ComplicitSplat: Downstream Models are Vulnerable to Blackbox Attacks by 3D Gaussian Splat Camouflages

Matthew Hull[1] Haoyang Yang[1] Pratham Mehta[1] Mansi Phute[1]
Aeree Cho[1] Haoran Wang[1] Matthew Lau[1] Wenke Lee[1]
Willian Lunardi[2] Martin Andreoni[2] Duen Horng Chau[1]

[1]Georgia Institute of Technology, Atlanta, GA, USA [2]Technology Innovation Institute, Abu Dhabi, UAE
matthewhull@gatech.edu

Abstract

As 3D Gaussian Splatting (3DGS) gains rapid adoption in safety-critical tasks for efficient novel-view synthesis from static images, how might an adversary tamper images to cause harm? We introduce ComplicitSplat, the first attack to embed 2D generative edits as viewpoint-localized camouflage in 3DGS, colors and textures that change with viewing angle so that adversarial content is concealed in scene objects and visible only from attacker-chosen viewpoints, without access to downstream model weights, architecture, or the scene's training images. Our extensive experiments show that ComplicitSplat generalizes across real-world captures of physical objects and synthetic indoor and outdoor scenes, successfully attacking popular single-stage, multi-stage, and transformer-based detectors while largely evading recent state-of-the-art 3DGS defenses, maintaining 97.5% of its attack effectiveness against *DEGauss* and 93.9% against *DefenseSplat*. To our knowledge, this is the first black-box attack on downstream object detectors via tampered 3DGS scenes, exposing a novel safety risk for applications like autonomous navigation and other mission-critical robotic systems. To enable reproducibility of our results, we have prepared our code for immediate release and included it in Supplementary Material.

1 Introduction

3D Gaussian Splatting (3DGS) [17] has rapidly gained popularity in safety-critical applications due to its efficiency in novel-view synthesis from a set of static images resulting in real-time 3D rendering of complex scenes, outperforming traditional methods like Neural Radiance Fields (NeRFs) [27]. The advantages of 3DGS have led to growing interest in safety-critical domains such as autonomous driving [10, 22, 42], airborne navigation [31], overhead (BEV) navigation [20], and grasping [32, 38], where rapid data generation and accurate sim2real transfer are essential.

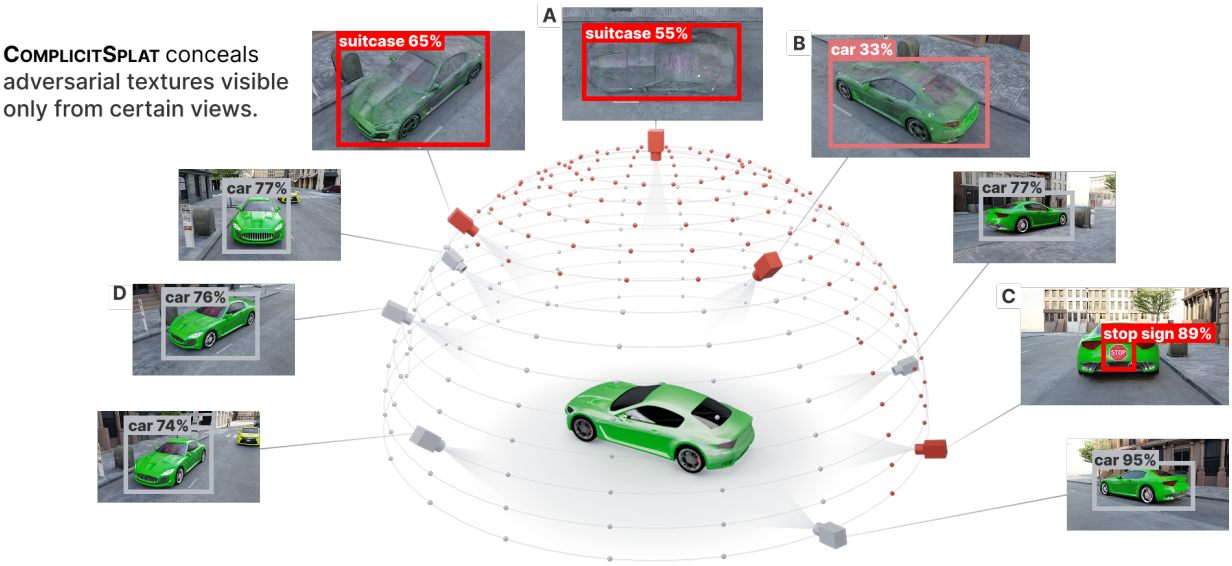


Fig. 1: **ComplicitSplat** conceals *multiple* adversarial cloaked textures in 3DGS scenes using Spherical Harmonics, causing the 3DGS representation of the car to become adversarial at different view points (red dots). For example, (A) when viewed from the top, the car appears as a suitcase, (B) "car" detection confidence decreases, (C) and when viewed directly from behind, displays a "stop sign."

Despite the increasing adoption of 3DGS, have the security vulnerabilities in the 3DGS scene representation been adequately considered? Currently, 3DGS scenes are widely available for download from various sources[1,2], but associated source images are often unavailable.

Given these challenges of obtaining 3DGS without source images, what harms could an attacker cause if they fine-tune a released 3DGS scene using generated training images? We study this digital threat surface, where the attacker tampers with a distributed 3DGS asset that downstream pipelines ingest and render. This exposes the risk of "quiet-tampering", *i.e.*, not being able to detect that a released 3DGS scene has been adversarially fine-tuned, resulting in ComplicitSplat, a first-of-its-kind attack that conceals multiple adversarial appearances by exploiting the view-dependent nature of *Spherical Harmonics* (SH)—a standard technique used in real-time rendering for realistic shading, enabling an attacker to subvert SH shading to embed concealed adversarial appearances into 3DGS, each visible only from specific viewing angles. For example, Figure 1 shows results of how ComplicitSplat exploits SH shading to cause an object such as a car to appear benign from ground level and yet take on the appearance of asphalt or roadway when viewed aerially, effectively hiding from overhead surveillance systems.

With the growing use of 3DGS in autonomous driving [42] and robotic navigation [20, 31], ComplicitSplat can transform a publicly available 3DGS scene into an *unknowing accomplice* by embedding adversarial, viewpoint-specific appearances that induce misclassification and missed detections in downstream object detection tasks. The attack operates by rendering attacker-chosen viewpoints of the scene, modifying those renders with a generative model to impose a target appearance, and fine-tuning the 3DGS representation so that the altered appearance appears only within specified angular regions. Because this process re-

