.56
Carrier-restricted Fine-tuning	9.20	515.64	2057	99.88	26.20

B Appendix-Additional Experiments

B.1 Additional Experimental Details

RealEstate10K contains real-estate videos collected from YouTube, with 7,289 scenes in the test split, while ACID contains 1,972 test scenes captured from aerial drone videos. Both datasets provide camera parameters for each frame. Watermark recovery is evaluated using Bit Accuracy (Bit Acc), defined as the proportion of target message bits correctly decoded from the rendered novel views by the frozen watermark decoder. Rendering fidelity is measured using PSNR (Peak Signal-to-Noise Ratio), SSIM (Structural Similarity Index Measure), and LPIPS (Learned Perceptual Image Patch Similarity) between the attacked and corresponding original renderings.

In all experiments, the context views remain unchanged throughout adaptation. Only the Gaussian prediction head is fine-tuned, while the remaining components of the pretrained feed-forward 3DGS model and the watermark decoder are kept frozen. The reported metrics therefore characterize the trade-off between message recovery and preservation of the original rendering quality. All experiments are implemented in PyTorch and conducted on a single NVIDIA A6000 GPU.

B.2 More Analyses regarding Bit Accuracy of the Original Model

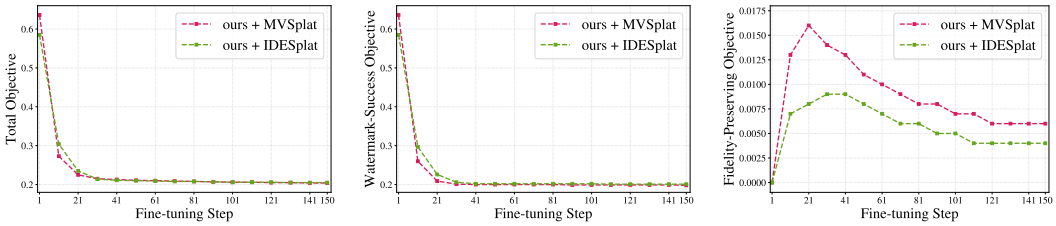
The original pretrained models in Table 1 are evaluated without a watermarking approach or parameter adaptation. Since the target binary message is not used to generate these clean renderings, there is no deliberate correlation between the decoder outputs and the target bits. Consequently, their Bit Acc remains close to 50%, which corresponds to the expected chance level for binary message decoding. This baseline confirms that the high Bit Acc obtained after adaptation results from the injected model-level watermark rather than an inherent response of the pretrained reconstruction model or watermark decoder.

B.3 Efficiency and Computational Cost of Adaptation Overhead

Table 6 compares the computational overhead of unrestricted fine-tuning and carrier-restricted fine-tuning. Since both methods update only the Gaussian prediction head while keeping the remaining feed-forward 3DGS pipeline frozen, their adaptation time and peak GPU memory remain comparable. Carrier-restricted fine-tuning introduces only lightweight carrier selection and masked prediction mixing, without modifying the underlying rendering pipeline. Consequently, the rendering speed after adaptation remains essentially unchanged, while carrier-restricted fine-tuning achieves substantially higher Bit Acc than unrestricted fine-tuning. These results indicate that the improved watermark recovery of carrier-restricted fine-tuning is obtained with limited additional computational overhead.

B.4 Optimization Convergence

Figure 5 shows the optimization curves of our carrier-restricted fine-tuning on RealEstate10K with MVSplat and IDESplat. In Figure 5a, the total objective decreases rapidly during the first several iterations and gradually converges afterward, indicating stable optimization for both backbones. Figure 5b shows a similar trend for the watermark-success objective, which quickly drops and stabilizes around a small non-zero value. This is expected because the binary cross-entropy term remains positive for finite decoder logits even when the target bits are already decoded correctly. As shown



(a) Total Objective (b) Watermark-Success Objective (c) Fidelity-Preserving Objective

Figure 5: Optimization convergence of our carrier-restricted fine-tuning on RealEstate10K with two feed-forward 3DGS backbones. We report the total objective, the watermark-success objective, and the fidelity-preserving objective over 150 fine-tuning steps.

Table 7: Ablation on optimization steps and learning rate.

