4DGUARD: TOWARDS PROACTIVE 4D GAUSSIAN ASSETS PROTECTION FOR DYNAMIC NOVEL-VIEW SYNTHESIS

Anonymous authors
Paper under double-blind review

ABSTRACT

Dynamic objects reconstructed as 4D Gaussian Splatting (4DGS) assets are valuable for simulation and content creation, yet they are shared as rendered videos from which anyone can segment, reconstruct, and reuse the object. We study *proactive* protection of 4D Gaussian assets: imperceptibly modifying the asset so that the target object cannot be reliably segmented in any video rendered from it. Beyond protecting images or static scenes, this raises two challenges: the protection must hold at every renderable time step, including those not seen during optimization, and it must survive video post-processing such as compression and frame averaging. We propose `4DGuard`, which embeds adversarial perturbations into the asset itself rather than into individual frames. `4DGuard` perturbs the appearance and opacity of the target Gaussians before deformation and lets the perturbation vary smoothly over time, so that every rendered frame carries it. To survive post-processing, it optimizes the perturbation against a promptable segmenter under a spatio-temporal expectation over transformations that covers compression and frame averaging, and a global calibration bounds the change of every released frame. Experiments on real-world dynamic scenes show that `4DGuard` keeps the released video visually unchanged while disrupting target segmentation, generalizes to unoptimized time steps, transfers to unseen segmenters, and retains its effect under compression.

1 INTRODUCTION

Gaussian splatting has become an asset format for photorealistic simulation: Gaussian scenes can be exported from world models and rendered in Isaac Sim, and real-to-sim pipelines rebuild objects and robots as Gaussian assets (World Labs, 2025; NVIDIA, 2026; Qureshi et al., 2024; Li et al., 2024; Abou-Chakra et al., 2025). 4D Gaussian Splatting (4DGS) extends such assets to moving objects by pairing canonical Gaussians with a deformation field (Wu et al., 2024; Yang et al., 2024). Building a 4D asset takes capture, reconstruction, and cleanup, yet the asset is usually shared as rendered video, and every released video provides the observations needed to rebuild it.

We consider two roles (Figure 1). The *owner* holds the 4D asset and a few masks of the target object, protects the asset once, and releases only videos rendered from the protected asset. The *recipient* receives the released, possibly re-encoded video and tries to obtain a reusable copy of the object: target-specific pipelines segment the object in every frame with a promptable segmenter and lift the masks into 3D/4D to isolate and reconstruct it (Ye et al., 2024; Li et al., 2025). The recipient may compress or average the frames and may use a segmenter that the owner has not seen.

Existing protection of Gaussian assets is mainly passive: watermarks verify ownership after misuse but do not hinder extraction (Huang et al., 2024; Chen et al., 2025; Jang et al., 2025; Lee et al., 2026). Proactive protection perturbs content so that a recipient's model fails, as for images (Salman et al., 2023; Zhang et al., 2023) and static 3DGS (Hong et al., 2025). Dynamic assets raise two further challenges. **(i) Time generalization.** The asset can be rendered at any time step, so the protection must be carried by the asset rather than by a fixed set of frames, while per-frame perturbations are inconsistent across time and cannot be stored in one set of Gaussians. **(ii) Persistence under video post-processing.** Released videos are compressed, and adjacent frames can be averaged to suppress

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.

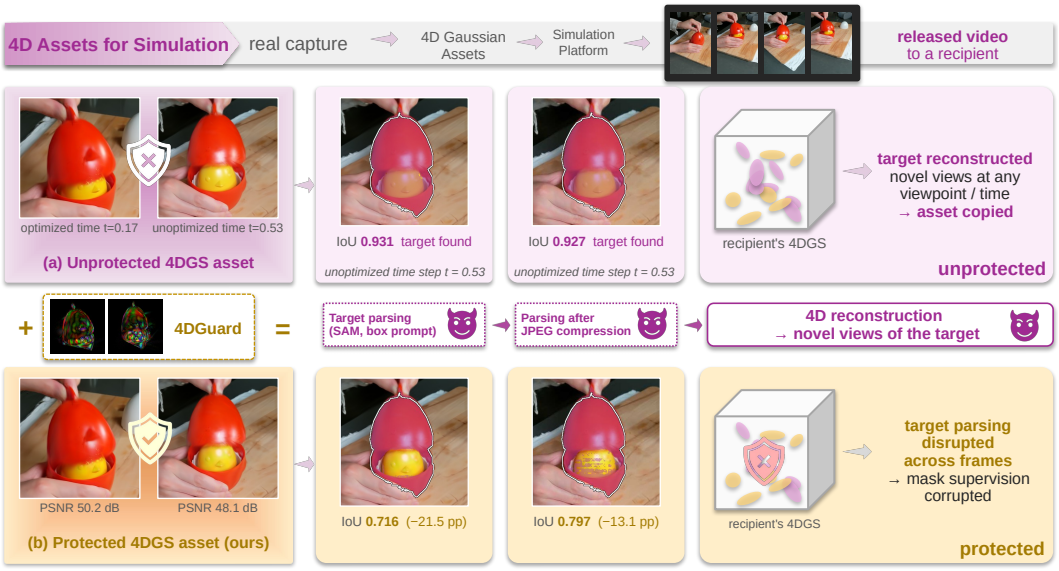
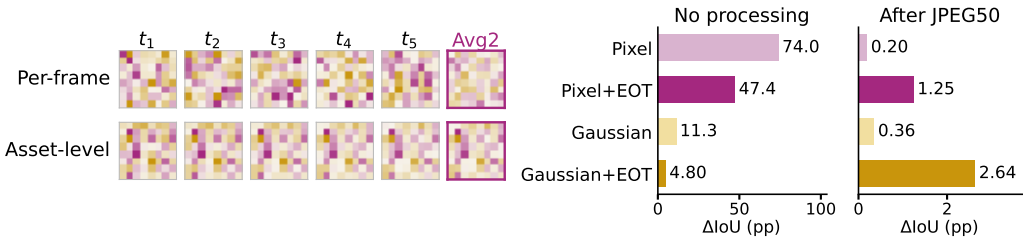


Figure 1: **Proactive protection of a 4D Gaussian asset.** The owner protects the asset once; every video rendered from it keeps its appearance, while SAM loses part of the target, also at an unoptimized time step and after JPEG compression.



(a) Time generalization **(b) Persistence under compression**

Figure 2: **Two challenges of protecting dynamic assets.** (a) Per-frame perturbations are independent across time and fade under frame averaging, whereas an asset-level perturbation stays consistent. (b) Segmentation disruption (ΔIoU, pp) of pixel and Gaussian perturbations with and without EOT, before and after JPEG50 (Chickchicken).

per-frame noise. Image-space perturbations are fragile under both: a pixel perturbation that removes 74 pp of target IoU keeps only 0.2 pp after JPEG compression (Figure 2).