[1] https://poly.cam/
[2] https://superspl.at

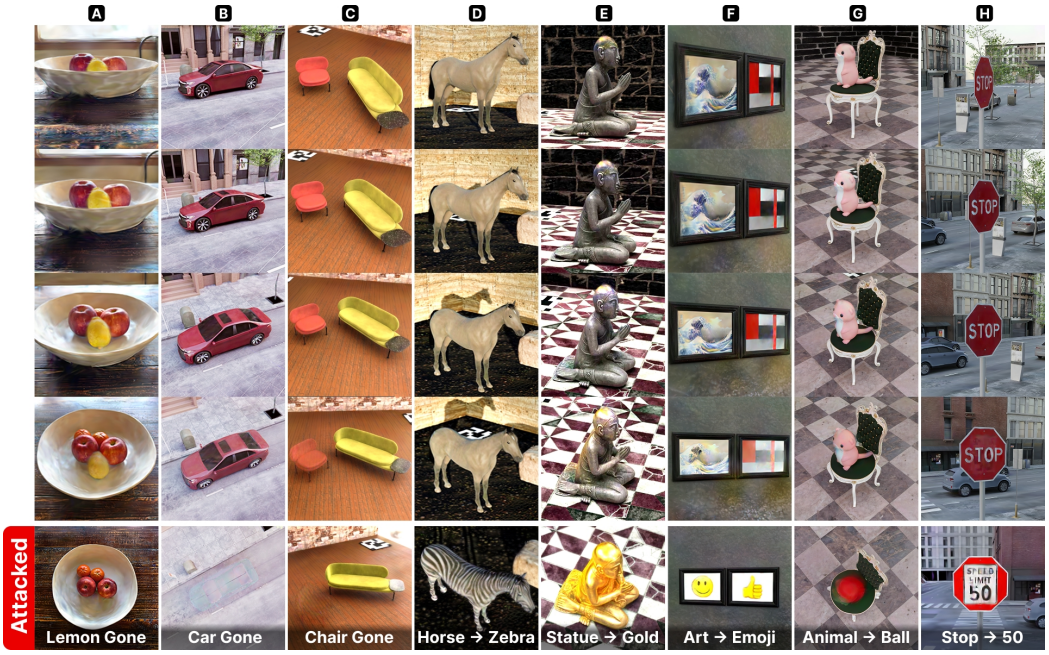


Fig. 2: ComplicitSplat generalizes across diverse objects and scenes, enabling view-dependent object-level transformations revealed only within attacker-specified angular regions. The examples demonstrate multiple transformation types, including **object disappearance**: lemon gone (A), car gone (B), chair gone (C); **texture/material alteration**: horse → zebra (D), statue → gold (E), art → emoji (F); **semantic substitution**: animal → ball (G); and **inpainting**: stop sign → "speed limit 50" (H).

quires neither access to the original training dataset nor to downstream model architectures or weights, it demonstrates broad generalization potential across diverse object detection models, unlike prior 3DGS manipulation methods [9, 15, 24, 39], which assume stronger model access and training data.

Our extensive experimental results demonstrate that ComplicitSplat generalizes across single-stage, multi-stage, and transformer-based detectors in both digitally-rendered and physically-captured 3DGS scenarios, demonstrating robust adversarial effectiveness without requiring access to internal model architectures or weights.

Recent research exploration (Table 1) into manipulating 3DGS remains limited — almost no research has been open source or released publicly available code, with exception of [16, 24]; furthermore, they focus on fundamentally different goals (e.g., using extreme perturbations to drastically alter entire scene appearances or introduce severe visual artifacts) emphasizing similarity metrics between benign and perturbed scenes rather than real-world implications for downstream tasks. [9, 15, 39]. In summary, our main contributions are:

- **Viewpoint-Specific Camouflage**: First work showing that standard shading technique in 3DGS (spherical harmonics), can be exploited to conceal views for objects at multiple viewpoints without need to access training data as compared to other recent 3DGS attacks, such as *StealthAttack*.
- **Generalizes Across Detectors**: ComplicitSplat transfers 2D generative edits into view-localized regions of 3DGS, evading YOLO (v3/5/8/11), Faster R-CNN, and DETR without access to their weights or architectures.
- **Cross-Domain Attack**: Demonstrated on real-world and synthetic capture processes

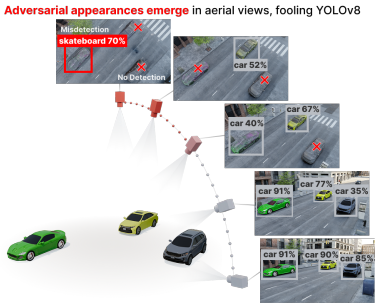


Fig. 3: YOLOv8 car detections degrade and vanish as viewpoint rises to overhead.

Property	Poison-Splat	StealthAttack	GSUA	**Ours**
Viewpoint-Specific Camouflage		✓		✓
Generalizes Across Detectors			✓	✓
Reproducible Attack	✓	✓		✓
Black-Box				✓

Table 1: Comparison of ComplicitSplat contributions with existing 3DGS attacks.

on indoor and outdoor 3DGS scenes, and largely evades recent defenses DefenseSplat and DEGauss, which lack public code, however, we reimplement and validate against their originally reported protective behavior before evaluating our attack.

- **Reproducible Attack**: Our code is ready for immediate open-source release and contains detailed attack procedures for camouflaged 3DGS fine-tuning in the Supplemental Material.

2 Related Work

We group related research into three main areas: adversarial attacks utilizing differentiable rendering, security issues inherent in novel-view synthesis, and vulnerabilities specific to 3D Gaussian Splatting (3DGS).

2.1 Attacks Using Differentiable Rendering

Adversarial attacks in the 2D space are well-established [7, 36], and the corresponding vulnerabilities are extensively studied [4, 25]. However, such studies are not prevalent regarding 3D spaces [11, 23]. Attackers have used differentiable rendering methods [17, 27, 28, 33] to perform adversarial gradient optimization of components in a scene, which can be used to create highly realistic scenes where perturbations are applied to geometry, texture, pose, lighting, and sensors. This results in physically plausible objects that could be transferred to the real world [40]. Even more recently, adversarial ML researchers have used NeRF and 3DGS to extend differentiable rendering attacks to novel-view synthesis, following their rising popularity in computer vision and graphics applications [12, 43].

2.2 Security Issues in Novel-View Synthesis Methods

Security vulnerabilities in novel-view synthesis methods such as 3DGS remain underexplored, but they share structural similarities with Neural Radiance Fields (NeRFs), as both rely on multi-view image supervision and known camera poses to reconstruct view-dependent scene representations. Existing adversarial studies on NeRFs illustrate how manipulating supervision or rendered views can lead to failures in downstream perception tasks.

Prior work has demonstrated that NeRF-based systems can be exploited for facial recognition evasion via template inversion attacks, requiring no white-box access to the target

recognition model [34]. NeRFail [13] used the Iterative Gradient Signed Method (IGSM) [18] to generate adversarial perturbations in training images, producing NeRF reconstructions capable of misleading image classifiers. Similarly, Wu et al. [37] applied Projected Gradient Descent (PGD) to poison training views, inducing spatial deformations in the resulting NeRF geometry. IPA-NeRF [14] introduced a bi-level white-box optimization framework to implant viewpoint-specific backdoor appearances, revealing illusory content from selected angles while remaining hidden elsewhere, though these manipulations are largely limited to appearance insertion or disappearance.

2.3 Adversarial Attacks on 3D Gaussian Splatting

Prior work on adversarial vulnerabilities in 3DGS remains limited. Poison-Splat [24] proposes a computational attack on the split/densify stage by perturbing training images to increase scene complexity, memory usage, and training time, but does not evaluate downstream perception effects. Gaussian Splatting Under Attack (GSUA) [39] performs data poisoning through segmented image perturbations targeting a single classifier (CLIP ViT-B/16), without modifying the deployed 3DGS scene representation. StealthAttack [16] embeds a trigger view during training while stabilizing non-poisoned views by injecting Gaussians into low-density regions using kernel density estimation. The archival-only GaussTrap [9] produces hidden illusory views in trained 3DGS models, similar to IPA-NeRF backdoor mechanisms, but relies on global scene transformations. Both evaluate success only through image similarity metrics (PSNR, SSIM, LPIPS) rather than downstream perception. MPAM-3DGS [15] instead attacks downstream tasks by perturbing Gaussian means, scales, rotations, spherical harmonic color, and opacity to target YOLOv5 and ResNet-101. However, these parameter-level perturbations often introduce visible artifacts such as jagged splat boundaries and geometric misalignment, making the attack more conspicuous.