Steps	Bit Acc ↑	PSNR ↑	SSIM ↑	LPIPS ↓	Learning rate	Bit Acc ↑	PSNR ↑	SSIM ↑	LPIPS ↓
50	96.02	26.26	0.865	0.134	1×10^{-5}	81.28	26.33	0.867	0.131
70	97.58	26.25	0.865	0.135	5×10^{-5}	98.04	26.25	0.865	0.135
100	98.48	26.22	0.864	0.137	1×10^{-4}	99.88	26.20	0.864	0.137
150	99.88	26.20	0.864	0.137	5×10^{-4}	99.54	25.55	0.838	0.181

in Figure 5c, the fidelity-preserving objective starts from zero, increases during the early stage as watermark information is injected, and then gradually decreases as the optimization reduces unnecessary rendering deviations. After convergence, the fidelity term remains small but non-zero, reflecting the expected trade-off between reliable message recovery and preservation of the original rendering quality.

B.5 Effectiveness of Optimization Steps and Learning Rate

Table 7 analyzes the sensitivity to the number of fine-tuning steps and learning rate. Increasing the steps consistently improves watermark recovery, with Bit Acc rising from 96.02% at 50 steps to 99.88% at 150 steps, while the rendering quality remains relatively stable with minor variations. A similar trend is observed for the learning rate, and increasing it substantially improves Bit Acc from 81.28% to 99.88%. However, a further increase slightly reduces Bit Acc and noticeably degrades rendering quality, with PSNR dropping to 25.55 and LPIPS increasing to 0.181. These results indicate that these optimization settings provide the best trade-off between reliable watermark recovery and visual fidelity.

B.6 Additional Qualitative Results

Moreover, Figure 6, Figure 7, Figure 8, and Figure 9 provide additional qualitative comparisons on RealEstate10K and ACID datasets using MVSplat and IDESplat. Across different scenes and views, unrestricted fine-tuning introduces noticeable appearance changes, which are reflected by strong and spatially widespread responses in the $\times 10$ residual maps, particularly around object boundaries, textured regions, and fine structures. In contrast, our carrier-restricted fine-tuning consistently produces renderings that remain visually closer to the original reconstruction, with substantially weaker and more localized residual responses. This behavior is observed for both indoor and outdoor scenes and across the two feed-forward 3DGS backbones, indicating that the fidelity improvement is not specific to a particular scene or model architecture. These qualitative results further support the quantitative findings in the main paper, showing that restricting adaptation to selected Gaussian carriers effectively suppresses unnecessary rendering modifications while maintaining the injected watermark across the evaluated views.