We propose `4DGuard`, which embeds the adversarial perturbation into the asset. First, an *asset-level perturbation* modifies the appearance (spherical-harmonic colors) and opacity of the target Gaussians before deformation. Since the deformation field moves only geometry, every rendered frame at every viewpoint and time carries the perturbation; a few smooth temporal basis functions let it change slowly over time. Second, a *spatio-temporal expectation over transformations (EOT)* optimizes the perturbation against a frozen SAM segmenter through JPEG compression, frame averaging, and their combination, and suppresses all of SAM's mask candidates. Finally, a *global calibration* rescales the whole perturbation once so that every released frame, including frames not used in optimization, stays within a strict pixel budget. We target the segmentation step because it is the common entry point of target-specific reconstruction and editing (Ye et al., 2024; Chen et al., 2024; Li et al., 2025), it keeps the protection on the target object, and it can be measured on every frame. Our contributions are:

- We formulate proactive protection of 4D Gaussian assets released as video, with an explicit owner–recipient threat model and two challenges: time generalization and persistence under post-processing.

- We propose 4DGuard, which combines an asset-level perturbation with a smooth temporal basis, a spatio-temporal EOT objective, and a global calibration of the released video.
- Experiments on real-world dynamic scenes show that 4DGuard disrupts target segmentation under white-box access, including unoptimized time steps and compressed video, and transfers to unseen segmenters under black-box access, while the released video stays visually unchanged. An end-to-end evaluation of the recipient's reconstruction and extraction further delimits where this protection currently applies.

2 Related Work

Dynamic novel view synthesis. NeRF-based methods model dynamic scenes with a canonical radiance field and a time-dependent deformation, synthesizing novel views of moving content from monocular or multi-view video (Pumarola et al., 2021; Park et al., 2021; Cao & Johnson, 2023). 3DGS replaces the radiance field with explicit Gaussians rendered in real time (Kerbl et al., 2023), and 4DGS and deformable 3DGS add a deformation field that moves canonical Gaussians to each time step (Wu et al., 2024; Yang et al., 2024). Target-specific pipelines isolate, edit, or query individual objects in these representations by lifting per-frame masks from a promptable segmenter (Ye et al., 2024; Chen et al., 2024; Li et al., 2025); this segmentation step is the entry point that 4DGuard targets.

IP protection for 3D/4D assets. GaussianMarker, GuardSplat, 3D-GSW, and GS-Hider embed watermarks or hidden messages in static Gaussian assets (Huang et al., 2024; Chen et al., 2025; Jang et al., 2025; Zhang et al., 2024), and Mark4D watermarks dynamic Gaussian assets (Lee et al., 2026). These methods verify ownership after misuse. Among proactive methods, AdLift lifts adversarial perturbations into static 3DGS against instruction-driven editing (Hong et al., 2025), and Shielding the Unseen perturbs training views of static scenes to degrade NeRF fitting (Wu et al., 2023). 4DGuard protects dynamic assets, whose perturbation must reach every time step through the deformation field and survive video post-processing.

Adversarial protection and transformation robustness. PhotoGuard, Anti-DreamBooth, Glaze, and Mist protect images against editing and mimicry (Salman et al., 2023; Van Le et al., 2023; Shan et al., 2023; Liang & Wu, 2023), and Attack-SAM and related work disrupt promptable segmentation (Zhang et al., 2023; Lu et al., 2024; Wang et al., 2023). Compression weakens image perturbations (Sandoval-Segura et al., 2023), which motivates EOT (Athalye et al., 2018b; Shin & Song, 2017). Our EOT also averages adjacent rendered frames, so each transformed input depends on a short video window rather than one image.

3 Method

4DGuard protects a 4D Gaussian asset by storing an adversarial perturbation in the asset itself (Figure 3). We first formalize the setting (Section 3.1), then describe how the perturbation is attached to the asset (Section 3.2), how it is optimized to survive video post-processing (Section 3.3), and how the whole release is kept imperceptible (Section 3.4).

3.1 Preliminaries and Problem Formulation

4D Gaussian Splatting. 3DGS (Kerbl et al., 2023) represents a scene with Gaussians $g_i = (\mu_i, \Sigma_i, \alpha_i, \mathbf{c}_i)$, i.e., a mean, a covariance, an opacity, and spherical-harmonic (SH) color coefficients. A pixel is rendered by sorting the projected Gaussians by depth and blending their colors, $C = \sum_i \mathbf{c}_i \alpha'_i \prod_{j<i}(1-\alpha'_j)$, where α'_i is the opacity weighted by the projected 2D Gaussian. 4DGS (Wu et al., 2024) stores each Gaussian once in a shared reference state, the *canonical space*, and a deformation field Φ (Cao & Johnson, 2023) predicts how each Gaussian moves at time t. We write the asset as $\mathcal{G} = (\{g_i\}, \Phi)$ and its 8-bit render at viewpoint v and time t as $x = Q(\mathcal{R}(\mathcal{G}; v, t))$.

Threat model. The *owner* holds a clean asset $\mathcal{G}^{\text{raw}}$, its cameras, and target masks $\{M_t\}_{t\in\mathcal{A}}$ on a small set $\mathcal{A}$ of annotated frames. It publishes only the frames $x_t^{\text{prot}} = Q(\mathcal{R}(\mathcal{G}^{\text{prot}}; v_t, t))$ of a release $\mathcal{F}_{\text{rel}} = \{(v_t, t)\}$ rendered from the protected asset $\mathcal{G}^{\text{prot}}$, and keeps $\mathcal{G}^{\text{prot}}$ private. The *recipient* receives these frames, may compress or temporally average them, and segments the target with a

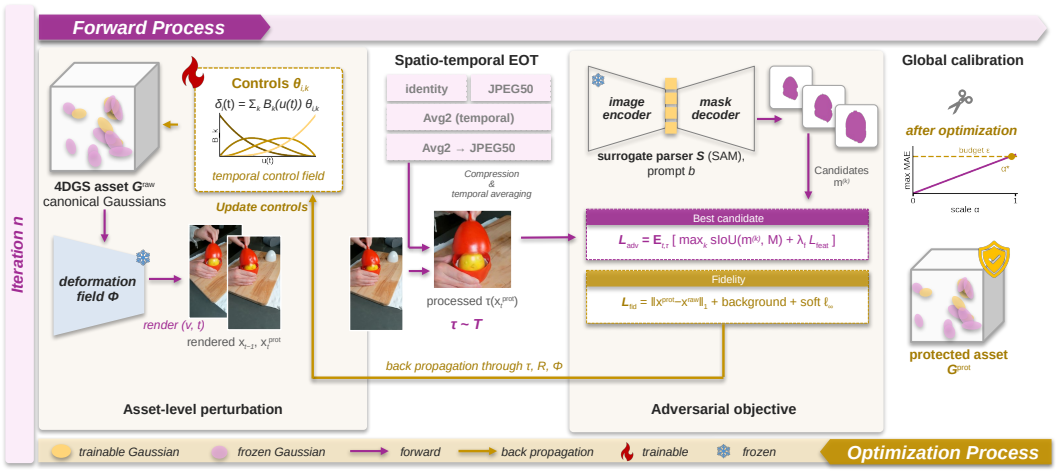


Figure 3: **Overview of 4DGuard.** The perturbation is stored on the target Gaussians and rendered through the frozen deformation field; spatio-temporal EOT transforms the rendered frames before the frozen SAM; the adversarial and fidelity losses update only the perturbation, and a global calibration rescales it for the whole release.

promptable segmenter before lifting the masks into its own 3D/4D model (Ye et al., 2024; Li et al., 2025). The owner has white-box access to one surrogate segmenter $\mathcal{S}$ (SAM (Kirillov et al., 2023)) but controls neither the recipient's segmenter nor its post-processing.

