ADVSPLAT: ADVERSARIAL ATTACKS ON FEED-FORWARD GAUSSIAN SPLATTING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

3D Gaussian Splatting (3DGS) is increasingly recognized as a powerful paradigm for real-time, high-fidelity 3D reconstruction. However, its per-scene optimization pipeline limits scalability and generalization, and prevents efficient inference. Recently emerged feed-forward 3DGS models address these limitations by enabling fast reconstruction from a few input views after large-scale pretraining, without scene-specific optimization. Despite their advantages and strong potential for commercial deployment, the use of neural networks as the backbone also amplifies the risk of adversarial manipulation. In this paper, we introduce AdvSplat, the first systematic study of adversarial attacks on feed-forward 3DGS. We first employ white-box attacks to reveal fundamental vulnerabilities of this model family. We then develop two improved, practically relevant, query-efficient black-box algorithms that optimize pixel-space perturbations via a frequency domain parameterization: one based on gradient estimation and the other gradient-free, without requiring any access to model internals. Extensive experiments across multiple datasets demonstrate that AdvSplat can significantly disrupt reconstruction results by injecting imperceptible perturbations into the input images. Our findings surface an overlooked yet urgent problem in this domain, and we hope to draw the community's attention to this emerging security and robustness challenge.

1 INTRODUCTION

3D reconstruction recovers 3D geometry and structure from multi-view 2D images. Representative paradigms include NeRF (Mildenhall et al., 2021) and 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023), the latter becoming dominant due to real-time rendering and high-fidelity reconstruction. These strengths have driven 3DGS adoption in robotics (Keetha et al., 2024; Lu et al., 2024a; Matsuki et al., 2024; Zou et al., 2025), autonomous driving (Hess et al., 2025; Khan et al., 2025; Zhou et al., 2024), and content generation (Xiang et al., 2025a;b; Hunyuan3D et al., 2025; Bahmani et al., 2025; Wu et al., 2025b). However, conventional 3DGS relies on per-scene optimization, limiting inference efficiency, scalability, and generalization, and often underperforming in sparse-view settings (Wu et al., 2025a). These limitations hinder its practicality in few-view, near-instant reconstruction scenarios (Liu et al., 2025; Charatan et al., 2024). In contrast, *feed-forward 3DGS* (Szymanowicz et al., 2024; Charatan et al., 2024) alleviates these issues through large-scale pretraining with few-view inputs (Zhang et al., 2025), offering strong potential for foundation-model-style 3D reconstruction and broader commercial deployment.

Given these desirable properties and strong potential of feed-forward 3DGS, ensuring its robustness becomes critically important. While a few studies (Lu et al., 2024b; Zeybey et al., 2024) have revealed the vulnerability of conventional 3DGS by designing adversarial attacks, the vulnerability of feed-forward 3DGS remains largely unexplored. This raises an overlooked yet crucial question, one that we aim to answer in this paper: *How robust are feed-forward 3DGS models against adversarial attacks, and how can they be effectively attacked?* Although white-box attacks are simple and effective, their applicability in real-world scenarios is limited by the requirement of access to model internals. Motivated by this, we focus on the more practical *black-box attack setting*. This problem remains uninvestigated in the community, especially due to two key challenges. **Challenge 1: unknown model parameter weights.** In practice, the internal weights of commercial models are typically unavailable. Instead, access is restricted to API queries, i.e., only the model's inputs and outputs can be observed. This makes the design of effective attacks substantially more challenging,

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: White-box attack (here we use PGD attack Madry et al. (2017)) results against feed-forward 3DGS model (here we use NoPoSplatYe et al. (2024a))) on the RE10K dataset. "Ref." denotes two input reference views, and "Novel Views" denote the newly rendered target viewpoints. The first row shows results on clean inputs, while the second row shows results on adversarially perturbed inputs. The rightmost radar chart shows the rendered image quality with clean/attacked inputs, where PSNR is normalized with 30 as the maximum value. As shown, both quantitative and qualitative results consistently indicate that even imperceptible perturbations can severely degrade the reconstruction quality.

especially for feed-forward 3DGS models which are often large-scale. **Challenge 2: high computational costs.** Traditional black-box attacks are typically categorized into transfer-based and query-based attacks. Feed-forward 3DGS models often exhibit substantially larger feature-space discrepancies than conventional classifiers, largely because they are built upon different high-capacity transformer backbones (Wang et al., 2025; Yang et al., 2024a) with distinct pretraining data, architectural biases, and representation geometries, making transfer-based methods almost impossible. Query-based attacks can be further divided into non-optimization (Andriushchenko et al., 2020; Guo et al., 2019; Brendel et al., 2017) and optimization-based (Ilyas et al., 2018) approaches. Non-optimization attacks often rely on decision-boundary exploration or random noise search, but both are ineffective for feed-forward 3DGS models: these models lack a decision boundary since their tasks are not classification, and their large capacity makes them relatively insensitive to random noise. We therefore focus on optimization-based black-box attacks, which optimize adversarial perturbations through model queries. However, despite faster inference compared to traditional 3DGS, feed-forward 3DGS models remain large, making the per-query cost a bottleneck for query-intensive attacks. Moreover, reconstruction outputs lie in a high-dimensional pixel space, requiring extensive queries to explore adversarial perturbations, further exacerbating this challenge.

In this paper, we **answer the research question of feed-forward 3DGS robustness and design effective adversarial attacks against them**. As an initial experiment to reveal the model vulnerability, we applied representative white-box attacks (in particular, Projected Gradient Descent (PGD) (Madry et al., 2017)) designed for conventional classifiers to attack feed-forward 3DGS models. Unlike traditional 3DGS, the neural network architecture of feed-forward 3DGS makes it possible to leverage gradient information to craft adversarial perturbations on the input images. By constructing subtle perturbations through maximizing the reconstruction loss, we observe a significant degradation in rendering quality, as shown in **??**. This reveals a potential risk: *feed-forward 3DGS models are not inherently robust, and an attacker can severely corrupt the model's outputs by introducing imperceptible perturbations*. Inspired by this, we design black-box attacks via optimizing subtle yet misleading perturbations through repeated queries to the feed-forward 3DGS model, either in a *gradient-based* or a *gradient-free* manner (addressing Challenge 1). To reduce the number of queries and improve efficiency, we devise a Discrete Cosine Transform (DCT) based approach. Specifically, we first partition the image into equal-sized blocks and apply DCT to each block to transform it into the frequency domain. We then perturb only the low-frequency components, whose dimensionality is far lower than that of the pixel space, and transform them back to the image space via the inverse DCT (addressing Challenge 2).

Given the vulnerabilities shown above, a highly plausible real-world scenario is that an adversary can repeatedly query a feed-forward 3DGS model to generate adversarially perturbed images and

subsequently upload them to the internet. Users may download such data without noticing any abnormality. Once these attacked inputs lead to low-quality reconstructions, both the company's commercial reputation and user trust may be adversely affected. Our work identifies and systematically investigates this issue, and draws the efficient and generalizable 3D reconstruction community's attention to security and robustness. The contributions of this work can be concluded in threefold:

- To the best of our knowledge, this work is the first to investigate a previously overlooked yet important security issue in the deployment of feed-forward 3DGS models, revealing their vulnerabilities to adversarial attacks.
- This paper proposes two black-box attack algorithms for feed-forward 3DGS, including a gradient-based and a gradient-free method, both operating in the frequency domain and requiring access only to model inputs and outputs.
- Extensive experiments across state-of-the-art feed-forward 3DGS models and widely used datasets demonstrate that the proposed methods consistently achieve successful attacks.

2 Related Work

3D Gaussian Splatting. 3DGS (Kerbl et al., 2023) represents scenes using anisotropic Gaussian ellipsoids parameterized by position, scale, orientation, and color. Unlike NeRF (Mildenhall et al., 2021), which repeatedly queries an MLP-based radiance field, 3DGS enables high-fidelity, real-time rendering. Its explicit, interpretable, and scalable representation supports virtual reality (Jiang et al., 2024; Tu et al., 2024), dynamic scene reconstruction (Yang et al., 2023; 2024b; Wu et al., 2024a), open-vocabulary 3D semantic segmentation (Wu et al., 2024b; Qin et al., 2024; Shi et al., 2024; Ye et al., 2024b; Lu et al., 2025), and 3D editing (Qu et al., 2025; Dong & Wang, 2026; Yan et al., 2024). However, per-scene optimization limits generalization, scalability, and real-time inference.