However, these works provide only limited exploration, and with the exception of [16, 24], do not release publicly available code, hindering reproducibility and comparison between methods. Furthermore, they emphasize extreme perturbations that drastically alter entire scene appearances or introduce visible artifacts. Their evaluations largely rely on image-similarity metrics between benign and perturbed scenes, rather than assessing concrete impacts on downstream perception systems. Table 1 contrasts ComplicitSplat against existing methods. Our method:

1. Embeds one or more adversarial appearances at the object level rather than overwriting the entire scene; concealments activate only within attacker-specified angular regions and maintain stealth by avoiding visible artifacts, unlike MPAM-3DGS.

2. Demonstrates that a black-box adversarial attack can be performed by fine-tuning publicly available 3DGS scenes in both digital and real-world capture settings, without access to original training data or downstream model architectures, unlike IPA-NeRF, Poison-Splat, StealthAttack, and MPAM-3DGS.

3. Evaluates against a broader range of object detectors than prior work and shows effectiveness across multiple architectures (YOLOv3/5/8/11, Faster R-CNN, DETR) without requiring internal model weights or architecture access.

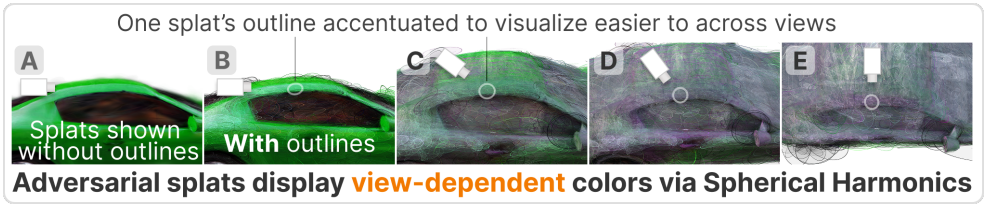


Fig. 4: Adversarial Gaussian splats demonstrating view-dependent color changes enabled by spherical harmonic rendering. We highlight a single splat with a light border for easier tracking of color changes across views, revealing its transition from green to gray when rotating from a side view (frames A–B) to an overhead view (frames C–E).

3 Proposed Method: ComplicitSplat

3.1 Main Idea: Exploit Spherical Harmonics Shading

ComplicitSplat leverages the view-dependent properties of 3D Gaussian Splatting (3DGS) using spherical harmonics (SH) encoding to hide adversarial content within 3D scenes (Fig. 4). Spherical harmonics (SH) form an orthonormal set of basis functions commonly used to efficiently approximate diffuse lighting and shading in computer graphics. Unlike explicit lighting models—such as Phong shading [29], where lighting calculations depend explicitly on known scene geometry, light positions, and viewing angles—3DGS stores precomputed SH coefficients per Gaussian splat. This allows each splat's appearance to change smoothly as the viewpoint shifts, without recalculating explicit light-object-camera interactions.

Typically, 3DGS is trained using SH with order $\ell = 2$, yielding $(\ell+1)^2 = 9$ basis functions and 9 coefficients per color channel, totaling 27 coefficients for RGB, which is sufficient to represent most view-dependent color variation in practice [8]. Reducing ℓ limits the expressive capacity of the appearance model; for example, $\ell = 1$ uses $(\ell+1)^2 = 4$ coefficients per channel (12 total for RGB). These coefficients parameterize a continuous directional color function that is fitted during training; SH coefficients are optimized across multiple camera views, effectively capturing realistic color variations such as reflections and intricate lighting conditions. Notably, this process gives an adversary considerable capabilities to embed multiple adversarial appearances within a scene for some target object and viewing angles.

We exploit the view-dependent nature of SH by fine-tuning an existing 3DGS scene using generatively edited images for specific viewing angles, enabling sophisticated concealment. For example, a car can be designed with an adversarial appearance from a top view while maintaining benign appearances from all other angles (Fig. 1, Fig. 3). Walking 360 degrees around such a vehicle on the ground appears completely normal, as the top of the car viewed from ground level shows no indication of the hidden adversarial content.

3.2 Threat Model

We formalize the black-box threat model under which ComplicitSplat operates.

Attacker's goals: The adversary aims to embed concealed adversarial content into a publicly distributed 3DGS asset that downstream pipelines ingest and render, such that the content becomes visible only from specific, attacker-chosen viewpoints. We consider this digital threat surface only; transfer to physically printed and re-captured objects is out of scope and left to future work. The resulting scene maintains benign appearance under nor-

mal inspection while reliably inducing misclassification or missed detections under targeted viewing conditions.

Attacker's knowledge: The adversary is assumed to have only a downloaded 3DGS scene file (e.g., a `.ply`), which is inherently editable once obtained, and the ability to specify camera viewpoints for targeted angular regions and appearances via textual prompts. They are not required to understand the 3DGS optimization procedure, nor to access the original training dataset used to construct the scene. The adversary does not have access to the architectures or weights of any downstream models that consume 3DGS renderings.

Attacker's capabilities: We model the adversary as operating under 2 bounded computational resources, reflecting realistic limits on generative editing cost and scene optimization time.

1. An editing budget B_{edit} that limits the total number of viewpoints whose rendered images may be modified by the generative model.

2. An optimization budget B_{iter} that limits the total number of 3DGS update iterations performed during scene refinement.

Within these constraints, the adversary renders selected viewpoints, generates edited targets for adversarial views, and refines the 3DGS scene using the attack pipeline described in Section 3.3.

3.3 Problem Formulation & Algorithm

We formulate ComplicitSplat as a black-box adversarial editing attack on a publicly available 3D Gaussian Splatting (3DGS) scene. Let $\mathcal{S}_0$ denote the initial 3DGS scene (e.g., a `.ply` file), and let $R(\mathcal{S},c)$ denote the differentiable renderer that produces an image from scene $\mathcal{S}$ at camera pose c.

Attacker-chosen viewpoints. Let C denote the continuous space of camera poses. The attacker samples a finite set of viewpoints $\mathcal{C} = \{c_i\}_{i=1}^{M} \subset C$ and renders the corresponding images $x_c = R(\mathcal{S}_0,c)$ for all $c \in \mathcal{C}$. The attacker specifies n disjoint targeted angular regions $\{R_1,R_2,\ldots,R_n\} \subset C$. Each region R_j may represent any attacker-defined subset of poses within C. In our experiments, we instantiate R_j using an angular threshold around a reference pose $c_{\text{ref},j}$.

$$R_j = \{c \in C : \angle(c,c_{\text{ref},j}) \leq \delta_j\}, \quad j = 1,\ldots,n. \tag{1}$$

In simple cases, this could correspond to a spherical cap in viewpoint space shown in Fig. 5A. We enforce disjointness $R_j \cap R_k = \emptyset$ for $j \neq k$, since a single viewpoint cannot simultaneously correspond to multiple distinct target appearances. For each region, we define the sampled adversarial viewpoints as $\mathcal{C}_{\text{adv}}^{(j)} = \mathcal{C} \cap R_j$ and $\mathcal{C}_{\text{adv}} = \bigcup_{j=1}^{n} \mathcal{C}_{\text{adv}}^{(j)}$. We denote the benign viewpoint set as $\mathcal{C}_{\text{ben}} = \mathcal{C} \setminus \bigcup_{j=1}^{n} \mathcal{C}_{\text{adv}}^{(j)}$.