Problem formulation. Given a box prompt b_t, SAM returns three candidate masks, and the recipient can keep the best one. We therefore measure parsing quality by the best candidate, $\mathrm{IoU}^\star(x_t) = \max_k \mathrm{IoU}(\mathcal{S}^{(k)}(x_t, b_t), M_t)$. With $\boldsymbol{\theta}$ the perturbation parameters stored in the asset and $\mathcal{T}$ a set of video transformations, the owner solves

$$\min_{\boldsymbol{\theta}} \; \mathbb{E}_{t\sim\mathcal{A}}\, \mathbb{E}_{\tau\sim\mathcal{T}} \Big[\mathrm{IoU}^\star\big(\tau(\mathbf{x}^{\mathrm{prot}})_t\big) \Big] \quad \text{s.t.} \quad \tfrac{1}{3HW} \big\| x_t^{\mathrm{prot}} - x_t^{\mathrm{raw}} \big\|_1 \le \epsilon \quad \forall\, (v_t, t) \in \mathcal{F}_{\mathrm{rel}}, \tag{1}$$

where $\tau(\mathbf{x})_t$ is frame t of the transformed video. Compared with image protection, the variable $\boldsymbol{\theta}$ is shared by all frames, the objective involves several frames at once through temporal transformations, and the constraint must hold on every released frame, not only on those used for optimization. The next three subsections address these three differences.

3.2 Asset-Level Perturbation

A perturbation optimized on a frame protects only that frame. A 4D asset, however, renders every frame from the same canonical Gaussians, so a perturbation stored on these Gaussians appears in every rendered frame, including time steps and viewpoints that were not optimized.

We perturb only the Gaussians that belong to the target object. Each Gaussian center is moved by Φ to every annotated time $t \in \mathcal{A}$ and projected into that frame; a Gaussian enters $\mathcal{G}_{\mathrm{tar}}$ if at least a fraction $\rho = 0.7$ of its valid projections falls inside M_t, with valid projections in at least two frames. The background, which is often shared with other content, is left untouched.

Perturbation with a smooth temporal basis. For $g_i \in \mathcal{G}_{\mathrm{tar}}$, let $\mathbf{a}_i^0$ collect its canonical SH coefficients and opacity logit. We add the perturbation to $\mathbf{a}_i^0$ *before* deformation and let it change smoothly over the sequence:

$$\mathbf{a}_i(t) = \mathbf{a}_i^0 + \sum_{k=0}^{K-1} B_k\big(u(t)\big)\, \boldsymbol{\theta}_{i,k}, \qquad B_k(u) = \tbinom{K-1}{k} (1-u)^{K-1-k} u^k, \tag{2}$$

where $u(t) = (t - t_{\min})/(t_{\max} - t_{\min})$ is the normalized timestamp. The Bernstein polynomials B_k are non-negative and sum to one, so the perturbation at any time is a convex combination of the K control vectors $\boldsymbol{\theta}_{i,k}$: it is bounded by them and varies smoothly with t. $K = 1$ yields one constant perturbation for the whole sequence, and $K = 4$ a cubic curve that can adapt to changes of the object's appearance. The number of parameters, $K \cdot |\mathcal{G}_{\mathrm{tar}}|$ control vectors, does not grow with the length of the video.

In our assets, Φ deforms positions, scales, and rotations but not colors or opacities. The perturbed appearance $\mathbf{a}_i(t)$ is therefore carried unchanged to every time step, and it moves with the object because the geometry still follows Φ. All canonical geometry and the deformation network stay frozen, so the motion of the asset is preserved exactly.

3.3 Spatio-Temporal EOT

The recipient rarely sees the raw render. Videos are compressed, and adjacent frames can be averaged to suppress frame-specific noise. A perturbation must therefore be optimized through these operations, and it must defeat every mask candidate that SAM returns rather than a single one.

Multi-candidate adversarial loss. For a processed frame $\tilde{x}_t$, SAM outputs the logits $\{z_t^{(k)}\}_{k=1}^{3}$ of its three candidates. We minimize the soft IoU of the currently best candidate,

$$\mathcal{L}_{\text{tar}}(\tilde{x}_t) = \max_{k=1,2,3} \text{sIoU}\big(\sigma(z_t^{(k)}), M_t\big), \qquad \text{sIoU}(p, M) = \frac{\sum p \odot M}{\sum p + \sum M - \sum p \odot M}, \tag{3}$$

so that the gradient always acts on the currently most accurate candidate. We further push the image features of the target region away from their clean counterparts. With e_p the SAM encoder feature at grid location p and m_p the target mask on that grid,

$$\mathcal{L}_{\text{feat}}(\tilde{x}_t^{\text{prot}}, \tilde{x}_t^{\text{raw}}) = \frac{\sum_p m_p \cos\big(e_p(\tilde{x}_t^{\text{prot}}), e_p(\tilde{x}_t^{\text{raw}})\big)}{\sum_p m_p}, \tag{4}$$

where the clean frame undergoes the same transformation as the protected one.

Spatio-temporal transformations. Standard EOT (Athalye et al., 2018b) averages the loss over image transformations. We add transformations across time,

$$\mathcal{T} = \{\text{id},\ \text{JPEG50},\ \text{Avg2},\ \text{JPEG50} \circ \text{Avg2}\}, \qquad \text{Avg2}(\mathbf{x})_t = \big\lfloor \tfrac{1}{2}(x_t + x_{t-1}) \big\rfloor, \tag{5}$$

where JPEG50 is the JPEG codec at quality 50 and x_{t-1} is the preceding frame of the release, rendered from the protected asset at its own camera and time. Averaging two rendered frames attenuates any component of the perturbation that changes from frame to frame; optimizing through Avg2 therefore favors perturbations that remain consistent over time, which the asset-level parameterization makes possible. The parsing objective is

$$\mathcal{L}_{\text{adv}} = \mathbb{E}_{t\sim\mathcal{A}}\, \mathbb{E}_{\tau\sim\mathcal{T}} \Big[\lambda_{\text{p}} \mathcal{L}_{\text{tar}}\big(\tau(\mathbf{x}^{\text{prot}})_t\big) + \lambda_{\text{f}} \mathcal{L}_{\text{feat}}\big(\tau(\mathbf{x}^{\text{prot}})_t, \tau(\mathbf{x}^{\text{raw}})_t\big) \Big], \tag{6}$$

with $\lambda_{\text{p}} = 4$ and $\lambda_{\text{f}} = 0.5$. The expectations follow a fixed cyclic schedule that visits every optimization frame with every transformation equally often. Quantization, flooring, and JPEG are not differentiable; we use straight-through estimators (Athalye et al., 2018a), so gradients reach the perturbation of both averaged frames through the renderer.