Feed-Forward 3D Gaussian Splatting. Pretrained on large-scale datasets (Zhang et al., 2025), feed-forward 3DGS predicts Gaussians from sparse input images in a single forward pass, enabling fast inference and strong novel-view synthesis. We categorize these methods as pose-known or pose-free. **Pose-Known Methods.** PixelSplat (Charatan et al., 2024) uses an epipolar transformer to infer scene scale and probabilistic depths for Gaussian placement. LatentSplat (Wewer et al., 2024) generatively models latent uncertainty to improve reconstruction in uncertain regions. To improve unreliable feature-to-depth mapping, MVSplat (Chen et al., 2024) exploits plane-swept cost volumes and cross-view similarities, while DepthSplat (Xu et al., 2025) incorporates pretrained monocular depth features. **Pose-Free Methods.** Since accurate camera poses are difficult to obtain, NoPoSplat (Ye et al., 2024a) predicts Gaussians in an input-view canonical coordinate system without ground-truth poses. SelfSplat (Kang et al., 2025) couples explicit Gaussians with self-supervised depth and pose estimation. AnySplat (Jiang et al., 2025) jointly predicts Gaussians and poses, using differentiable voxelization to aggregate pixel-wise primitives into voxel-wise Gaussians for efficiency.

Adversarial Attacks. Adversarial attacks manipulate classification predictions through imperceptible, non-random input perturbations (Szegedy et al., 2013). White-box attacks exploit model gradients (Moosavi-Dezfooli et al., 2016; Goodfellow et al., 2014; Madry et al., 2017), whereas black-box attacks rely on transferability (Liu et al., 2016) or query-based optimization (Ilyas et al., 2018; Andriushchenko et al., 2020; Guo et al., 2019; Brendel et al., 2017). For 3DGS, PoisonSplat (Lu et al., 2024b) formulates a bilevel computation-cost attack, exposing its vulnerability. However, security and robustness analyses of feed-forward 3DGS remain absent; our work fills this gap.

3 Method

We first reveal the vulnerability of feed-forward 3DGS models via white-box attacks using PGD (assuming access to the weights). We then focus on black-box attacks and propose two improved, effective, and query-efficient attack algorithms (gradient-based and gradient-free) based on an efficient frequency domain parameterization of perturbations, as shown in Fig. Figure 2.

3.1 Preliminaries

Feed-Forward 3DGS. Suppose we are given N input images $\{\boldsymbol{I}^i\}_{i=1}^N$ ($\boldsymbol{I}^i \in \mathbb{R}^{H\times W\times 3}$, H is image height and W is image width) along with their corresponding camera projection matrices $\{\boldsymbol{P}^i\}_{i=1}^N$

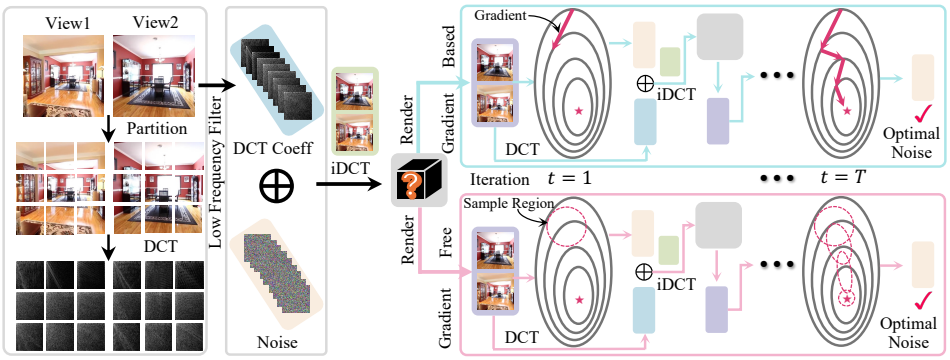


Figure 2: Pipeline of our proposed method, including gradient-based and gradient-free variants. The blue blocks denote the DCT coefficient maps, the yellow blocks denote the sampled noise, the green blocks denote the images after iDCT, and the purple blocks denote the rendered images. The black box in the center is the feed-forward 3DGS model with unknown parameters, illustrated by the gray block. The gray ellipses depict a contour map of the loss function, and the red star indicates the maximum. The red arrow inside the ellipse shows the gradient direction, while the red dashed ellipse indicates the noise sampling region. Finally, the noise sample marked with a red checkmark is the optimal perturbation we seek.

(calculated via intrinsic $\boldsymbol{K}^i$, rotation $\boldsymbol{R}^i$ and translation $\boldsymbol{t}^i$: $\boldsymbol{P}^i = \boldsymbol{K}^i[\boldsymbol{R}^i|\boldsymbol{t}^i]$, and can be empty for pose-free settings). The goal of feed-forward 3DGS is to learn a mapping $\boldsymbol{f_\theta}$ from input images to the parameters of 3D Gaussians:

$$\boldsymbol{f_\theta} : \{(\boldsymbol{I}^i, \boldsymbol{P}^i)\}_{i=1}^N \mapsto \{(\boldsymbol{\mu}_j, \alpha_j, \boldsymbol{\Sigma}_j, \boldsymbol{c}_j)\}_{j=1}^{H\times W\times N}, \tag{1}$$

where $\boldsymbol{\theta}$ denotes the parameters of the feed-forward network; $\boldsymbol{\mu}_j$ is the Gaussian mean; α_j is the opacity; $\boldsymbol{\Sigma}_j$ is the covariance; and $\boldsymbol{c}_j$ denotes the spherical harmonics coefficients. Given a specific camera pose, each pixel (p, q) color $\boldsymbol{I}_r(p, q)$ is calculated via alpha blending:

$$\boldsymbol{I}_r(p, q) = \sum_{j\in\mathcal{V}} c_j\alpha_j \prod_{l=1}^{j-1}(1 - \alpha_l), \tag{2}$$

where $\mathcal{V}$ is a set of visible Gaussians that contribute to the pixel.

PGD Attack. PGD is a white-box adversarial attack algorithm designed for a traditional classifier that iteratively updates an input image using the gradient of the loss $\mathcal{L}$ with respect to the input x, seeking a perturbation that most degrades the model's decision. Starting from the clean image, PGD performs multiple update steps; after each step t, the intermediate adversarial sample x_t is projected back into an ℓ_∞ constraint set centered at the original input with a specified radius. The procedure is summarized as follows:

$$x_{t+1} = \Pi_{x+\mathcal{S}}\left(x_t + \eta\,\mathrm{sgn}\left(\nabla_x\mathcal{L}(\theta, f(x_t), y)\right)\right), \tag{3}$$

where y is the label, Π is the projection operator, $\mathcal{S}$ defines the radius of the ℓ_∞-norm ball, η denotes the step size, $\mathrm{sgn}(\cdot)$ is the sign function, and θ represents the parameters of the target model.

3.2 Vulnerability of Feed-Forward 3DGS Model

Similar to classification models, exposing the vulnerability of feed-forward 3DGS can be cast as an equivalent objective: whether one can construct imperceptible perturbations on the input images that substantially degrade reconstruction quality. To achieve this goal, we assume full access to the weights of the feed-forward 3DGS model. We then employ PGD, as shown in Equation (3), to perform multi-step attacks and generate adversarial examples. Unlike classification, reconstruction does not involve discrete labels. Nevertheless, the attack objective is analogous: to increase the loss by crafting the manipulation of the input along the gradient direction. Instead of a classification loss, we use a standard combination of a photometric loss and an LPIPS loss as the overall objective.

Although different methods employ different training losses, these two terms are almost universally adopted and typically contribute the most, making our choice both reasonable and broadly applicable. We formulate the white-box attacks on feed-forward 3DGS models as follows:

$$\boldsymbol{I}_{t+1} = \Pi_{\boldsymbol{I}+\mathcal{S}} \left(\boldsymbol{I}_t + \eta \operatorname{sgn} \left(\nabla_{\boldsymbol{I}} \mathcal{L}(\theta, f(\boldsymbol{I}_t), \boldsymbol{I}_t^r) \right) \right), \tag{4}$$

$$\mathcal{L} = \text{MSE}(\boldsymbol{I}_t, \boldsymbol{I}_t^r) + \lambda \text{LPIPS}(\boldsymbol{I}_t, \boldsymbol{I}_t^r), \tag{5}$$

where MSE stands for mean square error, $\boldsymbol{I}_t$ are input images at step t, $\boldsymbol{I}_t^r$ are rendered images at step t, with the same poses of $\boldsymbol{I}_t$, λ is the balancing weight. The results in Figure 1 show that white-box attacks are effective against feed-forward 3DGS models, producing adversarial examples that cause the reconstruction quality to collapse. This is also reflected consistently by the quantitative metrics in the radar map (more details in Appendix C), revealing the model's inherent vulnerability.