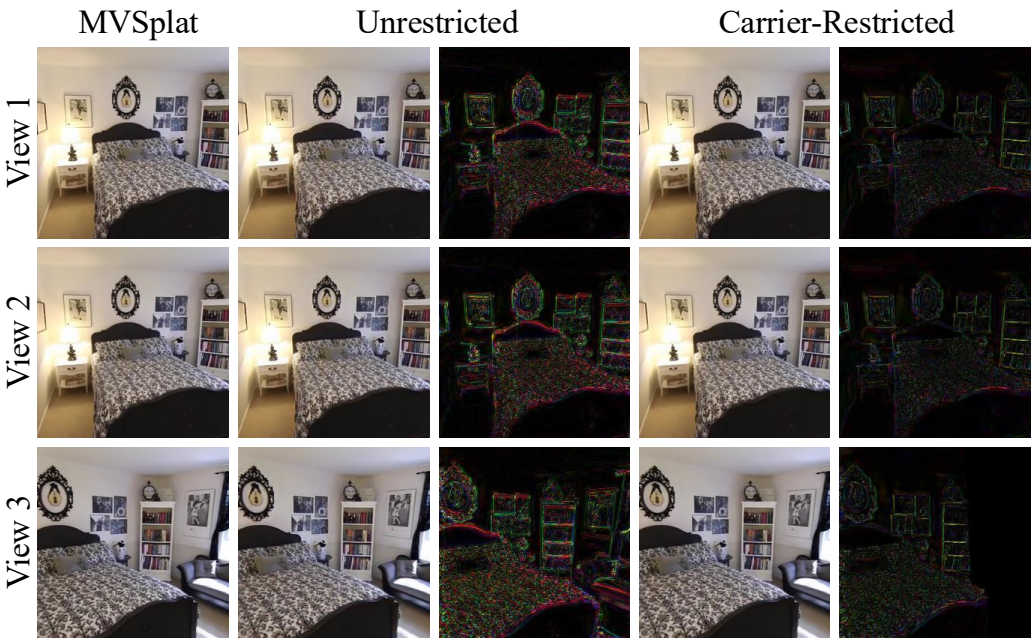


Figure 6: Detailed qualitative comparison on RealEstate10K dataset. We compare the original renderings from MVSplat, unrestricted fine-tuning, and carrier-restricted fine-tuning approaches with multiple novel views from the same scene. The corresponding pixel-wise residual maps are $\times 10$ for better view. Our carrier-restricted fine-tuning approach consistently produces smaller visual deviations from the original renderings across different views, indicating improved fidelity.

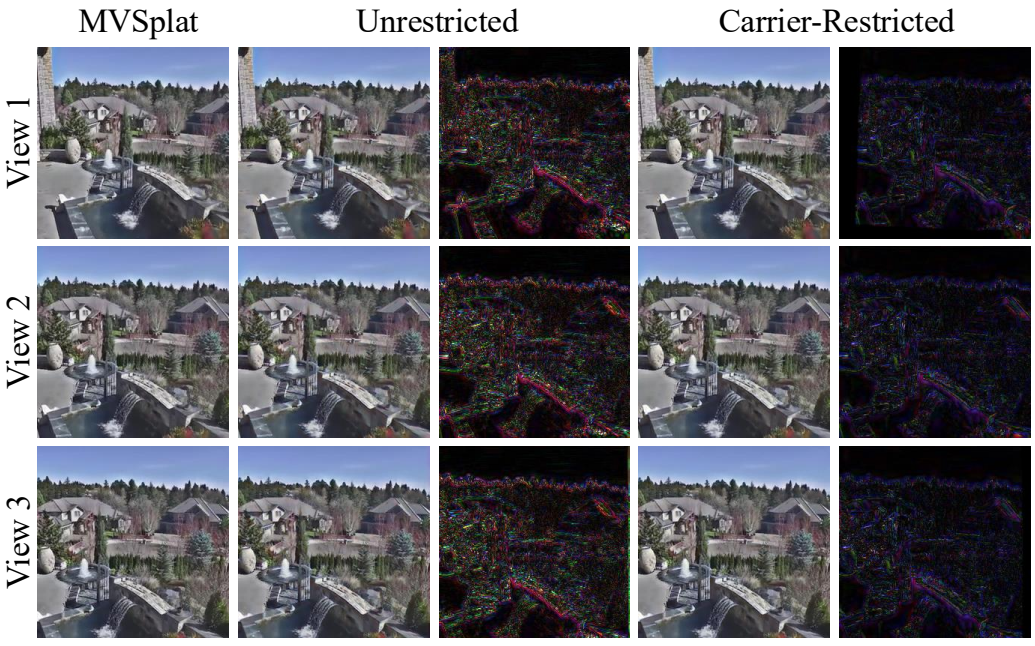


Figure 7: Detailed qualitative comparison on RealEstate10K dataset. We compare the original renderings from MVSplat, unrestricted fine-tuning, and carrier-restricted fine-tuning approaches with multiple novel views from the same scene. The corresponding pixel-wise residual maps are $\times 10$ for better view. Our carrier-restricted fine-tuning approach consistently produces smaller visual deviations from the original renderings across different views, indicating improved fidelity.