3.4 Optimization and Calibration

Perturbation lives in the asset and appears in every rendered frame, including frames that were not used in optimization, so a budget checked only on the optimized frames does not guaranty the quality of the released video. `4DGuard` therefore controls the distortion at three levels: a soft loss during optimization, a projection after every update, and a calibration on the whole release.

Fidelity loss. For an optimization frame with $N = 3HW$ RGB values, let $d_j = |x_j^{\text{prot}} - x_j^{\text{raw}}|$. We penalize the average change, weighting background pixels four times as heavily ($w_j = 1 + 3(1 - M_j)$), and large single-pixel changes above $\kappa = 8/255$:

$$\mathcal{L}_{\text{fid}} = \frac{8}{N} \sum_j w_j d_j + \frac{0.05 \cdot 255^2}{N} \sum_j \big[d_j - \kappa\big]_+^2. \tag{7}$$

A small ℓ_2 regularizer $\mathcal{L}_{\text{reg}}$ on the SH and opacity changes completes the training loss $\mathcal{L}_{\text{adv}} + \mathbb{E}_t[\mathcal{L}_{\text{fid}} + \mathcal{L}_{\text{reg}}]$.

Per-step projection. After every Adam update, we multiply all perturbation parameters by one common factor $\alpha \in [0, 1]$, chosen by bisection so that every optimization frame satisfies the budget

Algorithm 1 `4DGuard`: asset-level protection with spatio-temporal EOT

Require: clean asset $\mathcal{G}^{\text{raw}} = (\{g_i\}, \Phi)$; masks and boxes $\{M_t, b_t\}_{t\in\mathcal{A}}$; release $\mathcal{F}_{\text{rel}}$; surrogate $\mathcal{S}$; transformations $\mathcal{T}$; budget ϵ; basis size K; steps S
Ensure: protected asset $\mathcal{G}^{\text{prot}}$ and its released frames

$\mathcal{G}_{\text{tar}} \leftarrow \textsc{SelectTarget}(\mathcal{G}^{\text{raw}}, \{M_t\})$; $\boldsymbol{\theta} \leftarrow \mathbf{0}$
for $s = 1, \dots, S$ **do**
  draw a batch of pairs (t, τ) from the cyclic schedule over $\mathcal{A} \times \mathcal{T}$
  render the frames required by τ with Equation (2) and apply τ
  $\mathcal{L} \leftarrow \mathcal{L}_{\text{adv}} + \mathcal{L}_{\text{fid}} + \mathcal{L}_{\text{reg}}$ ▷ Equations (6) and (7)
  $\boldsymbol{\theta} \leftarrow \textsc{Adam}(\boldsymbol{\theta}, \nabla_{\boldsymbol{\theta}}\mathcal{L})$
  $\boldsymbol{\theta} \leftarrow \alpha\,\boldsymbol{\theta}$ with $\alpha \leftarrow \textsc{FeasibleScale}(\boldsymbol{\theta}, \mathcal{A})$ ▷ per-step projection
end for
$\boldsymbol{\theta} \leftarrow \hat{\alpha}\,\boldsymbol{\theta}$ with $\hat{\alpha} \leftarrow \textsc{FeasibleScale}(\boldsymbol{\theta}, \mathcal{F}_{\text{rel}})$ ▷ global calibration
return $\mathcal{G}^{\text{prot}} = (\{g_i \oplus \boldsymbol{\theta}\}, \Phi)$ and $\{Q(\mathcal{R}(\mathcal{G}^{\text{prot}}; v_t, t))\}_{(v_t,t)\in\mathcal{F}_{\text{rel}}}$

of Equation (1) with $\epsilon = 0.35/255$. A common factor keeps the relative structure of the perturbation across Gaussians and time.

Global calibration. After optimization, we render every frame of the release and search, again by bisection, for one scale $\hat{\alpha} \in [0, 1]$ of the whole perturbation that satisfies

$$\max_{(v_t,t)\in\mathcal{F}_{\text{rel}}} \tfrac{1}{3HW} \left\| Q\big(\mathcal{R}(\mathcal{G}_{\hat{\alpha}\boldsymbol{\theta}}; v_t, t)\big) - x_t^{\text{raw}} \right\|_1 \le \epsilon. \tag{8}$$

Because a single scale acts on the asset, the calibrated perturbation stays consistent across time, and the worst frame of the release, whether optimized or not, meets the budget. Each candidate $\hat{\alpha}$ is verified by re-rendering, since opacity and quantization can make the distortion non-monotone in $\hat{\alpha}$. Algorithm 1 summarizes the procedure.

4 Experiments

4.1 Experimental Setup

Datasets. We protect three real-world HyperNeRF scenes (Park et al., 2021): *Chickchicken* (a hand opening a toy container), *Americano*, and *Split-cookie*. Each asset is optimized on a subset of annotated frames; a held-out block of annotated time steps (*unoptimized times*) tests time generalization. For recipient-side evaluation we also use all eight D-NeRF scenes (Pumarola et al., 2021). Splits and implementation details are in Sections A and F.

Metrics. Following the threat model, we prompt SAM ViT-H with the target box and report ΔIoU, the drop of the best-candidate IoU from the clean to the protected release (in percentage points, higher is better), with both videos processed identically. Post-processing includes JPEG at quality 50/75, averaging of two or three adjacent frames (Avg2/Avg3), and Avg2 followed by JPEG50; JPEG75 and Avg3 are unseen during optimization. Fidelity is the per-frame mean absolute error (MAE, in $1/255$) over all released frames.

Baselines. We compare with *Pixel* perturbations and *Per-frame Gaussian* perturbations, which optimize independent Gaussian perturbations for every frame with the same objective and budget. `4DGuard` uses $K \in \{1, 4\}$ temporal basis functions with JPEG-only EOT (-J) or spatio-temporal EOT (-ST); the default is -ST with $K = 4$.

4.2 Main Results

Imperceptibility. All protected assets meet the budget on every released frame (maximum MAE $0.350/255$, mean 0.198–$0.212/255$ on Chickchicken), with PSNR above 48 dB on the sampled frames. As Figure 4 shows, the protected renders are imperceptibly different from the originals, and the residual stays on the moving target.