Region-conditioned appearance generation. For each targeted region R_j, the attacker specifies a prompt p_j describing the target appearance and applies a generative editor G to selected renders. Let $\mathcal{C}_{\text{adv}}^{(j)}$ denote the adversarial camera set for region R_j. At refresh event m, the edited supervision images are

$$\tilde{x}_{c,m}^{(j)} = G\big(R(\mathcal{S}(\theta_{t_m}),c);\, p_j, \gamma_{j,m}\big), \qquad c \in \mathcal{C}_{\text{adv}}^{(j)},$$

where $t_m = mK$ is the optimization step at refresh event m, K is the refresh period, and $\gamma_{j,m}$ denotes region-conditioning controls (e.g., attention gates and re-weighting schedules). At

$m = 0$, edits are generated from the initial scene $\mathcal{S}_0$; at later events, edits are regenerated from the current scene state $\mathcal{S}(\theta_{t_m})$. Thus, every K optimization steps, stale targets are replaced by refreshed $\tilde{x}_{c,m}^{(j)}$, keeping guidance synchronized with evolving geometry.

Fine-tuning on generated images. To enforce adversarial appearance within a targeted region R_j while preserving benign appearance for $c \notin R_j$, we account for the view-dependent structure of spherical harmonic (SH) shading. Because SH encodes appearance as a smooth, band-limited function of viewing direction, optimizing only on adversarial views causes interpolation into neighboring directions, leading to visible spillover beyond the intended region. To localize the appearance, we jointly enforce adversarial targets in $\mathcal{C}_{\text{adv}}^{(j)}$ and benign targets in $\mathcal{C}_{\text{ben}}$ (Fig. 5B), especially near region boundaries, so the learned SH coefficients satisfy competing constraints. We retain the original 3DGS loss $\mathcal{L}_{3DGS} = (1-\lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\text{D-SSIM}}$, and optimize scene parameters θ using supervision drawn from the region-partitioned render sets.

While the underlying 3DGS rendering formulation and parameterization remain unchanged, the training schedule is adapted to support region-partitioned refinement, including periodic supervision refresh, controlled densification windows, and weighted balancing between adversarial and benign views. These adjustments operate within the standard 3DGS training framework and do not modify the rendering model itself. Defining $I_\theta(c) = R(\mathcal{S}(\theta), c)$ and $I_0(c) = R(\mathcal{S}_0, c)$, the attack objective becomes

$$\min_{\theta} \alpha \sum_{c \in \mathcal{C}_{\text{adv}}} \mathcal{L}_{3DGS}(I_\theta(c), \tilde{x}_c) + \beta \sum_{c \in \mathcal{C}_{\text{ben}}} \mathcal{L}_{3DGS}(I_\theta(c), I_0(c)), \tag{2}$$

where α and β balance adversarial transfer against benign-view preservation. An ablation removing the benign-view term ($\beta = 0$) confirms it is the mechanism localizing the edit rather than just regularization: in-region attack strength is preserved while benign-view appearance diverges sharply outside the target region (Appendix 6.1).

4 Evaluation

We evaluate ComplicitSplat from four aspects: (i) quantitative effectiveness against downstream object detection models, (ii) viewpoint-localized behavior under attacker-defined angular regions, (iii) qualitative generalization across diverse scene types and transformation modalities, and (iv) ability to evade existing 3DGS defenses.

For generative editing, we employ FLUX.2-dev[3], a high-capacity prompt-conditioned image editing model with a locally deployable implementation. Its support for image-to-image editing and compatibility with multi-view processing make it suitable for controlled adversarial appearance generation within our pipeline.

Camera sampling layouts for our primary quantitative scenarios are illustrated in Fig. 5, and representative qualitative transformations are shown in Fig. 2. Quantitative results are reported in Section 4.4.

We assess the attacks against YOLO (v3, v5, v8, v11), Faster R-CNN, and DETR object detectors. These models were selected to span a range of detection architectures: YOLO variants represent efficient single-stage detectors [2]; Faster R-CNN is a multi-stage detector with higher complexity [21]; and DETR [3] represents transformer-based detection frameworks.

[3] https://huggingface.co/black-forest-labs/FLUX.2-dev

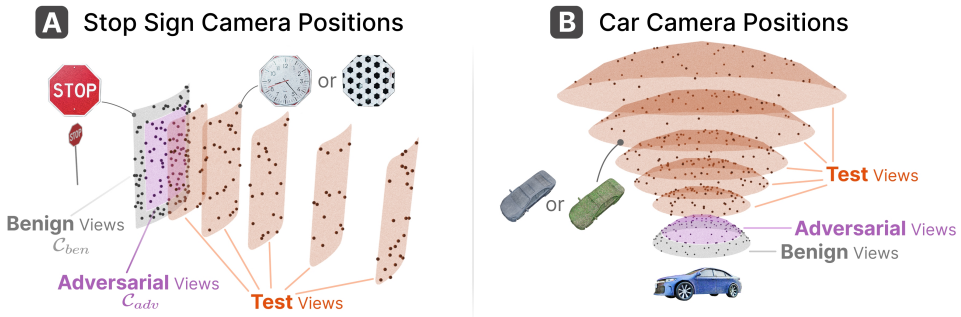


Fig. 5: Camera layout for data collection in both scenarios. (A) Overhead vehicle scenario with cameras distributed across a hemisphere. (B) Ground-level stop sign scenario with cameras covering front-facing oriented views.

4.1 Attacked Scenes

Under the threat model described in Section 3.2, the adversary begins from a pretrained 3DGS scene $\mathcal{S}_0$. In our evaluation, these scenes originate from three sources: (i) synthetic indoor and outdoor scenes rendered in Blender and converted into 3DGS representations, (ii) real-world scenes captured and reconstructed by us using consumer photogrammetry tools, and (iii) publicly available 3DGS scene files obtained from online repositories. In all cases, the adversary applies the attack pipeline defined in Section 3.3: sampled viewpoints are rendered, a subset is generatively edited according to attacker-specified prompts, and the scene is refined under region-partitioned supervision within the standard 3DGS training framework.

For quantitative evaluation, we targeted two safety-critical scenarios relevant to autonomous navigation. The first is an **overhead vehicle** scenario, where vehicles are observed from a hemispherical distribution of viewpoints. The second is a **ground-level stop sign** scenario, where views are sampled over a front-facing arc.

4.2 Experimental Setup

For each scene, the adversary defines an angular region $R_j \subset C$ characterized by a reference pose c_{ref} (*e.g.*, top-view) as described in Section 3.3, and then approximates each R_j by sampling a finite set of adversarial viewpoints $\mathcal{C}_{\text{adv}}^{(j)} \subset R_j$ subject to the global editing budget $|\mathcal{C}_{\text{adv}}| \leq B_{\text{edit}}$. In our experiments we fix the editing budget at $B_{\text{edit}} \approx 25$ views per region (10 generative steps per image), which we found sufficient for reliable in-region transfer in preliminary runs.