3.3 Black-Box Attack on Feed-Forward 3DGS Model

As discussed in Section 1, an adversarial example crafted on model A does not substantially affect feature extraction in model B. We validate this empirically, with results provided in Appendix D. Moreover, non-optimization query-based attacks are inherently ineffective against feed-forward 3DGS models. Based on this, we focus on optimization-based query attacks.

Frequency-domain Parameterization of Perturbations. In query-based attacks, the required number of queries is positively correlated with the dimensionality of the attacked object: as the dimensionality increases, the number of explorable directions also grows, requiring more queries to distinguish favorable optimization directions from random search directions. Because input images in 3D reconstruction tasks are typically high-resolution, a large number of queries is required. This introduces several drawbacks. A larger number of queries substantially increases the computational overhead, thereby raising both the time and monetary costs of the attack. To reduce the number of queries and improve attack efficiency, we propose a frequency-domain parameterization of perturbations.

3D-GSW (Jang et al., 2024) points out that the low-frequency components of an image contain the main structural information and can support robust information embedding. Inspired by this, we apply perturbations to the low-frequency components. We first divide the image into blocks $\boldsymbol{I_b}$ of equal size, each of size $n \times n$. Applying DCT in a block-wise manner provides a localized frequency representation, enabling finer spatial control and more efficient manipulation of region-specific image structures. In this work, we use the orthonormal DCT-II transform matrix $\boldsymbol{C} \in \mathbb{R}^{n \times n}$. The (k, i)-th entry $\boldsymbol{C}_{k,i}$ specifies the contribution of the input sample at spatial index $i \in \{0, 1, \ldots, n-1\}$ to the DCT coefficient at frequency index $k \in \{0, 1, \ldots, n-1\}$ (with $k = 0$ corresponding to the DC component), and is defined as

$$\boldsymbol{C}_{k,i} = \gamma(k) \cos\left(\frac{\pi}{n} \left(i + \frac{1}{2} \right) k \right), \tag{6}$$

where $\gamma(k)$ is the normalization factor ensuring orthonormality (i.e., $\boldsymbol{C}\boldsymbol{C}^\top = \boldsymbol{I}$), given by $\gamma(0) = \sqrt{\frac{1}{n}}$ and $\gamma(k) = \sqrt{\frac{2}{n}}$ for $k \geq 1$, and the half-sample shift $\left(i + \frac{1}{2}\right)$ is the standard DCT-II sampling convention. Then the DCT coefficient matrix $\boldsymbol{F}$ of $\boldsymbol{I_b}$ is obtained by:

$$\boldsymbol{F}^j = \boldsymbol{C}^j \boldsymbol{I}_b^j \boldsymbol{C}^{j\top}, \tag{7}$$

where j represents the j-th block. During the attack, we perturb only the low-frequency components in each DCT coefficient block, i.e., the top-left $s \times s$ sub-block $\boldsymbol{F}^j_{0:s-1,\,0:s-1}$ for each j, while keeping the remaining coefficients unchanged. After applying the perturbation, we use the inverse DCT (iDCT) to transform the perturbed coefficient map back to the image space:

$$\boldsymbol{I}_b^{j\prime} = \boldsymbol{C}^{j\top} \begin{bmatrix} \boldsymbol{F}^j_{0:s-1,\,0:s-1} + \boldsymbol{\delta}^j & \boldsymbol{F}^j_{0:s-1,\,s:n-1} \\ \boldsymbol{F}^j_{s:n-1,\,0:s-1} & \boldsymbol{F}^j_{s:n-1,\,s:n-1} \end{bmatrix} \boldsymbol{C}^j, \tag{8}$$

where $\boldsymbol{I}_b^{j\prime}$ is the j-th perturbed image block, $\boldsymbol{\delta}^j \in \mathbb{R}^{s \times s}$ denotes the low-frequency perturbation. For simplicity, we denote the intermediate block-wise coefficient matrix as $\boldsymbol{F}^j_{ss} + \boldsymbol{\delta}^j_{ss}$. After concatenating all image blocks, we obtain the final perturbed image $\boldsymbol{I}'$. Although we optimize

the perturbation in the frequency domain, applying the iDCT projects it back to the pixel space, allowing us to obtain the desired adversarial example. Next, we apply our method to two black-box attack techniques to construct adversarial examples.

Gradient-based Approach. We leverage a widely used variant of Natural Evolution Strategies (NES)(Ilyas et al., 2018) to estimate gradients for feed-forward 3DGS models, which approximates gradients by averaging loss-weighted perturbations sampled from a noise distribution. The procedure can be formulated as follows:

$$\nabla_{\boldsymbol{I}}\mathcal{L}(\theta) \approx \frac{1}{2M\sigma}\sum_{m=1}^{M}(\mathcal{L}(\boldsymbol{I}'^{+}_{m}, \boldsymbol{I}'^{+}_{r,(m)}) - \mathcal{L}(\boldsymbol{I}'^{-}_{m}, \boldsymbol{I}'^{-}_{r,(m)})\mathbf{u}_m, \tag{9}$$

where

$$\boldsymbol{I}'^{+}_{m} = \text{cat}_{j=1}^{J}(\boldsymbol{C}^{j\top}(\boldsymbol{F}^{j}_{ss} + \sigma\boldsymbol{u}^{j}_{m})\boldsymbol{C}^{j}), \boldsymbol{I}'^{-}_{m} = \text{cat}_{j=1}^{J}(\boldsymbol{C}^{j\top}(\boldsymbol{F}^{j}_{ss} - \sigma\boldsymbol{u}^{j}_{m})\boldsymbol{C}^{j}), \tag{10}$$

$\boldsymbol{u}_m$ is standard Gaussian noise, $\text{cat}(\cdot)$ denotes the concatenation operation, σ is the search variance, J is the number of blocks, and M is the number of noise samples, $\boldsymbol{I}'^{+}_{r,(m)}$ and $\boldsymbol{I}'^{-}_{r,(m)}$ are the corresponding rendered images. Once we obtain the estimated gradient, we can employ it in Equation (4) to optimize the perturbation. The detailed algorithm is provided in Appendix B.

Gradient-free Approach. Although gradient-based methods are intuitive and efficient, they can fail when the model is insensitive to noise or when the gradient is non-smooth. Therefore, we apply our approach to another gradient-free black-box search technique, covariance matrix adaptation evolution strategy (CMA-ES)(Hansen, 2016). CMA-ES iteratively samples candidate perturbations from a multivariate Gaussian distribution and adapts its mean and covariance based on the best-performing samples to guide the search toward better optima. At each iteration t, we sample $\mathcal{B}$ candidate perturbations on DCT coefficient:

$$\boldsymbol{\delta}^{j,(t)}_{\beta} \sim \mathcal{N}_{\beta}\big(\boldsymbol{a}^{(t)}, \boldsymbol{V}^{(t)}\big), \qquad \beta = 1, 2, \ldots, \mathcal{B}, \tag{11}$$

where $\boldsymbol{a}^{(t)}$ is the mean and $\boldsymbol{V}^{(t)}$ is the covariance matrix. Their initial values are set to zero and the identity matrix, respectively. We use the first expression in Equation (10) to obtain the perturbed image $\boldsymbol{I}'^{(t)}_{\beta}$ ($\boldsymbol{I}'^{(t)}_{\beta} = \text{cat}_{j=1}^{J}(\boldsymbol{C}^{j\top}(\boldsymbol{F}^{j,(t)}_{ss} + \boldsymbol{\delta}^{j,(t)}_{\beta})\boldsymbol{C}^{j})$), and then perform a forward pass through the model to obtain the rendered image $\boldsymbol{I}'^{(t)}_{r,(\beta)}$. Then, we compute the loss of the $\mathcal{B}$ candidate pertubed images using Equation (5):

$$\mathcal{L}\Big(\boldsymbol{I}'^{(t)}_{1:\mathcal{B}}, \boldsymbol{I}'^{(t)}_{r,(1:\mathcal{B})}\Big) \;\geq\; \mathcal{L}\Big(\boldsymbol{I}'^{(t)}_{2:\mathcal{B}}, \boldsymbol{I}'^{(t)}_{r,(2:\mathcal{B})}\Big) \;\geq\; \cdots \;\geq\; \mathcal{L}\Big(\boldsymbol{I}'^{(t)}_{\mathcal{B}:\mathcal{B}}, \boldsymbol{I}'^{(t)}_{r,(\mathcal{B}:\mathcal{B})}\Big), \tag{12}$$

sort them in descending order, and select the top $\mathcal{B}/2$ perturbation candidates, which aligns with our attack objective of maximizing the loss, effectively providing a gradient-like search direction without explicit gradients. Next, the mean and covariance are updated by the following formulations:

$$\boldsymbol{a}^{(t+1)} \;=\; \sum_{\beta=1}^{\mathcal{B}/2} w_{\beta}\,\boldsymbol{\delta}^{(t)}_{\beta:\mathcal{B}}, \tag{13}$$