Segmentation disruption. Per-frame Gaussian perturbations reach 19.1 pp on the frames they are optimized for (Table 1), which shows that the target Gaussians can mislead SAM under the budget.

Table 1: ΔIoU (pp, $\uparrow$) of SAM ViT-H on Chickchicken and release MAE ($\times 1/255$, $\downarrow$). *: unseen during optimization; –: not applicable.

Frames	Method	MAE	None	JPEG50	JPEG75*	Avg2	Avg3*	Avg2→J50
Optimized	Per-frame Gaussian	0.340	19.10	12.54	13.49	–	–	–
	4DGuard-ST (K=1)	0.212	6.85	2.64	3.54	2.17	0.19	0.39
	4DGuard-J (K=4)	0.206	**9.11**	**4.45**	**5.02**	0.45	0.00	0.25
	4DGuard-ST (K=4)	**0.198**	7.25	3.43	3.81	**3.22**	**0.20**	**0.63**
Unoptimized	4DGuard-ST (K=1)		3.96	**1.93**	**2.17**	**0.47**	**0.45**	**0.07**
	4DGuard-J (K=4)		**4.31**	0.27	1.83	0.17	0.19	0.02
	4DGuard-ST (K=4)		3.91	1.41	2.03	0.32	**0.45**	0.05

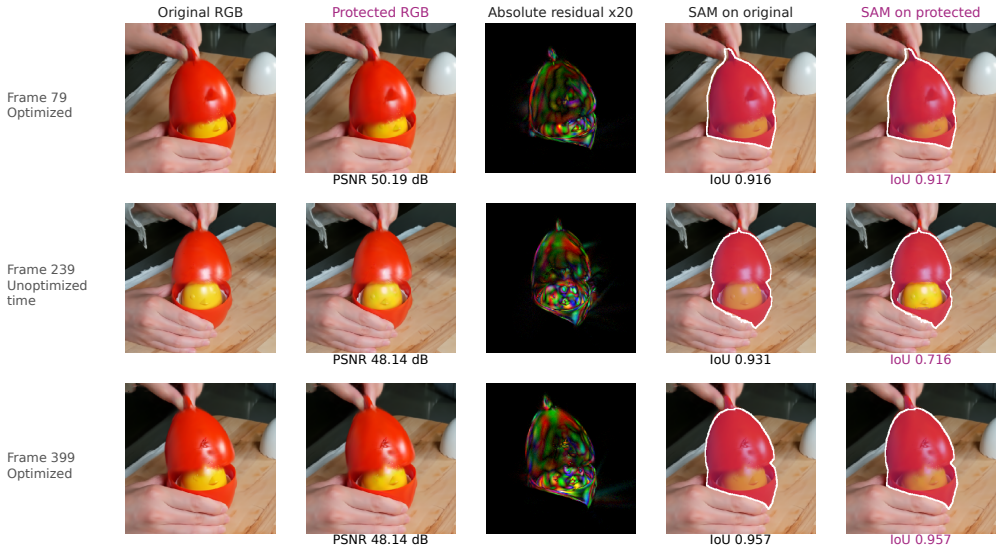


Figure 4: **Qualitative results on Chickchicken.** Original and protected renders, residual ($\times 20$), and SAM predictions (color) against the annotation (white).

4DGuard obtains 6.9–9.1 pp with a single perturbation shared by the whole sequence, and exceeds 10 pp on 10 of the 34 optimized frames (up to 26.5 pp; Section B).

Time generalization. The perturbation also transfers to time steps that were excluded from optimization, with 3.9–4.3 pp on average and up to 21.5 pp on a single frame (frame 239, where SAM misses part of the open container; Figure 4), whereas per-frame perturbations are undefined there. The asset-level perturbation thus carries the protection to new time steps.

Per-frame analysis. Table 2 shows that the average hides strong per-frame effects: the default variant lowers the IoU by more than 10 pp on 10 optimized frames and by up to 26.5 pp, and the strongest frames remain above 20 pp after JPEG compression. Figure 5 compares the optimization of per-frame and shared perturbations under the same number of updates per frame: per-frame perturbations keep improving on the frames they own, whereas the shared perturbation reaches a similar effect early and spreads it over the whole timeline.

4.3 Robustness

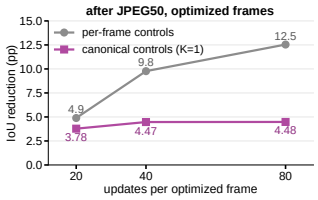
Spatio-temporal EOT. Table 5 and Figure 2(b) isolate the role of the perturbation space and EOT. Pixel perturbations are strongest on raw frames but lose almost all of their effect after JPEG50, and pixel-space EOT recovers little. Gaussian perturbations with EOT keep the largest effect after JPEG50 and under the unseen JPEG75. Adding frame averaging to EOT raises the effect under Avg2 from 0.45 to 3.22 pp (Table 1, -J vs. -ST, K=4), at a small cost under JPEG50. Averaging over three frames or combining averaging with JPEG remains the hardest setting and removes most of the effect.

Unseen segmenters. Table 3 evaluates the same releases with four segmenters not used in optimization. On Split-cookie, the perturbation optimized against SAM ViT-H transfers to all of them

Table 2: Per-frame view of Chickchicken: maximum ΔIoU (pp) / number of frames with ΔIoU > 10 pp.

Frames	Variant	None	JPEG50	JPEG75*	Avg2	Avg3*	A2→J50
Opt. (34)	-ST (K=1)	24.2/10	23.4/4	22.8/7	23.0/3	2.6/0	8.0/0
	-J (K=4)	26.3/14	24.5/7	23.6/9	11.5/1	1.6/0	5.5/0
	-ST (K=4)	26.5/10	24.5/6	23.3/6	23.6/5	2.5/0	8.0/0
Unopt. (10)	-ST (K=1)	22.2/2	18.2/1	20.0/1	1.9/0	3.8/0	0.8/0
	-J (K=4)	21.7/2	1.9/0	17.4/1	1.2/0	1.8/0	0.3/0
	-ST (K=4)	21.5/2	13.1/1	18.7/1	1.2/0	3.8/0	0.4/0

Figure 5: ΔIoU after JPEG50 vs. updates per frame.