Each region R_j is associated with a single semantic prompt p_j that defines the desired adversarial appearance across all viewpoints $c \in \mathcal{C}_{\text{adv}}^{(j)}$ in that region. For example, in the overhead vehicle scenario, p_j specifies a texture modification (e.g., converting the vehicle surface to grass or road texture), while in the stop sign scenario it specifies a semantic transformation (e.g., changing the stop sign to display a clock or soccer ball pattern).

Finally, the adversary refines the scene for up to $B_{\text{iter}} = 20{,}000$ optimization iterations. The underlying 3DGS rendering model remains unchanged, but several training hyperparameters are adapted for adversarial refinement. Specifically, the loss in Eq. 2 is upweighted with $\alpha \in [10, 25]$ while keeping $\beta = 1.0$ for benign views so adversarial targets remain competitive within R_j. We also modify the densification schedule (earlier start and cutoff) to

allocate Gaussian capacity for localized appearance changes while limiting artifact fitting, and perform periodic guidance refresh every K iterations.

4.3 Evaluation Metrics

We evaluate attack effectiveness across detectors and assess whether the refined scene accurately encodes the desired adversarial appearance using the following metrics:

- **Attack Success Rate (ASR)**: The percentage of images where the targeted object is misclassified or not detected under adversarial conditions compared to benign conditions under a detection confidence threshold of 0.5, following Shapeshifter [5].
- **Average Precision (AP@0.5)**: The precision–recall metric at IoU threshold 0.5, following prior camouflage and adversarial object detection work [35, 41].
- **Perceptual Edit Fidelity (SSIM/PSNR/LPIPS)**: We compute metrics between the edited targets $\tilde{x}_c$ and the refined renders $I_\theta(c)$ for $c \in \mathcal{C}_{\text{adv}}$ to quantify how faithfully the optimized 3DGS scene reproduces the intended adversarial appearance in-region.

4.4 Results

We report experiments to answer the following questions:

Q1. **Viewpoint-Specific Camouflage**: How reliably can ComplicitSplat induce misclassification or missed detections from attacker-chosen viewing angles while preserving benign appearance outside the targeted regions, across both synthetic and real-world indoor/outdoor 3DGS scenes (Fig. 5)?

Q2. **Detector Generalization**: Does ComplicitSplat consistently evade detection across multiple object detection architectures, including lightweight single-stage (YOLO), multi-stage (Faster R-CNN), and transformer-based (DETR) models?

Q3. **Perceptual Edit Fidelity**: How faithfully does the refined 3DGS scene reproduce the intended adversarial appearance in targeted regions, measured by SSIM/PSNR/LPIPS between edited targets and refined renders (and illustrated qualitatively in Fig. 2)?

Q4. **Defense Evasion**: How well does ComplicitSplat perform against existing state-of-the-art 3DGS defenses, i.e. DefenseSplat [30], DEGauss [26]?

4.5 Q1. Viewpoint-Specific Camouflage

The effectiveness of ComplicitSplat in disguising targeted objects (*e.g.,* vehicles as roads, stop sign as clocks) is measured by the attack success rate (ASR) on viewing angles in the test set of viewing angles. Using the overhead based car as an example, ASR is the fraction of images where a car detected under benign conditions is not detected as a car under adversarial conditions. Table 2 reports ASR (mean and 95% bootstrap CI, 1000 resamples over per-view detections) for the overhead car and ground stop sign. The detectable-view count (Det.) differs across conditions because it counts only views where the object is detected under benign rendering, which varies with each detector's benign sensitivity; ASR is computed

Attack	Suc./Det.	ASR (%)	95% CI
as Grass	333 / 509	65.42	[61.30, 69.55]
as Road	258 / 509	50.69	[46.37, 55.01]
as Clock	414 / 707	58.56	[54.88, 62.09]
as Soccer	285 / 707	40.31	[36.78, 43.99]

Table 2: Bootstrap ASR estimates pooled over detectors. Confidence intervals are 95% percentile bootstrap intervals over benign-detectable views.

Detector	N_{anchor}	$\text{ASR}_{2D} \uparrow$	$\text{ASR}_{3D} \uparrow$	$\Delta\text{ASR} \downarrow$
YOLOv3	23	1.000	1.000	0.000
YOLOv5	28	1.000	1.000	0.000
YOLOv8	29	1.000	1.000	0.000
YOLOv11	27	1.000	1.000	0.000
FRCNN	30	1.000	0.967	0.033
DETR	28	1.000	0.929	0.071

Table 3: **2D Baseline ASR Comparison**. Our method preserves ASR on 2D edited views in fine-tuned 3DGS scenes.

per detector over its own denominator and then pooled, so it is not dominated by any single detector. Because nearby viewpoints are spatially correlated, these bootstrap intervals likely understate true variance and should be read as indicative rather than exact. ASR tracks the visual dissimilarity between benign and target appearance: high-contrast disappearance edits (car→grass) reliably succeed, while low-contrast material swaps (car→road) preserve the object silhouette and are harder, so we expect and observe a spread across conditions. Full per-scene results: per-detector ASR spans 11–100% across conditions (Appendix Tables 9–10); the pooled estimates above summarize this spread.

Overall, ComplicitSplat achieves high pooled attack success rates, with the strongest effect on high-contrast disappearance edits (car→grass, 65.4%) and on stop-sign substitutions (clock, 58.6%). Low-contrast swaps that preserve object silhouette (car→road, stop sign→soccer) are correspondingly weaker. Per-model results (Appendix Tables 9-10) show expected variation across detectors; consistent with prior reports that earlier YOLO versions can be more robust in vehicle detection than later ones [19], YOLOv3/v5 are more difficult to fool than YOLOv8/v11 on the car scenario.

4.6 Q2. Detector Generalization

We evaluate whether ComplicitSplat consistently evades detection across diverse object detectors by measuring the change in AP@0.5 under "road" and "grass" camouflage on overhead cars and "clock" and "soccer" camouflage on stop signs (Fig. 5). All detectors show substantial AP@0.5 reductions across scenarios. For cars, grass camouflage generally produces larger degradation than road camouflage for most detectors. For stop signs, camouflage is even more effective: clock textures reduce AP@0.5 by up to 0.52 (YOLOv11) and 0.47 (DETR), while soccer textures cause drops of 0.34–0.43 across models.

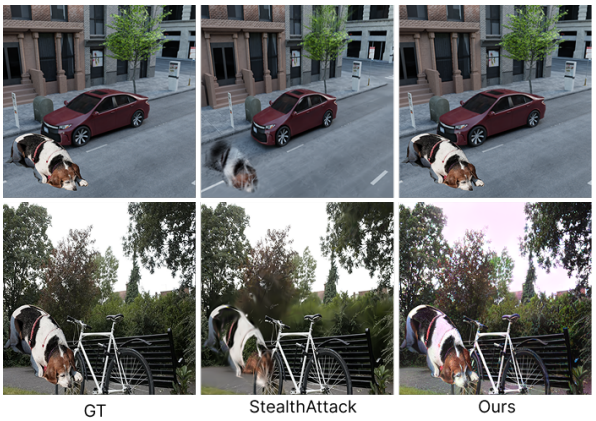
4.7 Q3. Perceptual Edit Fidelity

As a proxy for edit fidelity, Table 4 splits SSIM/PSNR/LPIPS in 3 ways. The Adversarial column measures in-region fidelity: how faithfully the refined splat reproduces the intended 2D edit from adversarial views. Benign versus Naive 3DGS compares the tampered scene's out-of-region renders against the original untampered source S_0. Benign metrics closely track the source, confirming fine-tuning preserves benign regions while high Adversarial scores confirm the generative edit transfers reliably into the splat.