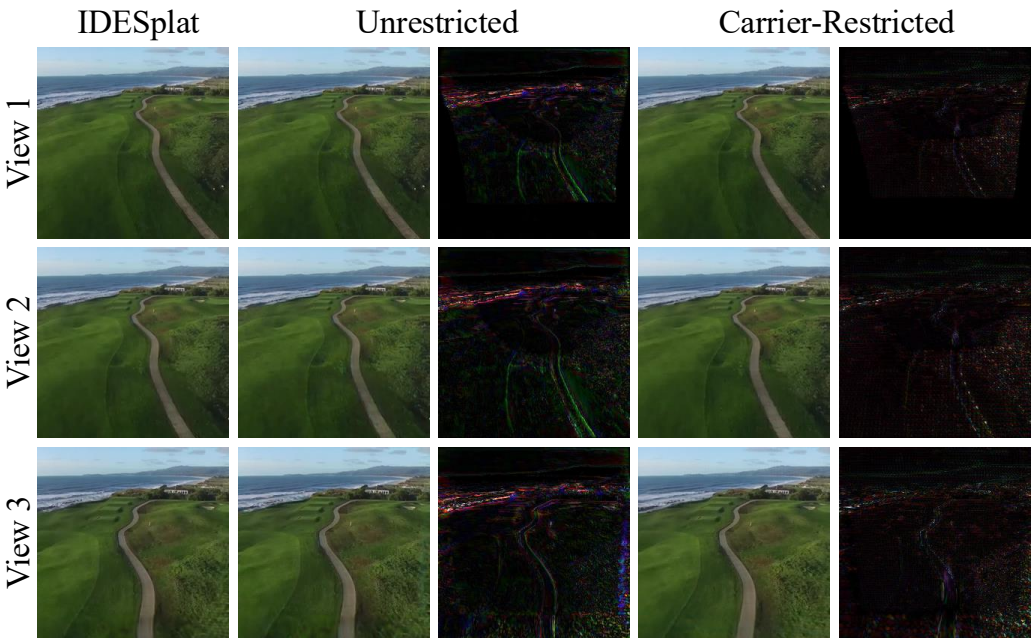


Figure 8: Detailed qualitative comparison on ACID dataset. We compare the original renderings from IDESplat, unrestricted fine-tuning, and carrier-restricted fine-tuning approaches with multiple novel views from the same scene. The corresponding pixel-wise residual maps are $\times 10$ for better view. Our carrier-restricted fine-tuning approach consistently produces smaller visual deviations from the original renderings across different views, indicating improved fidelity.

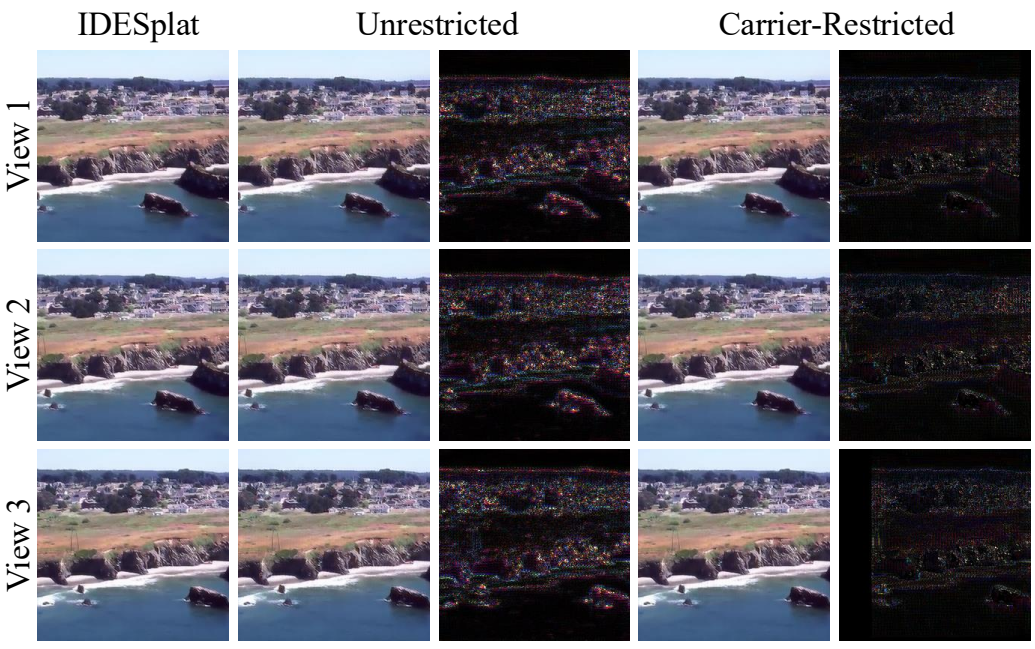


Figure 9: Detailed qualitative comparison on ACID dataset. We compare the original renderings from IDESplat, unrestricted fine-tuning, and carrier-restricted fine-tuning approaches with multiple novel views from the same scene. The corresponding pixel-wise residual maps are $\times 10$ for better view. Our carrier-restricted fine-tuning approach consistently produces smaller visual deviations from the original renderings across different views, indicating improved fidelity.