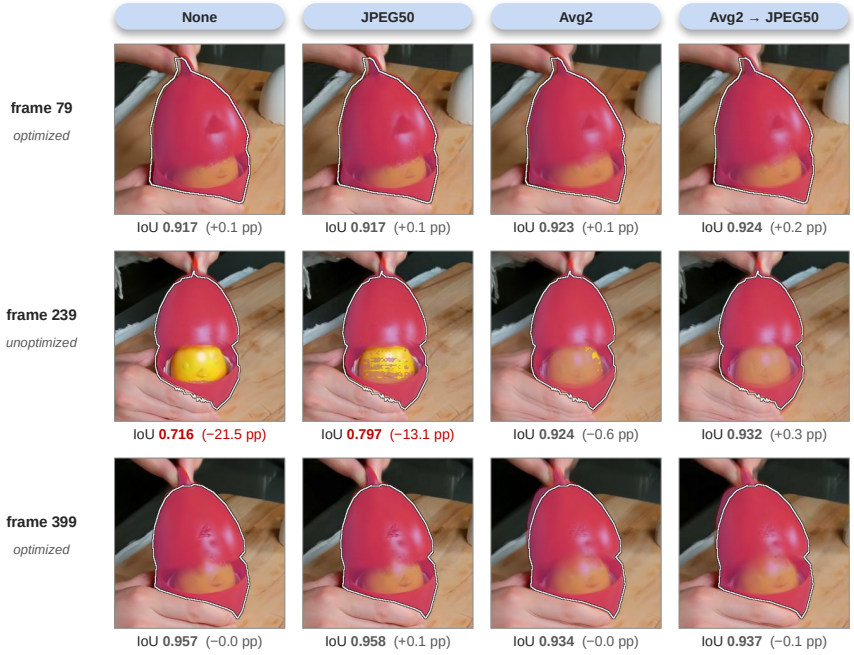


Figure 6: Parsing of the default `4DGuard` under post-processing on frames 79, 239, and 399 (best candidate in color, annotation in white).

(16.4–40.2 pp) and largely persists after JPEG50. On unoptimized times, ViT-L, ViT-B, and MobileSAM are affected even more than the surrogate. On Americano, the effect is small for every segmenter, so the protection strength depends on the scene.

4.4 What Does the Recipient Obtain?

We finally evaluate the asset that the recipient obtains, comparing `4DGuard` with random and smooth perturbations of similar distortion. An *RGB-only* recipient fits a fresh 4DGS to the released frames without masks; a *mask-guided* recipient additionally propagates SAM 2.1 video masks and extracts the Gaussians assigned to the target (Section G). The RGB-only recipient bypasses segmentation, and its reconstructions on the eight D-NeRF scenes have the same quality for all releases (Table 6 and Figure 12; renders in Figure 14). The mask-guided recipient still extracts a recognizable object (Figure 7); its silhouette IoU decreases slightly, by a similar amount as under ordinary perturbations. Swapping protected masks for clean ones on the same reconstruction recovers part of this drop (Figure 12, top right), so the disrupted masks reach the extraction stage, but their effect on the final object is small. Protecting the extracted asset itself therefore calls for objectives on the recipient's reconstruction, a direction our evaluation protocol makes measurable.

Ablation Study. Suppressing the best of SAM's three candidates instead of a single mask output more than triples the disruption, and the gain grows after compression (11.78 vs. 2.51 pp after JPEG50; Table 4). The improvement also holds under the unseen JPEG75 (12.09 vs. 2.54 pp), since an attack on a single output leaves the other candidates correct. Pixel perturbations keep only 0.3% of their effect

Table 3: Transfer to unseen segmenters: ΔIoU (pp); ViT-H is the surrogate.

	Optimized		Unoptimized	
Segmenter	None	JPEG50	None	JPEG50
Split-cookie				
SAM ViT-H	60.8	26.6	2.0	5.2
SAM ViT-L	40.2	18.0	7.6	8.4
SAM ViT-B	29.7	17.1	9.7	11.0
SAM 2.1-L	16.4	3.1	0.7	1.0
MobileSAM	21.9	18.6	12.3	11.3
Americano				
SAM ViT-H	0.9	0.9	0.1	0.3
Unseen (4)	0.5–2.1	0.6–1.1	0.0–0.2	0.2–1.3

Table 4: SAM objective: single mask vs. best of three candidates (ΔIoU, pp).

Objective	None	JPEG50	JPEG75*
Single mask	5.00	2.51	2.54
Best of three	**18.58**	**11.78**	**12.09**

Table 5: Perturbation space and EOT (ΔIoU, pp).

Space	EOT	None	JPEG50	JPEG75*
Pixel	–	74.04	0.20	0.56
Pixel	JPEG	47.40	1.25	0.99
Gaussian	–	11.33	0.36	0.54
Gaussian	JPEG	4.80	**2.64**	**2.43**

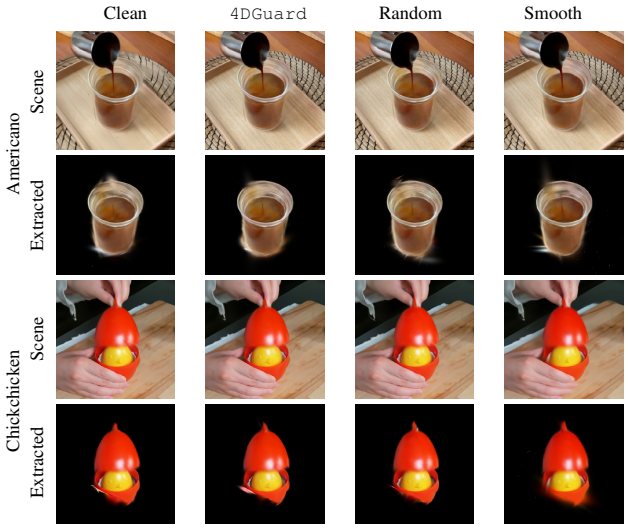


Figure 7: Recipient's reconstructed scene and extracted target under each release.

Table 6: Recipient-side quality (higher means better reconstruction for the recipient).

	Extraction IoU		RGB PSNR
Release	Amer.	Chick.†	D-NeRF
Clean	0.940	0.860	24.56
Random	0.935	0.854	24.53
Smooth	0.931	0.838	24.54
4DGuard	0.930	0.839	24.57

†Paired runs; see Section G.

after JPEG50, whereas Gaussian perturbations optimized with EOT keep 55% (Table 5, Figure 2(b)). EOT thus trades part of the raw effect for robustness, and this trade pays off only with the asset-level parameterization. Adding frame averaging to EOT raises the effect under Avg2 from 0.45 to 3.22 pp and, on unoptimized times, after JPEG50 from 0.27 to 1.41 pp (Table 1). The cost is moderate under JPEG50 on optimized frames (4.45 vs. 3.43 pp), while every processed condition on

5 Conclusion

We presented 4DGuard, a proactive protection for 4D Gaussian assets released as video. Instead of perturbing individual frames, 4DGuard stores the perturbation in the canonical appearance of the target Gaussians, so that every frame rendered from the asset carries it at any viewpoint and time. By optimizing this asset-level perturbation with spatio-temporal EOT, 4DGuard disrupts target segmentation across time steps, transfers to unseen segmenters, and retains its effect under compression, while a global calibration keeps every released frame within a strict imperceptibility budget. More broadly, our results show that segmentation-level protection of dynamic content can be carried by the asset itself rather than by individual released frames, and we hope 4DGuard serves as a step toward proactive safeguards for 4D assets in simulation and content creation.