$$\boldsymbol{V}^{(t+1)} = (1-c_{\mu})\,\boldsymbol{V}^{(t)} + c_{\mu}\sum_{\beta=1}^{\mathcal{B}/2} w_{\beta}\,\big(\boldsymbol{\delta}^{(t)}_{\beta:\mathcal{B}} - \boldsymbol{a}^{(t)}\big)\big(\boldsymbol{\delta}^{(t)}_{\beta:\mathcal{B}} - \boldsymbol{a}^{(t)}\big)^{\top}, \tag{14}$$

where c_{μ} is the learning rate for updating the covariance matrix, $\boldsymbol{\delta}^{(t)}_{\beta:\mathcal{B}} = \text{cat}_{j=1}^{J}(\boldsymbol{\delta}^{j,(t)}_{\beta:\mathcal{B}})$. w_{β} is the recombination weight of the β-th ranked perturbation, which is computed by $w'_{\beta} = ln(\frac{\mathcal{B}+1}{2}) - ln(\beta)$ and then normalized by $w_{\beta} = w'_{\beta}/\sum_{\beta=1}^{\mathcal{B}/2} w'_{\beta}$. At final step, the noise sampled from $\mathcal{N}\big(\boldsymbol{a}^{(T)}, \boldsymbol{V}^{(T)}\big)$ is the desired perturbation. The detailed algorithm is provided in Appendix B.

4 Experiments

In this section, we answer the following research questions through extensive experiments: **RQ1:** How effective are our attack algorithms against different feed-forward 3DGS victim models? **RQ2:** Can the DCT-based strategy improve attack efficiency? **RQ3:** How do different levels of attack strength affect the victim model performance?

Table 1: Quantitative comparison of victim model performance with vs. without our attack on the RE10K dataset. GB denotes the gradient-based variant, and GF denotes the gradient-free variant. Percentages measure the degree of performance degradation.

Victim Models	PSNR ↓	SSIM ↓	LPIPS ↑	CLIP ↓	DINO ↓
DepthSplat	21.09	0.710	0.228	0.956	0.930
DepthSplat+Ours (GB)	7.57(−64.1%)	0.289(−59.3%)	0.581(+154.8%)	0.740(−22.3%)	0.492(−47.1%)
DepthSplat+Ours (GF)	9.73(−53.9%)	0.437(−38.5%)	0.514(+125.4%)	0.803(−16.0%)	0.727(−21.8%)
NoPoSplat	22.50	0.781	0.165	0.957	0.938
NoPoSplat+Ours (GB)	17.64(−21.6%)	0.523(−33.0%)	0.417(+152.7%)	0.893(−6.7%)	0.816(−13.0%)
NoPoSplat+Ours (GF)	14.02(−37.7%)	0.397(−49.2%)	0.549(+232.7%)	0.856(−10.6%)	0.729(−22.3%)
AnySplat	18.94	0.672	0.271	0.928	0.938
AnySplat+Ours (GB)	12.80(−32.4%)	0.431(−35.9%)	0.588(+117.0%)	0.813(−12.4%)	0.791(−15.7%)
AnySplat+Ours (GF)	14.49(−23.5%)	0.460(−31.5%)	0.561(+107.0%)	0.859(−7.4%)	0.832(−11.3%)

Table 2: Quantitative comparison of victim model performance with vs. without our attack on the DL3DV dataset. GB denotes the gradient-based variant, and GF denotes the gradient-free variant. Percentages measure the degree of performance degradation.

Victim Models	PSNR ↓	SSIM ↓	LPIPS ↑	CLIP ↓	DINO ↓
DepthSplat	22.21	0.785	0.165	0.963	0.909
DepthSplat+Ours (GB)	14.38(−35.3%)	0.546(−30.4%)	0.448(+171.5%)	0.837(−13.1%)	0.738(−18.8%)
DepthSplat+Ours (GF)	16.59(−25.3%)	0.588(−25.1%)	0.396(+140.0%)	0.851(−11.6%)	0.798(−12.2%)
NoPoSplat	21.91	0.745	0.173	0.960	0.903
NoPoSplat+Ours (GB)	19.80(−9.6%)	0.582(−21.9%)	0.338(+95.4%)	0.886(−7.7%)	0.819(−9.3%)
NoPoSplat+Ours (GF)	17.17(−21.6%)	0.425(−43.0%)	0.439(+153.8%)	0.858(−10.6%)	0.722(−20.0%)
AnySplat	19.34	0.639	0.285	0.950	0.923
AnySplat+Ours (GB)	15.47(−20.0%)	0.467(−26.9%)	0.502(+76.1%)	0.867(−8.7%)	0.815(−11.7%)
AnySplat+Ours (GF)	17.52(−9.4%)	0.497(−22.2%)	0.481(+68.8%)	0.876(−7.8%)	0.839(−9.1%)

4.1 Experiment Settings

Datasets & Baselines. We use two benchmark datasets for feed-forward 3DGS, including Re10K(Zhou et al., 2018) and DL3DV(Ling et al., 2024). These two datasets cover a diverse set of indoor and outdoor scenes. To better validate the effectiveness of our AdvSplat, we focus on feed-forward 3DGS models with sufficiently strong performance. Specifically, we select three representative state-of-the-art models, including one pose-known model, DepthSplat(Xu et al., 2025), and two pose-free models NoPoSplat(Ye et al., 2024a) and AnySplat(Jiang et al., 2025).

Implementation Details. We run all models and evaluate our method on a server equipped with four NVIDIA RTX A6000 GPUs. Other details can be found in Appendix A.

Metrics. We adopt standard image quality metrics, including peak signal-to-noise ratio (PSNR), structural similarity (SSIM)(Wang et al., 2004), and learned perceptual image patch similarity (LPIPS)(Zhang et al., 2018). We additionally include the DINO(Zhang et al., 2022) similarity and the CLIP(Radford et al., 2021) similarity to measure semantic consistency. All metrics are computed between the rendered images and the ground-truth images.

4.2 Experiment Results

Attack Effectiveness of AdvSplat (RQ1). We provide a comprehensive evaluation through both quantitative and qualitative results. As shown in Table 1 and Table 5, our attack method significantly degrades the performance of all victim models across all metrics. Notice that, since our objective is the opposite of the reconstruction goal, the interpretation of all metrics is correspondingly reversed: lower is better for PSNR, SSIM, CLIP, and DINO, while higher is better for LPIPS. In general, when gradient estimation is sufficiently accurate, gradient-based methods outperform gradient-free ones, since moving along the gradient direction increases the loss more rapidly. However, in NoPoSplat, the gradient-free variant outperforms the gradient-based variant because NoPoSplat is trained on a mixture of two datasets, resulting in a larger model capacity and making its gradients harder to estimate accurately. Figure 3 visualizes the input images and the corresponding rendered outputs. As shown, the perturbations on adversarial inputs are nearly imperceptible to human eyes. This is because the ℓ_∞ constraint is strictly set to 8/255. Even in the worst case that each pixel is perturbed by at most 8/255, the PSNR between the adversarial and clean images still remains above

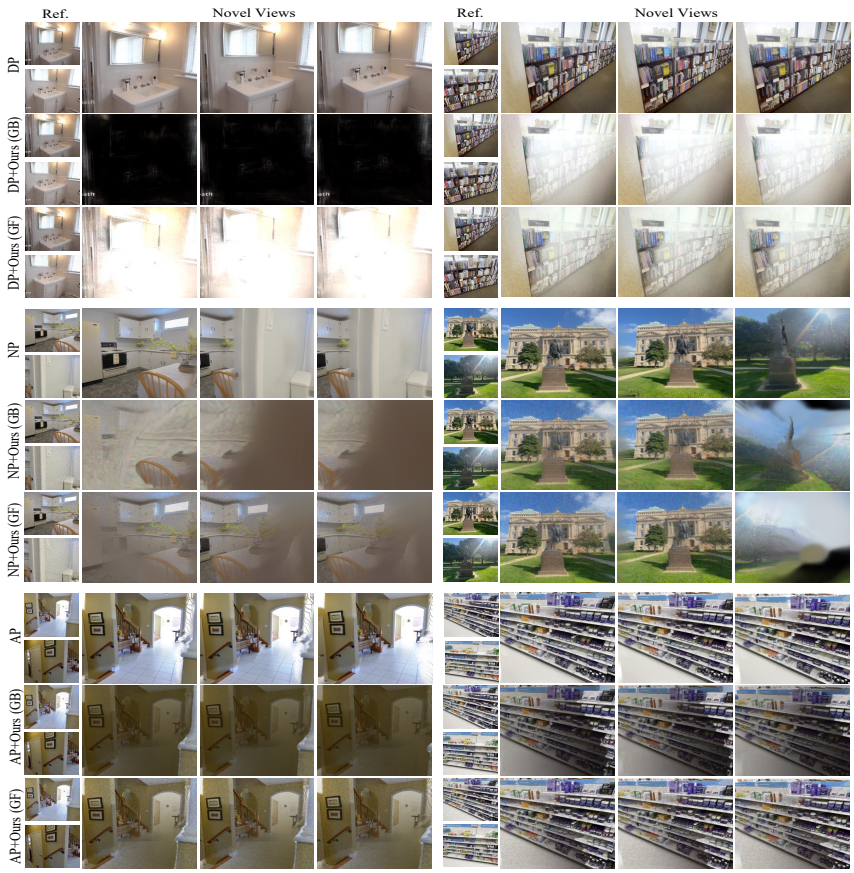


Figure 3: Qualitative results on RE10K and DL3DV ($\epsilon = 8/255$). DP denotes DepthSplat, NP denotes NoPoSplat, and AP denotes AnySplat. GB denotes the gradient-based variant, and GF denotes the gradient-free variant. All baselines use clean inputs, whereas our method uses the optimized adversarial examples as inputs.