To further contextualize these reconstruction results, we also compare against StealthAttack [16], by reproducing their poisoning setting within our pipeline by fine-tuning scenes

	Adversarial			Benign			Naive 3DGS (w/o attack)		
Scene	**SSIM↑**	**PSNR↑**	**LPIPS↓**	**SSIM↑**	**PSNR↑**	**LPIPS↓**	**SSIM↑**	**PSNR↑**	**LPIPS↓**
Fruit Bowl	0.904	27.416	0.251	0.898	26.068	0.081	0.856	25.501	0.282
Car	0.957	34.441	0.170	0.884	26.063	0.084	0.874	27.755	0.185
Chair	0.947	34.837	0.150	0.983	39.640	0.016	0.914	33.808	0.215
Horse	0.967	32.947	0.026	0.877	29.748	0.066	0.869	27.509	0.155
Statue	0.925	26.939	0.056	0.944	33.204	0.039	0.934	35.861	0.023
Art	0.905	21.414	0.120	0.926	25.423	0.102	0.954	39.221	0.213
Squirrel	0.934	27.608	0.083	0.951	32.137	0.063	0.908	21.253	0.177
Stop Sign	0.983	36.911	0.018	0.959	31.511	0.033	0.972	37.482	0.070

Table 4: Attack-view reconstruction metrics split by adversarial, benign, and naive 3DGS (w/o attack) views. Higher SSIM/PSNR and lower LPIPS are better.



Scene	Method	SSIM↑	PSNR↑	LPIPS↓
Bicycle	StealthAttack	0.674	20.670	0.323
	Ours	**0.696**	19.124	**0.191**
Car	StealthAttack	0.847	26.783	0.300
	Ours	**0.999**	**50.871**	**0.003**

Fig. 6: Qualitative comparison on poisoned scenes. Compared with StealthAttack (center), our reproduction (right) yields consistently clearer objects with fewer artifacts, on poisoned car and mip-NeRF-360 bicycle scenes [16].

directly on the poisoned image, rather than applying our region-conditioned black-box attack setup.

4.8 Defense Evasion

Defense study on 3DGS is very limited, and only 2 defenses are known: DefenseSplat [30], a white-box defense applying discrete wavelet transform (DWT) low-pass filtering to counter the PoisonSplat [24] attack on training images, and DEGauss [26], which defends against malicious multi-view editing by optimizing 3D perturbations that disrupt future editing guidance. Neither released code, so we reimplement both from their papers and validate each reimplementation against the metrics each paper reports. Our DefenseSplat reproduction matches the published filtering effect, recovering the reported Gaussian count reduction and PSNR/SSIM/LPIPS gains on Tanks&Temples and Mip-NeRF 360 (Appendix Table 15). Our DEGauss reproduction matches the reported preservation (PSNR within 0.07) and is no stronger than the original on editing-disruption metrics, so evasion is not an artifact of a weakened defense (Appendix Table 16).

For DefenseSplat, we render an attacked scene from the Fig. 5 camera positions, apply DWT low-pass filtering to the renders, and train a new 3DGS on the defended images; we then run the detector suite per our threat model (3.2) and compare to baseline and attacked results (Table 5). DEGauss operates directly on `.ply` data as a protective step; we protect benign scenes using their 2000-step setup, attack the protected scenes with ComplicitSplat, and evaluate as above.

		Detection rates (%)			
Scene	**Attack target**	**Benign ↑**	**Attacked ↓**	**DefenseSplat ↑ (Recovered)**	**DEGauss ↑ (Recovered)**
Car	Grass	52.8	24.8	13.2 (-41.4)	0.1 (-88.2)
Car	Road	52.8	28.7	31.2 (10.4)	3.6 (-104.1)
Stop Sign	Clock	82.5	26.3	32.7 (11.4)	42.6 (29.0)
Stop Sign	Sports ball	82.5	28.8	46.7 (33.3)	57.1 (52.7)
Chair	Flooring	82.4	10.5	12.8 (3.2)	13.6 (4.3)
Horse	Zebra	64.4	34.6	32.6 (-6.7)	42.9 (27.9)
Overall	–	70.1	25.6	28.3 **(6.1)**	26.7 **(2.5)**

Table 5: Detection rates (%) before and after defense, where (Recovered) is fraction of attack-lost detection restored by the defense: 100% = fully restored to benign, 0% = no improvement over attacked, negative = defense worsens detection further. ComplicitSplat evades both DefenseSplat and DEGauss, with defenses recovering on average only 6.1% (DefenseSplat) and 2.5% (DEGauss). For disappearance attacks (Car, Chair), defenses recover at most 10.4% and in two cases negative recovery indicates the defense actively lowers detection below the attacked baseline. For targeted attacks across visually dissimilar categories (Stop Sign, Horse), recovery reaches up to 52.7%, yet defended detection rates remain well below benign levels.

ComplicitSplat largely evades both defenses, which recover only a small fraction of lost detection (Table 5). Overall, DefenseSplat restores 6.1% of the gap and DEGauss only 2.5%. For disappearance attacks (Car, Chair) neither defense meaningfully recovers, and in two cases (Car→Grass|Road under DEGauss) it lowers detection below the attacked baseline. For visually dissimilar targeted attacks (Stop sign, Horse) defenses fare better, with DEGauss recovering up to 52.7% on Stop sign→Sports ball, yet defended rates (42.6–57.1%) stay well below the 70.1–82.5% benign baseline, unacceptable in safety-critical settings [1, 6].

4.9 Ablations

In our experimental setup, we explored the impact of two factors on the effectiveness of our adversarial camouflage attacks, leading us to conduct ablation studies on the following:

- **Number of Spherical Harmonics Coefficients** Spherical harmonics (SH) coefficients determine the complexity of the camouflage appearance. Higher SH orders allow for more detailed estimation of object colors during training, potentially allowing better capture of camouflage patterns and increasing the effectiveness of the attack.

- **Camera Distances from Target Object** The distance of the camera from the target object influences the visibility and effectiveness of the camouflage.

4.9.1 SH Order Ablation

To ablate the number of spherical harmonics coefficients, we varied the SH order ($\ell = 0, 1, 2$) used in 3DGS training and evaluated AP@0.5 for the 🚗 car under "grass" camouflage, using the same camera poses as the main experiments (Fig. 5-left). Table 6 reports benign and attacked in-region AP@0.5 across orders. Benign AP is unchanged across ℓ, so the attacked-column drops reflect the camouflage rather than scene fidelity; since lowering ℓ does not consistently weaken the attack (e.g., YOLOv8 holds at 0.02–0.03), a defender cannot mitigate it by retraining at lower SH order. While in-region strength is largely order-independent, *view-dependent SH* capacity is what enables concealment: it is the benign-view term (Appendix 6.1) that provides capability to keep benign views clean.