ETHICS STATEMENT

This work studies how an owner can proactively protect a 4D Gaussian asset that is released as rendered video. The perturbations are applied only to assets the owner controls, and all experiments use publicly available research datasets.

AI USE STATEMENT

Generative AI tools assisted with refining and polishing the text. They were not used to synthesize benchmark observations, ground truth, or reported experimental results. Numerical claims are checked against saved experiment records. The authors remain responsible for verifying the code, scientific claims, citations, and manuscript.

REPRODUCIBILITY STATEMENT

Details of the datasets, the surrogate parser, the optimization hyperparameters, the transformation set, and the recipient protocols are given in Section 4.1, Section A (including implementation-aligned pseudocode), Section F, and Section G. All reported numbers are computed from archived per-frame measurements, and the complete tables for every processing condition are provided in Section B. Code and the protected assets will be released upon publication.

REFERENCES

Jad Abou-Chakra, Lingfeng Sun, Krishan Rana, Brandon May, Karl Schmeckpeper, Niko Suenderhauf, Maria Vittoria Minniti, and Laura Herlant. Real-is-Sim: Bridging the sim-to-real gap with a dynamic digital twin. *arXiv preprint arXiv:2504.03597*, 2025.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 274–283, 2018a.

Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 284–293, 2018b.

Ang Cao and Justin Johnson. HexPlane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 130–141, 2023.

Minghao Chen, Iro Laina, and Andrea Vedaldi. DGE: Direct gaussian 3D editing by consistent multi-view editing. In *European Conference on Computer Vision*, pp. 74–92, 2024.

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

Ziming Hong, Tianyu Huang, Runnan Chen, Shanshan Ye, Mingming Gong, Bo Han, and Tongliang Liu. AdLift: Lifting adversarial perturbations to safeguard 3D gaussian splatting assets against instruction-driven editing. *arXiv preprint arXiv:2512.07247*, 2025.

Xiufeng Huang, Ruiqi Li, Yiu-ming Cheung, Ka Chun Cheung, Simon See, and Renjie Wan. GaussianMarker: Uncertainty-aware copyright protection of 3D gaussian splatting. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

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

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4015–4026, 2023.

Jaejin Lee, Minjae Jeong, Joonhyuk Park, Yechan Hwang, Seunghun Baek, and Won Hwa Kim. Mark4D: Temporally-consistent watermarking for 4D Gaussian Splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 39393–39402, June 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Lee_Mark4D_Temporally-Consistent_Watermarking_for_4D_Gaussian_Splatting_CVPR_2026_paper.html.

Wanhua Li, Renping Zhou, Jiawei Zhou, Yingwei Song, Johannes Herter, Minghan Qin, Gao Huang, and Hanspeter Pfister. 4D LangSplat: 4D language gaussian splatting via multimodal large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Xinhai Li, Jialin Li, Ziheng Zhang, Rui Zhang, Fan Jia, Tiancai Wang, Haoqiang Fan, Kuo-Kun Tseng, and Ruiping Wang. RoboGSim: A Real2Sim2Real robotic Gaussian splatting simulator. *arXiv preprint arXiv:2411.11839*, 2024.

Chumeng Liang and Xiaoyu Wu. Mist: Towards improved adversarial examples for diffusion models. *arXiv preprint arXiv:2305.12683*, 2023.

Jiahao Lu, Xingyi Yang, and Xinchao Wang. Unsegment anything by simulating deformation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

NVIDIA. Neural volume rendering. Isaac Sim Documentation, 2026. URL https://docs.isaacsim.omniverse.nvidia.com/latest/assets/usd_assets_nurec.html. Updated September 18, 2026. Accessed September 25, 2026.

Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. HyperNeRF: A higher-dimensional representation for topologically varying neural radiance fields. *ACM Transactions on Graphics*, 40(6), 2021.

Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10318–10327, 2021.

Mohammad Nomaan Qureshi, Sparsh Garg, Francisco Yandun, David Held, George Kantor, and Abhisesh Silwal. SplatSim: Zero-shot Sim2Real transfer of RGB manipulation policies using Gaussian splatting. *arXiv preprint arXiv:2409.10161*, 2024.

Hadi Salman, Alaa Khaddaj, Guillaume Leclerc, Andrew Ilyas, and Aleksander Madry. Raising the cost of malicious AI-powered image editing. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Pedro Sandoval-Segura, Jonas Geiping, and Tom Goldstein. JPEG compressed images can bypass protections against AI editing. *arXiv preprint arXiv:2304.02234*, 2023.

Shawn Shan, Jenna Cryan, Emily Wenger, Haitao Zheng, Rana Hanocka, and Ben Y. Zhao. Glaze: Protecting artists from style mimicry by text-to-image models. In *32nd USENIX Security Symposium*, pp. 2187–2204, 2023.

Richard Shin and Dawn Song. JPEG-resistant adversarial images. In *NIPS Workshop on Machine Learning and Computer Security*, 2017.

Thanh Van Le, Hao Phung, Thuan Hoang Nguyen, Quan Dao, Ngoc N. Tran, and Anh Tran. Anti-DreamBooth: Protecting users from personalized text-to-image synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2116–2127, 2023.

Yuqing Wang, Yun Zhao, and Linda Petzold. An empirical study on the robustness of the segment anything model (SAM). *arXiv preprint arXiv:2305.06422*, 2023.

World Labs. Marble: A multimodal world model. World Labs Blog, November 2025. URL https://www.worldlabs.ai/blog/marble-world-model. Published November 12, 2025. Accessed September 25, 2026.

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, 2024.

Yihan Wu, Brandon Y. Feng, and Heng Huang. Shielding the unseen: Privacy protection through poisoning NeRF with spatial deformation. *arXiv preprint arXiv:2310.03125*, 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, 2024.

Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3D scenes. In *European Conference on Computer Vision*, 2024.

Chenshuang Zhang, Chaoning Zhang, Taegoo Kang, Donghun Kim, Sung-Ho Bae, and In So Kweon. Attack-SAM: Towards attacking segment anything model with adversarial examples. *arXiv preprint arXiv:2305.00866*, 2023.

Xuanyu Zhang, Jiarui Meng, Runyi Li, Zhipei Xu, Yongbing Zhang, and Jian Zhang. GS-Hider: Hiding messages into 3D gaussian splatting. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

APPENDICES

- Section A: Implementation details, optimization flow, and pseudocode of `4DGuard`.
- Section B: Additional quantitative results.
- Section C: Additional qualitative results.
- Section D: Discussions.
- Section E: Limitations and future directions.
- Section F: Additional scenes, unseen segmenters, and RGB-only reconstruction on D-NeRF.
- Section G: Mask-guided recipient protocol and object-extraction results.