30 dB, showing closeness between them. Despite the small perturbation, the quality of the rendered images after attack degrades dramatically, and some results are even completely corrupted. This indicates that, even in the black-box setting, AdvSplat can generate sufficiently strong adversarial examples, exposing the vulnerability of existing feed-forward 3DGS models under such attacks.

Efficiency of DCT-based Strategy (RQ2). Using DCT to sample noise from the low-frequency components in the frequency domain is a key strategy of our method, which lowers the dimensionality of the optimization space. Fewer samples translate into fewer queries to the feed-forward 3DGS model, which not only reduces optimization time but also lowers the cost of querying commercial models. Figure 4 shows the loss (in Equation (5), computed as the average across all victim models and datasets) curves with and without DCT. "no DCT" means sampling noise directly in the full pixel space. For fairness, we use the same number of samples. As shown, for both the gradient-based and gradient-free variants, incorporating DCT and sampling noise in the low-frequency components yields a higher loss at the same query step once convergence is reached, validating the improved efficiency of DCT.

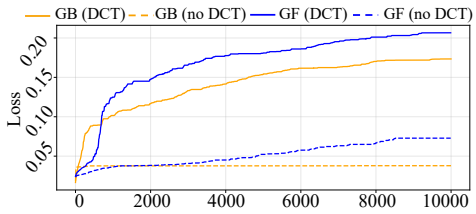


Figure 4: Loss comparison with and without DCT. GB denotes the gradient-based variant, and GF denotes the gradient-free variant.

Attack Effectiveness Under Various Attack Strengths (RQ3). We vary the attack strength, i.e., the ℓ_∞ perturbation budget, to further evaluate the effectiveness of our attack. We consider

Table 3: Victim model performance on the RE10K and DL3DV datasets (We select one scene per dataset and use DepthSplat) under different attack strengths (ϵ values). GB denotes the gradient-based variant, and GF denotes the gradient-free variant.

ϵ	Method	RE10K					DL3DV				
		PSNR ↓	SSIM ↓	LPIPS ↑	CLIP ↓	DINO ↓	PSNR ↓	SSIM ↓	LPIPS ↑	CLIP ↓	DINO ↓
2/255	GB	23.65	0.777	0.350	0.937	0.933	15.61	0.617	0.246	0.956	0.957
	GF	23.59	0.772	0.321	0.942	0.946	20.23	0.698	0.231	0.967	0.978
4/255	GB	5.96	0.195	0.504	0.711	0.571	9.71	0.455	0.378	0.907	0.898
	GF	9.914	0.464	0.363	0.874	0.802	14.08	0.561	0.292	0.937	0.954
8/255	GB	4.38	0.050	0.663	0.554	0.060	8.09	0.384	0.493	0.858	0.726
	GF	8.39	0.608	0.548	0.701	0.908	10.82	0.456	0.406	0.854	0.847
16/255	GB	7.20	0.325	0.851	0.589	0.019	8.01	0.343	0.562	0.800	0.724
	GF	7.54	0.522	0.649	0.664	0.758	9.49	0.373	0.521	0.820	0.802

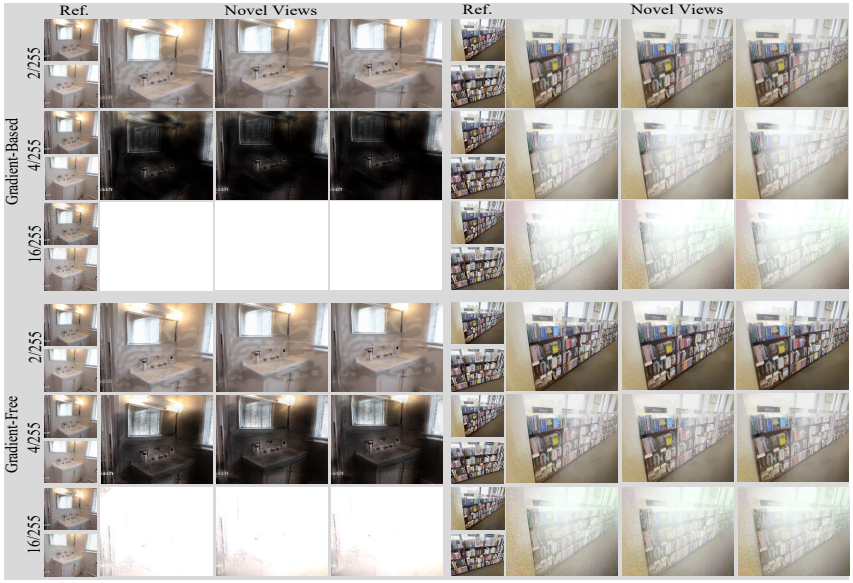


Figure 5: Qualitative results under different attack strengths. The gray background is used to better visualize the all-white rendered images.

four attack strengths and evaluate both the gradient-based and gradient-free variants on the two datasets. As shown in Table 3, as the attack strength increases, all metrics exhibit a consistent trend of deterioration. One exception is that on RE10K, for the gradient-based variant, PSNR and SSIM actually improve when the perturbation budget increases from 8/255 to 16/255. However, this does not imply that a stronger attack leads to better rendering quality. As observed in Figure 3, when $\epsilon = 8/255$, the rendered images contain many black artifacts. When $\epsilon = 16/255$ (in Figure 5), the renderings collapse to nearly all-white images. Since the clean inputs also contain many near-white pixels, this failure case can misleadingly inflate PSNR and SSIM, leading to the observed increase. However, by inspecting the rendered images and considering LPIPS together with the CLIP and DINO similarities, we find that the renderings under $\epsilon = 16/255$ are still of worse quality.

5 Conclusion

In this paper, we present the first study on the unexplored robustness issue of feed-forward 3DGS models, and propose AdvSplat, an efficient attack method for them. AdvSplat operates in a black-box setting, requiring only access to inputs and outputs, and performs attacks by sampling perturbations through a frequency-domain parameterization. Our results reveal the vulnerability of feed-forward 3DGS models and highlight new challenges for their commercial deployment. We hope this work will draw the community's attention and inspire future research on improving the robustness and defenses of these models.

AI USE STATEMENT

Generative AI tools were used solely to improve the clarity, grammar, and presentation of this manuscript. They were not used to formulate the research problem, design the proposed attacks, implement the experimental code, or generate the reported experimental results. All AI-assisted revisions were carefully reviewed and edited by the authors. We take full responsibility for the accuracy and integrity of the final manuscript, including its scientific claims, experimental findings, and associated artifacts.

ETHICS STATEMENT

This work does not involve human participants. ADVSPLAT investigates adversarial vulnerabilities in feed-forward 3D Gaussian Splatting models through controlled experiments on existing benchmark datasets. Our goal is to raise awareness of these vulnerabilities and support future research on robust reconstruction and effective defenses. We acknowledge the dual-use nature of adversarial attack research: the proposed techniques could be misused to degrade reconstruction quality or disrupt downstream applications. We advocate responsible use of these methods for robustness evaluation and defense development and discourage their use against systems without authorization.

REPRODUCIBILITY STATEMENT

The main paper describes the ADVSPLAT framework, including the threat models, attack objectives, frequency-domain parameterization, and experimental settings. We report the evaluated models, datasets, perturbation constraints, hyperparameters, and evaluation metrics, while the appendices provide detailed algorithms and additional quantitative and qualitative results. These materials support reproduction and independent assessment of our findings. Upon acceptance, we will publicly release the source code, configuration files, and evaluation scripts.