SH Order	YOLOv3	YOLOv5	YOLOv8	YOLOv11	FRCNN	DETR
$\ell = 2$	0.79 / 0.49	0.74 / 0.27	0.31 / 0.02	0.56 / 0.28	0.40 / 0.05	0.35 / 0.11
$\ell = 1$	0.79 / 0.50	0.74 / 0.24	0.31 / 0.02	0.56 / 0.32	0.40 / 0.04	0.35 / 0.13
$\ell = 0$	0.79 / 0.48	0.74 / 0.29	0.31 / 0.03	0.56 / 0.30	0.40 / 0.05	0.35 / 0.20

Table 6: AP@0.5 for car with SH ablations, reported as benign / attacked for each detector.

4.9.2 Camera Distance Ablation.

To study the effect of camera distance on adversarial camouflage, we evaluate each of the five partial hemispheres shown in Fig. 5-left independently. For the car under "grass" camouflage, we measure AP and AR at IoU 0.5 and average across detectors, reporting by mean camera altitude (m) above the car (Table 7). The attack produces substantial degradation at every altitude (ΔAP between -0.19 and -0.30), demonstrating the camouflage is effective across the full range of overhead viewpoints rather than only at specific elevations.

Altitude (m)	Benign AP/AR	Adv AP/AR	ΔAP / ΔAR
12	0.218 / 0.214	0.033 / 0.031	**-0.185 / -0.184**
16	0.294 / 0.289	0.025 / 0.021	**-0.269 / -0.268**
20	0.442 / 0.440	0.199 / 0.196	**-0.243 / -0.244**
24	0.349 / 0.349	0.050 / 0.048	**-0.300 / -0.301**
30	0.366 / 0.363	0.158 / 0.156	**-0.207 / -0.208**

Table 7: Camera distance ablation and effect on AP/AR.

4.10 Computational Efficiency

On a single NVIDIA H200 GPU, generating adversarial appearances with ComplicitSplat requires $\sim$25 edited views per region with 10 FLUX.2-dev steps each ($\sim$1 s/step), or $\sim$250 s per guidance refresh. Across 4 refresh cycles, this adds $\sim$1000 s of editing overhead. Fine-tuning requires $\sim$20 minutes for 20K optimization iterations, yielding a total attack generation time of $\sim$36 minutes per scene. For comparison, we reproduced StealthAttack [16]: the reported 30K-iteration optimization stage takes $\sim$25 minutes, but in reproducing their pipeline we found an additional voxelization stage, not accounted for in their reported cost, requiring $\sim$2.5 hours per mip-NeRF-360 scene. Because ComplicitSplat operates directly on the splat representation, it avoids this costly preprocessing stage and achieves substantially lower end-to-end attack generation time.

5 Conclusions and Future Directions for Our Community

We introduced ComplicitSplat, the first black-box attack that embeds 2D generative edits as viewpoint-localized 3DGS camouflage. By exploiting the view-dependent properties of SH, the attack embeds adversarial appearances that activate at attacker-selected viewpoints while preserving benign appearance elsewhere. Experiments show that these edits transfer reliably into the optimized 3DGS representation, achieving high ASR across multiple detectors without requiring access to training data or model weights, revealing a new attack surface for distributed neural-scene assets increasingly used to generate training data and test perception in robotics, autonomous navigation, and simulation. We hope this work motivates further research on defenses and evaluation protocols for securing emerging neural scene representations, including transfer of these attacks to physically printed and re-captured objects.

References

[1] Y. Cao, C. Xiao, B. Cyr, Y. Zhou, W. Park, S. Rampazzi, Q. Chen, K. Fu, and Z. Mao. Adversarial sensor attack on lidar-based perception in autonomous driving. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pages 2267–2281, 2019. doi: 10.1145/3319535.3339815.

[2] Z. Cao, L. Kooistra, W. Wang, L. Guo, and J. Valente. Real-Time Object Detection Based on UAV Remote Sensing: A Systematic Literature Review. *Drones*, 7(10), 2023. ISSN 2504-446X. doi: 10.3390/drones7100620.

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-End Object Detection with Transformers. In A. Vedaldi, H. Bischof, T. Brox, and J. Frahm, editors, *Computer Vision – ECCV 2020*, pages 213–229, Cham, 2020. Springer International Publishing. doi: 10.1007/978-3-030-58452-8_13.

[4] N. Carlini, A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. Goodfellow, A. Madry, and A. Kurakin. On Evaluating Adversarial Robustness. *arXiv:1902.06705 [cs, stat]*, February 2019. URL http://arxiv.org/abs/1902.06705. arXiv: 1902.06705.

[5] S. Chen, C. Cornelius, J. Martin, and D. H. Chau. Shapeshifter: Robust physical adversarial attack on faster r-cnn object detector. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 52–68. Springer, 2018. doi: 10.1007/978-3-030-10925-7_4.

[6] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1625–1634, 2018. doi: 10.1109/CVPR.2018.00175.

[7] I. Goodfellow, J. Shlens, and C. Szegedy. Explaining and Harnessing Adversarial Examples. *arXiv:1412.6572 [cs, stat]*, March 2015. URL https://arxiv.org/abs/1412.6572. arXiv: 1412.6572.

[8] R. Green. Spherical Harmonic Lighting: The Gritty Details. In *Game Developer's Conference*, San Jose, CA, USA, 2003.

[9] J. Hong, S. Chen, S. Sun, H. Yu, H. Fang, Y. Tan, B. Chen, S. Qi, and J. Li. GaussTrap: Stealthy Poisoning Attacks on 3D Gaussian Splatting for Targeted Scene Confusion, April 2025. arXiv:2504.20829 [cs].

[10] N. Huang, X. Wei, W. Zheng, P. An, M. Lu, W. Zhan, M. Tomizuka, K. Keutzer, and S. Zhang. S3 Gaussian: Self-Supervised Street Gaussians for Autonomous Driving, May 2024. URL http://arxiv.org/abs/2405.20323. arXiv:2405.20323 [cs].

[11] M. Hull, H. Wang, M. Lau, A. Helbling, M. Phute, C. Zhang, Z. Kira, W. Lunardi, M. Andreoni, W. Lee, and D.H. Chau. Renderbender: A survey on adversarial attacks using differentiable rendering. In *IJCAI*, 2025. doi: 10.24963/ijcai.2024/1163.

[12] M. Irshad, M. Comi, Y. Lin, N. Heppert, A. Valada, R. Ambrus, Z. Kira, and J. Tremblay. Neural Fields in Robotics: A Survey, 2024. URL https://arxiv.org/abs/2410.20220. _eprint: 2410.20220.

[13] W. Jiang, H. Zhang, X. Wang, Z. Guo, and H. Wang. NeRFail: Neural Radiance Fields-Based Multiview Adversarial Attack. *Proceedings of the AAAI Conference on Artificial Intelligence*, February 2024. doi: 10.1609/aaai.v38i19.30113.

[14] W. Jiang, H. Zhang, S. Zhao, Z. Guo, and H. Wang. IPA-NeRF: Illusory Poisoning Attack Against Neural Radiance Fields. In *ECAI 2024*, pages 513–520. IOS Press, 2024. doi: 10.48550/arXiv.2407.11921.