REFERENCES

Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *European conference on computer vision*, pp. 484–501. Springer, 2020.

Sherwin Bahmani, Tianchang Shen, Jiawei Ren, Jiahui Huang, Yifeng Jiang, Haithem Turki, Andrea Tagliasacchi, David B Lindell, Zan Gojcic, Sanja Fidler, et al. Lyra: Generative 3d scene reconstruction via video diffusion model self-distillation. *arXiv preprint arXiv:2509.19296*, 2025.

Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

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.

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. Springer, 2024.

Jiahua Dong and Yu-Xiong Wang. 3dgs-drag: Dragging gaussians for intuitive point-based 3d editing. *arXiv preprint arXiv:2601.07963*, 2026.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Chuan Guo, Jacob Gardner, Yurong You, Andrew Gordon Wilson, and Kilian Weinberger. Simple black-box adversarial attacks. In *International conference on machine learning*, pp. 2484–2493. PMLR, 2019.

Nikolaus Hansen. The cma evolution strategy: A tutorial. *arXiv preprint arXiv:1604.00772*, 2016.

Georg Hess, Carl Lindström, Maryam Fatemi, Christoffer Petersson, and Lennart Svensson. Splatad: Real-time lidar and camera rendering with 3d gaussian splatting for autonomous driving. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 11982–11992, 2025.

Team Hunyuan3D, Shuhui Yang, Mingxin Yang, Yifei Feng, Xin Huang, Sheng Zhang, Zebin He, Di Luo, Haolin Liu, Yunfei Zhao, et al. Hunyuan3d 2.1: From images to high-fidelity 3d assets with production-ready pbr material. *arXiv preprint arXiv:2506.15442*, 2025.

Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *International conference on machine learning*, pp. 2137–2146. PMLR, 2018.

Youngdong Jang, Hyunje Park, Feng Yang, Heeju Ko, Euijin Choo, and Sangpil Kim. 3d-gsw: 3d gaussian splatting for robust watermarking. *arXiv preprint arXiv:2409.13222*, 2024.

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 (TOG)*, 44(6):1–16, 2025.

Ying Jiang, Chang Yu, Tianyi Xie, Xuan Li, Yutao Feng, Huamin Wang, Minchen Li, Henry Lau, Feng Gao, Yin Yang, et al. Vr-gs: A physical dynamics-aware interactive gaussian splatting system in virtual reality. In *ACM SIGGRAPH 2024 conference papers*, pp. 1–1, 2024.

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 Computer Vision and Pattern Recognition Conference*, pp. 22012–22022, 2025.

Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. Splatam: Splat track & map 3d gaussians for dense rgb-d slam. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 21357–21366, 2024.

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

Mustafa Khan, Hamidreza Fazlali, Dhruv Sharma, Tongtong Cao, Dongfeng Bai, Yuan Ren, and Bingbing Liu. Autosplat: Constrained gaussian splatting for autonomous driving scene reconstruction. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8315–8321. IEEE, 2025.

Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22160–22169, 2024.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

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, 2025.

Guanxing Lu, Shiyi Zhang, Ziwei Wang, Changliu Liu, Jiwen Lu, and Yansong Tang. Manigaussian: Dynamic gaussian splatting for multi-task robotic manipulation. In *European Conference on Computer Vision*, pp. 349–366. Springer, 2024a.

Jiahao Lu, Yifan Zhang, Qiuhong Shen, Xinchao Wang, and Shuicheng Yan. Poison-splat: Computation cost attack on 3d gaussian splatting. *arXiv preprint arXiv:2410.08190*, 2024b.

Yiren Lu, Yunlai Zhou, Yiran Qiao, Chaoda Song, Tuo Liang, Jing Ma, Huan Wang, and Yu Yin. Segment then splat: Unified 3d open-vocabulary segmentation via gaussian splatting. *arXiv preprint arXiv:2503.22204*, 2025.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. Gaussian splatting slam. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 18039–18048, 2024.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582, 2016.

Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3d language gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20051–20060, 2024.

Yansong Qu, Dian Chen, Xinyang Li, Xiaofan Li, Shengchuan Zhang, Liujuan Cao, and Rongrong Ji. Drag your gaussian: Effective drag-based editing with score distillation for 3d gaussian splatting. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–12, 2025.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Jin-Chuan Shi, Miao Wang, Hao-Bin Duan, and Shao-Hua Guan. Language embedded 3d gaussians for open-vocabulary scene understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5333–5343, 2024.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Stanislaw Szymanowicz, Chrisitian Rupprecht, and Andrea Vedaldi. Splatter image: Ultra-fast single-view 3d reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10208–10217, 2024.

Xuechang Tu, Bernhard Kerbl, and Fernando De la Torre. Fast and robust 3d gaussian splatting for virtual reality. In *SIGGRAPH Asia 2024 Posters*, pp. 1–3. 2024.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 5294–5306, 2025.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Christopher Wewer, Kevin Raj, Eddy Ilg, Bernt Schiele, and Jan Eric Lenssen. latentsplat: Autoencoding variational gaussians for fast generalizable 3d reconstruction. In *European conference on computer vision*, pp. 456–473. Springer, 2024.

Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 20310–20320, 2024a.

Jay Zhangjie Wu, Yuxuan Zhang, Haithem Turki, Xuanchi Ren, Jun Gao, Mike Zheng Shou, Sanja Fidler, Zan Gojcic, and Huan Ling. Difix3d+: Improving 3d reconstructions with single-step diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26024–26035, 2025a.

Shuang Wu, Youtian Lin, Feihu Zhang, Yifei Zeng, Yikang Yang, Yajie Bao, Jiachen Qian, Siyu Zhu, Xun Cao, Philip Torr, et al. Direct3d-s2: Gigascale 3d generation made easy with spatial sparse attention. *arXiv preprint arXiv:2505.17412*, 2025b.

Yanmin Wu, Jiarui Meng, Haijie Li, Chenming Wu, Yahao Shi, Xinhua Cheng, Chen Zhao, Haocheng Feng, Errui Ding, Jingdong Wang, et al. Opengaussian: Towards point-level 3d gaussian-based open vocabulary understanding. *Advances in Neural Information Processing Systems*, 37:19114–19138, 2024b.

Jianfeng Xiang, Xiaoxue Chen, Sicheng Xu, Ruicheng Wang, Zelong Lv, Yu Deng, Hongyuan Zhu, Yue Dong, Hao Zhao, Nicholas Jing Yuan, et al. Native and compact structured latents for 3d generation. *arXiv preprint arXiv:2512.14692*, 2025a.

Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 21469–21480, 2025b.

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.

Ziyang Yan, Lei Li, Yihua Shao, Siyu Chen, Zongkai Wu, Jenq-Neng Hwang, Hao Zhao, and Fabio Remondino. 3dsceneeditor: Controllable 3d scene editing with gaussian splatting. *arXiv preprint arXiv:2412.01583*, 2024.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *Advances in Neural Information Processing Systems*, 37:21875–21911, 2024a.

Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. *arXiv preprint arXiv:2310.10642*, 2023.

Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 20331–20341, 2024b.

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. *arXiv preprint arXiv:2410.24207*, 2024a.

Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3d scenes. In *European conference on computer vision*, pp. 162–179. Springer, 2024b.

Abdurrahman Zeybey, Mehmet Ergezer, and Tommy Nguyen. Gaussian splatting under attack: Investigating adversarial noise in 3d objects. *arXiv preprint arXiv:2412.02803*, 2024.

Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022.

Jiahui Zhang, Yuelei Li, Anpei Chen, Muyu Xu, Kunhao Liu, Jianyuan Wang, Xiao-Xiao Long, Hanxue Liang, Zexiang Xu, Hao Su, et al. Advances in feed-forward 3d reconstruction and view synthesis: A survey. *arXiv preprint arXiv:2507.14501*, 2025.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018.

Xiaoyu Zhou, Zhiwei Lin, Xiaojun Shan, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. Drivinggaussian: Composite gaussian splatting for surrounding dynamic autonomous driving scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 21634–21643, 2024.

Xueyan Zou, Yuchen Song, Ri-Zhao Qiu, Xuanbin Peng, Jianglong Ye, Sifei Liu, and Xiaolong Wang. 3d-spatial multimodal memory. In *The Thirteenth International Conference on Learning Representations*, 2025.

A Implentation Details

Since we do not know the original training loss used by each black-box model, for fairness and generality, we consistently adopt Equation (5) with $\lambda = 0.05$. We set the image block size $n = 8$ and low frequency block size $s = 3$; accordingly, the number of blocks J is computed as $H \times W/n^2$. We set the number of attack iterations to 10,000, the step size η to 2/255, while constraining the perturbation magnitude to $\ell_\infty = 8/255$. For all baselines, the number of input images N is equal to 2. M and $\mathcal{B}$ are both equal to 40. Notably, to avoid frequent GPU–CPU I/O and enable more efficient computation, **we implement all AdvSplat operations on CUDA**.