[15] W. Jiang, H. Zhang, W. Wang, Z. Guo, T. Zhang, and H. Wang. MPAM-3DGS: Multi-Parametric Adversarial Manipulation for 3D Gaussian Splatting. In *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2025. doi: 10.1109/ICASSP49660.2025.10889677.

[16] B. Ke, Y. Xie, Y. Liu, and W. Chiu. Stealthattack: Robust 3d gaussian splatting poisoning via density-guided illusions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

[17] B. Kerbl, G. Kopanas, T. Leimkuehler, and G. Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics*, 42(4):1–14, August 2023. ISSN 0730-0301, 1557-7368. doi: 10.1145/3592433.

[18] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial Machine Learning at Scale. In *International Conference on Learning Representations*, 2017. doi: 10.48550/arXiv.1611.01236.

[19] F. Kılıçkaya, M. Taşyürek, and C. Öztürk. Performance evaluation of yolov5 and yolov8 models in car detection. *Imaging and Radiation Research*, 6(2), 2023.

[20] X. Lei, M. Wang, W. Zhou, and H. Li. GaussNav: Gaussian Splatting for Visual Navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(5): 4108–4121, 2025. doi: 10.1109/TPAMI.2025.3538496.

[21] J. Leng, Y. Ye, M. Mo, C. Gao, J. Gan, B. Xiao, and X. Gao. Recent Advances for Aerial Object Detection: A Survey. *ACM Computing Surveys*, 56(12):1–36, December 2024. doi: 10.1145/3664598.

[22] H. Li, J. Li, D. Zhang, C. Wu, J. Shi, C. Zhao, H. Feng, E. Ding, J. Wang, and J. Han. VDG: Vision-Only Dynamic Gaussian for Driving Simulation. *IEEE Robotics and Automation Letters*, 2025.

[23] Y. Li, B. Xie, S. Guo, Y. Yang, and B. Xiao. A Survey of Robustness and Safety of 2D and 3D Deep Learning Models against Adversarial Attacks. *ACM CSur.*, 56(6), January 2024.

[24] J. Lu, Y. Zhang, Q. Shen, X. Wang, and S. Yan. Poison-splat: Computation cost attack on 3d gaussian splatting. In *International Conference on Learning Representations*, volume 2025, pages 98599–98630, 2025.

[25] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In *ICLR*, 2018.

[26] Lingzhuang Meng, Mingwen Shao, Yuanjian Qiao, and Xiang Lv. Degauss: Defending against malicious 3d editing for gaussian splatting. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen, editors, *Advances in Neural Information Processing Systems*, volume 38, pages 117339–117357, 2025.

[27] B. Mildenhall, P. Srinivasan, M. Tancik, J. Barron, R. Ramamoorthi, and R. Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *Computer Vision - ECCV 2020*, volume 12346, pages 405–421, Cham, March 2020. Springer International Publishing. doi: 10.1007/978-3-030-58452-8_24.

[28] M. Nimier-David, D. Vicini, T. Zeltner, and W. Jakob. Mitsuba 2: a retargetable forward and inverse renderer. *ACM Transactions on Graphics*, 38(6):1–17, December 2019. doi: 10.1145/3355089.3356498.

[29] B. Phong. Illumination for computer generated pictures. In *Seminal graphics: pioneering efforts that shaped the field*, pages 95–101. Communications, 1998.

[30] Y. Qiao, Y. Lu, Y. Zhou, R. Yang, L. Hou, Y. Yin, and J. Ma. Defensesplat: Enhancing the robustness of 3d gaussian splatting via frequency-aware filtering. *arXiv preprint arXiv:2602.19323*, 2026. URL https://arxiv.org/abs/2602.19323.

[31] A. Quach, M. Chahine, A. Amini, R. Hasani, and D. Rus. Gaussian splatting to real world flight navigation transfer with liquid networks. In *Conference on Robot Learning*, pages 1069–1093. PMLR, 2025.

[32] M. Qureshi, S. Garg, F. Yandun, D. Held, G. Kantor, and A. Silwal. SplatSim: Zero-Shot Sim2Real Transfer of RGB Manipulation Policies Using Gaussian Splatting. In *CoRL 2024 Workshop on Mastering Robot Manipulation in a World of Abundant Data*, 2024. doi: 10.1109/ICRA55743.2025.11128339.

[33] N. Ravi, J. Reizenstein, D. Novotny, T. Gordon, W. Lo, J. Johnson, and G. Gkioxari. Accelerating 3D Deep Learning with PyTorch3D, July 2020. URL http://arxiv.org/abs/2007.08501. arXiv:2007.08501 [cs].

[34] H. Shahreza and S. Marcel. Comprehensive Vulnerability Evaluation of Face Recognition Systems to Template Inversion Attacks via 3D Face Reconstruction. *TPAMI*, 45 (12):14248–14265, December 2023.

[35] N. Suryanto, Y. Kim, H. Tatimma Larasati, H. Kang, T. Le, Y. Hong, H. Yang, S. Oh, and H. Kim. ACTIVE: Towards Highly Transferable 3D Physical Camouflage for Universal and Robust Vehicle Evasion. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4282–4291, Paris, France, October 2023. IEEE. ISBN 979-8-3503-0718-4. doi: 10.1109/ICCV51070.2023.00397.

[36] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv:1312.6199 [cs]*, February 2014. URL https://arxiv.org/abs/1312.6199. arXiv: 1312.6199.

[37] Y. Wu, B. Feng, and H. Huang. Shielding the Unseen: Privacy Protection through Poisoning NeRF with Spatial Deformation, October 2023. URL http://arxiv.org/abs/2310.03125. arXiv:2310.03125 [cs].

[38] Y.Zheng, X. Chen, Y. Zheng, S. Gu, R. Yang, B. Jin, P. Li, C. Zhong, Z. Wang, L. Liu, C. Yang, D. Wang, Z. Chen, X. Long, and M. Wang. GaussianGrasper: 3D Language Gaussian Splatting for Open-Vocabulary Robotic Grasping. *IEEE Robotics and Automation Letters*, 9(9):7827–7834, September 2024.

[39] A. Zeybey, M. Ergezer, and T. Nguyen. Gaussian Splatting Under Attack: Investigating Adversarial Noise in 3D Objects. In *Neurips Safe Generative AI Workshop 2024*, 2024.

[40] J. Zheng, C. Lin, J. Sun, Z. Zhao, Q. Li, and C. Shen. Physical 3D Adversarial Attacks against Monocular Depth Estimation in Autonomous Driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24452–24461, June 2024. doi: 10.1145/3674029.367405.

[41] J. Zhou, L. Lyu, D. He, and Y. LI. RAUCA: A Novel Physical Adversarial Attack on Vehicle Detectors via Robust and Accurate Camouflage Generation. In *Forty-first International Conference on Machine Learning*, 2024.

[42] X. Zhou, Z. Lin, X. Shan, Y. Wang, D. Sun, and M. Yang. DrivingGaussian: Composite Gaussian Splatting for Surrounding Dynamic Autonomous Driving Scenes. In *(CVPR)*, June 2024.

[43] S. Zhu, G. Wang, X. Kong, D. Kong, and H. Wang. 3D Gaussian Splatting in Robotics: A Survey, December 2024. URL https://arxiv.org/abs/2410.12262. arXiv:2410.12262 [cs].