B Algorithm Details

In the main paper, we present only the core components of our method. Therefore, in the appendix, we provide pseudocode for both the gradient-based and gradient-free variants to help readers better understand the algorithm. Moreover, the gradient-free variant described in the main paper is a simplified version; in this section, we provide the full version in detail.

Algorithm 1 Gradient-Based Frequency-Domain Black-Box Attack

Input: Original image $\boldsymbol{I} \in \mathbb{R}^{H\times W\times 3}$, victim feed-forward 3DGS model $f(\cdot)$, adversarial loss $\mathcal{L}$, DCT block size n, low-frequency sub-block size s, PGD iterations T, step size η, L_∞ perturbation bound ϵ, number of NES noise samples M, NES search variance σ, zero-padding operator $\text{pad}_{s\to n}(\cdot)$ mapping $\mathbb{R}^{s\times s\times 3} \to \mathbb{R}^{n\times n\times 3}$, element-wise sign function $\text{sign}(\cdot)$.
Output: Adversarial example $\boldsymbol{I}' \in \mathbb{R}^{H\times W\times 3}$

Initialize adversarial example: $\boldsymbol{I}' \leftarrow \boldsymbol{I}$
for $t = 0$ **to** $T-1$ **do**
  Divide $\boldsymbol{I}'$ into J blocks of equal size $n \times n$: $\{\boldsymbol{I}_b'^0, \boldsymbol{I}_b'^1, \ldots, \boldsymbol{I}_b'^{J-1}\}$
  Extract current DCT coefficients: $\boldsymbol{F}^j = \boldsymbol{C}^j \boldsymbol{I}_b'^j \boldsymbol{C}^{j\top}, \quad \forall j \in \{0, \ldots, J-1\}$
  Initialize gradient accumulators: $\mathbf{g}^j = \mathbf{0} \in \mathbb{R}^{s\times s\times 3}, \quad \forall j$
  for $m = 0$ **to** $M-1$ **do**
    Sample standard Gaussian noise in frequency domain: $\mathbf{u}_m^j \sim \mathcal{N}(0, \mathbf{I}_{s\times s\times 3}), \quad \forall j$
    Construct perturbed low-frequency coefficients:
    $\boldsymbol{F}_{ss,m}^{j,+} = \boldsymbol{F}_{0:s-1,0:s-1}^j + \sigma \mathbf{u}_m^j$
    $\boldsymbol{F}_{ss,m}^{j,-} = \boldsymbol{F}_{0:s-1,0:s-1}^j - \sigma \mathbf{u}_m^j$
    Apply iDCT and concatenate to generate probe images $\boldsymbol{I}_m'^+, \boldsymbol{I}_m'^-$ (Eq. (10))
    Obtain rendered images: $\boldsymbol{I}_{r,(m)}'^+ = f(\boldsymbol{I}_m'^+),\ \boldsymbol{I}_{r,(m)}'^- = f(\boldsymbol{I}_m'^-)$
    Accumulate gradients: $\mathbf{g}^j \leftarrow \mathbf{g}^j + [\mathcal{L}(\boldsymbol{I}_m'^+, \boldsymbol{I}_{r,(m)}'^+) - \mathcal{L}(\boldsymbol{I}_m'^-, \boldsymbol{I}_{r,(m)}'^-)]\mathbf{u}_m^j$
  end for
  Compute spatial block gradients: $\mathbf{g}_{img}^j = \boldsymbol{C}^{j\top} \text{pad}_{s\to n}(\mathbf{g}^j) \boldsymbol{C}^j, \quad \forall j$
  Concatenate all spatial blocks $\mathbf{g}_{img}^j$ to form global spatial gradient $\mathbf{G} \in \mathbb{R}^{H\times W\times 3}$
  Estimate global spatial gradient: $\nabla_{\boldsymbol{I}'}\mathcal{L} \approx \frac{1}{2M\sigma}\mathbf{G}$
  Sign gradient update in pixel space: $\boldsymbol{I}' \leftarrow \boldsymbol{I}' + \eta \cdot \text{sign}(\nabla_{\boldsymbol{I}'}\mathcal{L})$
  Spatial domain L_∞ projection around clean image: $\boldsymbol{I}' \leftarrow \text{Clip}(\boldsymbol{I}', \boldsymbol{I} - \epsilon, \boldsymbol{I} + \epsilon)$
  Ensure valid pixel range: $\boldsymbol{I}' \leftarrow \text{Clip}(\boldsymbol{I}', 0, 1)$
end for
return $\boldsymbol{I}'$

C Additional Results of White-Box Attack

We provide additional quantitative and qualitative results for the white-box setting in this section. All hyperparameter settings follow the main paper, except that the number of attack iterations is set to 50. Since white-box attacks assume access to model weights and gradients, the resulting renderings

Algorithm 2 Gradient-Free Frequency-Domain Black-Box Attack

Input: Original image $\boldsymbol{I} \in \mathbb{R}^{H\times W\times 3}$, victim model $f(\cdot)$, adversarial loss $\mathcal{L}$, DCT block size n, low-frequency subset size s, total image blocks J, iterations T, population size $\mathcal{B}$, initial step size σ_{init}, L_∞ bound ϵ, zero-padding operator $\text{pad}_{s\to n}(\cdot)$.

Output: Adversarial example $\boldsymbol{I}' \in \mathbb{R}^{H\times W\times 3}$

Let $D = s \times s \times 3$ be the sub-block dimension in the frequency domain, and $D_{total} = J \times D$ be the global dimension

Initialize block distributions $\forall j \in \{0, \dots, J-1\}$: mean $\boldsymbol{a}^j \leftarrow \mathbf{0} \in \mathbb{R}^D$, diagonal covariance $\boldsymbol{V}^j \leftarrow \mathbf{1} \in \mathbb{R}^D$

Initialize global step size $\sigma \leftarrow \sigma_{init}$, and block evolution paths $\boldsymbol{p}_c^j \leftarrow \mathbf{0} \in \mathbb{R}^D, \boldsymbol{p}_s^j \leftarrow \mathbf{0} \in \mathbb{R}^D, \forall j$

Set $\mu = \mathcal{B}/2$. Calculate weights: $w'_\beta = \ln(\mu + 0.5) - \ln(\beta + 1)$ and normalize $w_\beta = \frac{w'_\beta}{\sum_{i=0}^{\mu-1} w'_i}$

Compute variance effective selection mass: $\mu_{eff} = 1/\sum_{\beta=0}^{\mu-1} w_\beta^2$

Evolution path rates: $c_c = \frac{4+\mu_{eff}/D_{total}}{D_{total}+4+2\mu_{eff}/D_{total}}$, $c_s = \frac{\mu_{eff}+2}{D_{total}+\mu_{eff}+5}$

Covariance update rates: $c_1 = \frac{2}{(D_{total}+1.3)^2+\mu_{eff}}$, $c_\mu = \min\left(1-c_1, 2\frac{\mu_{eff}-2+1/\mu_{eff}}{(D_{total}+2)^2+\mu_{eff}}\right)$

Step-size damping $d_\sigma = 1 + 2\max\left(0, \sqrt{\frac{\mu_{eff}-1}{D_{total}+1}} - 1\right) + c_s$, expected norm $\chi_N \approx \sqrt{D_{total}}\left(1 - \frac{1}{4D_{total}} + \frac{1}{21D_{total}^2}\right)$

for $t = 0$ **to** $T-1$ **do**

  for $\beta = 0$ **to** $\mathcal{B}-1$ **do**

    Sample noise and generate candidate sub-blocks: $\boldsymbol{z}_\beta^j \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_D)$ and $\boldsymbol{\delta}_\beta^j = \boldsymbol{a}^j + \sigma\sqrt{\boldsymbol{V}^j} \odot \boldsymbol{z}_\beta^j, \ \forall j$

    Compute spatial blocks via padding & iDCT: $\mathbf{p}_\beta^j = \boldsymbol{C}^{j\top}\text{pad}_{s\to n}(\boldsymbol{\delta}_\beta^j)\boldsymbol{C}^j, \ \forall j$

    Concatenate blocks $\mathbf{p}_\beta^j$ to form global spatial perturbation $\mathbf{P}_\beta$

    Apply L_∞ and valid pixel range clipping:

    $\boldsymbol{I}'_\beta \leftarrow \text{Clip}(\text{Clip}(\boldsymbol{I} + \mathbf{P}_\beta, \boldsymbol{I} - \epsilon, \boldsymbol{I} + \epsilon), 0, 1)$

    Compute fitness: $L_\beta = \mathcal{L}(\boldsymbol{I}'_\beta, f(\boldsymbol{I}'_\beta))$

  end for

  Sort descending by fitness: $L_{(0)} \geq L_{(1)} \geq \cdots \geq L_{(\mathcal{B}-1)}$ and select top μ candidates

  Update per-block means: $\boldsymbol{a}_{old}^j \leftarrow \boldsymbol{a}^j$ and $\boldsymbol{a}^j \leftarrow \sum_{\beta=0}^{\mu-1} w_\beta \boldsymbol{\delta}_\beta^j, \ \forall j$

  Compute normalized mean differences: $\boldsymbol{y}^j \leftarrow (\boldsymbol{a}^j - \boldsymbol{a}_{old}^j)/\sigma, \ \forall j$

  Update conjugate paths: $\boldsymbol{p}_s^j \leftarrow (1-c_s)\boldsymbol{p}_s^j + \sqrt{c_s(2-c_s)\mu_{eff}}(\boldsymbol{V}^j)^{-1/2} \odot \boldsymbol{y}^j, \ \forall j$

  Evaluate global Heaviside function $h_\sigma \in \{0, 1\}$ based on the concatenated global path norm $\|\boldsymbol{p}_s\|_{global}$ and χ_N

  Update covariance paths: $\boldsymbol{p}_c^j \leftarrow (1-c_c)\boldsymbol{p}_c^j + h_\sigma\sqrt{c_c(2-c_c)\mu_{eff}}\boldsymbol{y}^j, \ \forall j$

  Update global step size: $\sigma \leftarrow \sigma \exp\left(\frac{c_s}{d_\sigma}\left(\frac{\|\boldsymbol{p}_s\|_{global}}{\chi_N} - 1\right)\right)$

  Update diagonal covariance vectors $\boldsymbol{V}^j, \ \forall j$:

  $\boldsymbol{V}^j \leftarrow (1-c_1-c_\mu)\boldsymbol{V}^j + c_1\left((\boldsymbol{p}_c^j)^{\odot 2} + (1-h_\sigma)c_c(2-c_c)\boldsymbol{V}^j\right) + c_\mu\sum_{\beta=0}^{\mu-1} w_\beta\left((\boldsymbol{V}^j)^{-1/2} \odot \frac{\boldsymbol{\delta}_\beta^j - \boldsymbol{a}_{old}^j}{\sigma}\right)^{\odot 2}$

end for

Extract converged spatial blocks via padding & iDCT: $\mathbf{p}_{final}^j = \boldsymbol{C}^{j\top}\text{pad}_{s\to n}(\boldsymbol{a}^j)\boldsymbol{C}^j, \ \forall j$

Concatenate blocks $\mathbf{p}_{final}^j$ to form $\mathbf{P}_{final}$

return $\boldsymbol{I}' \leftarrow \text{Clip}(\text{Clip}(\boldsymbol{I} + \mathbf{P}_{final}, \boldsymbol{I} - \epsilon, \boldsymbol{I} + \epsilon), 0, 1)$

are typically worse than the corresponding black-box results reported in the main paper. For pose-free models such as NoPoSplat and AnySplat, evaluating reconstructions requires not only checking for large-scale artifacts but also examining whether the scene coordinate system has shifted. As shown in the second row of the right half of Figure 6, under the target pose, the rendered viewpoint can deviate substantially from the ground-truth viewpoint.

Table 4: Quantitative comparison of victim model performance with vs. without our white-box attack on the RE10K dataset. Percentages measure the degree of performance degradation.

Victim Models	PSNR ↓	SSIM ↓	LPIPS ↑	CLIP ↓	DINO ↓
DepthSplat	21.09	0.710	0.228	0.956	0.930
DepthSplat+Ours	6.82(−67.7%)	0.223(−68.6%)	0.594(+160.5%)	0.677(−29.2%)	0.370(−60.2%)
NoPoSplat	22.50	0.781	0.165	0.957	0.938
NoPoSplat+Ours	8.82(−60.8%)	0.268(−65.7%)	0.692(+319.4%)	0.780(−18.5%)	0.598(−36.2%)
AnySplat	18.94	0.672	0.271	0.928	0.938
AnySplat+Ours	10.20(−46.1%)	0.422(−37.2%)	0.658(+142.8%)	0.763(−17.8%)	0.450(−52.0%)

Table 5: Quantitative comparison of victim model performance with vs. without our white-box attack on the DL3DV dataset. Percentages measure the degree of performance degradation.

Victim Models	PSNR ↓	SSIM ↓	LPIPS ↑	CLIP ↓	DINO ↓
DepthSplat	22.21	0.785	0.165	0.963	0.909
DepthSplat+Ours	6.54(−70.6%)	0.202(−74.3%)	0.625(+278.8%)	0.665(−30.9%)	0.205(−77.4%)
NoPoSplat	21.91	0.745	0.173	0.960	0.903
NoPoSplat+Ours	11.44(−47.8%)	0.249(−66.6%)	0.634(+266.5%)	0.787(−18.0%)	0.527(−41.6%)
AnySplat	19.34	0.639	0.285	0.950	0.923
AnySplat+Ours	14.43(−25.4%)	0.427(−33.2%)	0.523(+83.5%)	0.860(−9.5%)	0.849(−8.0%)

D Cross Validation of Transfer-Based Attack

In this section, we empirically demonstrate that transfer-based attacks are infeasible for feed-forward 3DGS models. We only use the RE10K dataset because, on DL3DV, the three victim models adopt different preprocessing pipelines; naively unifying image sizes via resizing and cropping would substantially shift the data distribution, making a fair comparison impossible. We perform an exhaustive cross-evaluation over all permutations, i.e. using one model as the victim and the other two as surrogate models. Adversarial examples are generated via white-box attacks on the surrogate models, with all other settings kept unchanged. As shown in Table 6, while transfer-based attacks do degrade reconstruction quality for all victim models, the magnitude is far smaller than that of query-based attacks. Figure 7 further shows that, visually, the impact of transfer-based attacks on the rendered images is not pronounced. Overall, transferability does not work well for feed-forward 3DGS models.



Figure 6: Qualitative results for white-box attack on RE10K and DL3DV ($\epsilon = 8/255$). The left half shows scenes from RE10K, while the right half shows scenes from DL3DV. The clean reconstruction results are shown in Figure 3.

Table 6: Quantitative comparison of transfer attack performance on RE10K. Adversarial examples are generated on a surrogate model and transferred to the victim model. "None" means using the clean data.

Victim Model	Surrogate Model	PSNR ↓	SSIM ↓	LPIPS ↑	CLIP ↓	DINO ↓
DepthSplat	None	21.09	0.710	0.228	0.956	0.930
	NoPoSplat	20.32	0.653	0.406	0.927	0.889
	AnySplat	20.10	0.658	0.294	0.947	0.905
NoPoSplat	None	22.50	0.781	0.165	0.957	0.938
	DepthSplat	22.36	0.733	0.310	0.934	0.894
	AnySplat	22.02	0.743	0.265	0.946	0.911
AnySplat	None	18.94	0.672	0.271	0.928	0.938
	DepthSplat	18.15	0.580	0.440	0.894	0.894
	NoPoSplat	17.86	0.572	0.454	0.889	0.864

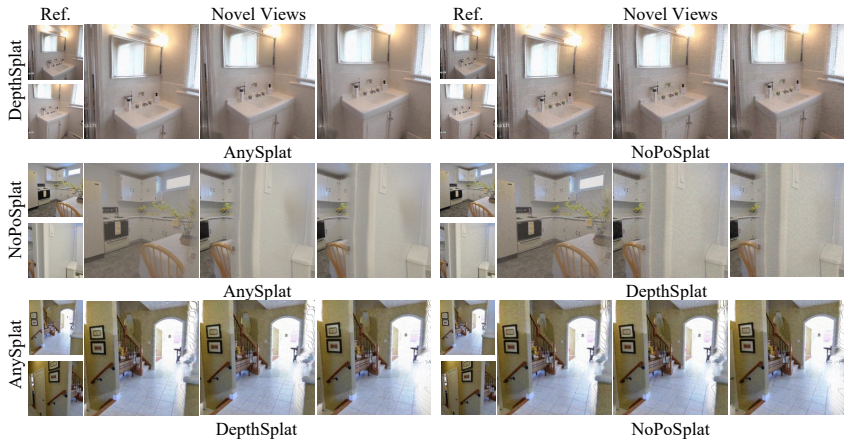


Figure 7: Qualitative results for transfer attack on RE10K ($\epsilon = 8/255$). Each row corresponds to a victim model, and the surrogate model is indicated in each column. The clean reconstruction results are shown in Figure 3.