A IMPLEMENTATION DETAILS

This section first documents the three historical Chickchicken continuous-field arms: 34 optimization frames, 10 unoptimized annotated frames, and 1,360 Adam steps with two centers per step. The per-frame comparison and the head/space diagnostics use separate schedules specified below. Later multi-scene protocols appear in Section F.

Scenes and splits. Chickchicken has 456 moving-camera frames rendered at 536×960 from an asset with 135,683 canonical Gaussians. Of its 46 annotated frames, 34 are used for optimization and target selection, and the 10 annotated frames of a held-out central 100-frame block form the unoptimized-time set; twenty adjacent guard frames (two annotated) are excluded from optimization and from the main tables. The clean asset itself is fitted to the full sequence. Americano (526 frames) and Split-cookie (539 frames) use 38 optimization centers and 11 and 12 unoptimized-time frames (Section F.1). Recipient protocols are specified in Sections F.2 and G.1.

Fidelity loss. For an unprocessed optimization frame with $N = 3HW$ RGB elements, let $d_j = |x_j^{\text{prot}} - x_j^{\text{raw}}|$ and $w_j = 1 + 3(1 - M_j)$. The fidelity loss is $\mathcal{L}_{\text{fid}} = 8\mathcal{L}_1 + 0.05\mathcal{L}_\infty$ with $\mathcal{L}_1 = \frac{1}{N}\sum_j w_j d_j$ and $\mathcal{L}_\infty = \frac{255^2}{N}\sum_j [d_j - \kappa]_+^2$, $\kappa = 8/255$; the squared hinge is a soft penalty, not a hard maximum-pixel bound. The feature loss is $\mathcal{L}_{\text{feat}} = \sum_p m_p \cos_\eta(e_p^{\text{prot}}, e_p^{\text{raw}}) / \max(\sum_p m_p, 10^{-6})$ over SAM encoder features e_p at grid location p with target mask m_p, and the soft-IoU denominator in Equation (3) is clamped below at 10^{-6}.

Target Gaussian selection. Every canonical Gaussian is transported to each annotated frame by the deformation field and projected with the frame's camera. A Gaussian is added to $\mathcal{G}_{\text{tar}}$ when at least a fraction $\rho = 0.7$ of its valid in-frame projections falls inside the target mask and it has valid projections in at least two annotated frames. This projection test does not perform an occlusion-aware visibility test. Only the 34 optimized frames are used for selection, so no annotation of the unoptimized block enters support selection or the protection loss. The procedure selects 8,776 of the 135,683 canonical Gaussians.

Controls and rendering. The controls modify all SH coefficients (DC and higher orders) and the raw opacity logit of the target Gaussians before deformation. Canonical geometry and deformation weights remain frozen. The recorded configuration has `no_dx=no_ds=no_dr=False` and `no_do=no_dshs=True`: positions, scales, and rotations deform with time, while opacity and SH pass through the deformation layer unchanged. Thus the appearance controls remain bound to the moving primitives. The normalized camera time evaluates Equation (2) throughout the asset's time interval. Native-resolution renders are quantized to 8 bits before processing; only RGB is released.

The implementation stores absolute control values $C_k = \mathbf{a}^0 + \boldsymbol{\theta}_k$ and evaluates them as $C_0 + \sum_{k>0} B_k(u)(C_k - C_0)$, algebraically equivalent to Equation (2). A $K = 4$ internal asset requires the temporal-control payload and its loader in addition to the PLY and deformation weights.

Transformation set. JPEG uses the standard codec at quality 50. Two-frame temporal averaging combines the current frame with the preceding frame of the video (the first frame is averaged with its successor), and both frames are rendered from the protected asset at their actual cameras and timestamps. The integer-pixel mean is floored, and JPEG is applied afterward for Avg2→JPEG50. The backward pass uses the arithmetic-mean gradient for the source images and straight-through approximations for quantization, flooring, and JPEG. Three-frame averaging additionally includes the succeeding frame. JPEG75 and Avg3 are evaluated but are absent from this protocol's optimization

transformations. Each clean reference undergoes the same processing, so $\Delta\mathrm{IoU}$ measures a matched difference in parsing quality.

Surrogate parser protocol. We use SAM ViT-H with the ground-truth bounding box of the target dilated by 8 pixels as the prompt and request three candidate masks. Evaluation reports the best candidate IoU against the fixed target mask. During optimization, Equation (3) takes the maximum of three independently computed soft IoUs. The feature term minimizes target-weighted cosine similarity between clean and protected encoder features under the same transformation. The parser is frozen throughout, with loss weights $\lambda_{\mathrm{p}} = 4$ and $\lambda_{\mathrm{f}} = 0.5$.

For native size $H \times W$, let (n_h, n_w) be SAM's long-side-1024 resize. The feature mask is bilinearly resized to $\lceil n_h/16 \rceil \times \lceil n_w/16 \rceil$ and placed in the upper-left of a zero-padded 64×64 grid. The decoder target is nearest-neighbor resized to (n_h, n_w), padded to 1024^2, then nearest-neighbor resized to 256^2. Clean embeddings, feature and decoder masks, and resized boxes are cached in FP16 and converted to FP32 for the loss; the clean feature reference therefore includes this rounding. Cosine similarity divides by $\max(\|a\|_2, 10^{-8}) \max(\|b\|_2, 10^{-8})$. Training uses Torch bilinear image resizing; hard-mask evaluation uses SamPredictor's preprocessing.

Optimization schedule and global calibration. Each iteration visits two optimized time steps with one transformation per time step according to a fixed cyclic schedule. Their losses are averaged before one Adam step. Learning rates are 3.75×10^{-3}, 7.5×10^{-4}, and 3×10^{-3} for SH DC, higher-order SH, and opacity. The 1,360 steps give 80 visits per center. After every update, a common-scale feasibility projection checks all 34 optimization frames at $\epsilon = 0.35/255$; it does not reset Adam's moments. The per-center objective is $4\mathcal{L}_{\mathrm{tar}} + 0.5\mathcal{L}_{\mathrm{feat}} + 8\mathcal{L}_1 + 0.05\mathcal{L}_\infty + \mathcal{L}_{\mathrm{reg}}$, with fidelity terms evaluated before processing. Writing $\mathrm{ms}(z) = \sum_j z_j^2/|z|$, the regularizer is

$$\mathcal{L}_{\mathrm{reg}}(t) = 3\times10^{-4}\big[\,\mathrm{ms}(\Delta\mathbf{c}_{\mathrm{DC}}(t)) + \mathrm{ms}(\Delta\mathbf{c}_{\mathrm{rest}}(t)) + \mathrm{ms}(\Delta\mathbf{o}(t))\big].$$

Each array contains only selected Gaussians' changes from the clean anchor, after interpolation at the visited time; $\mathbf{o}$ denotes raw opacity logits. The DC and higher-order means are separate, not one mean over all SH coefficients.

Final calibration checks all 456 release frames. Both projection stages first test scale 1, then use 14 bisection steps if needed and re-verify the accepted state, with backoff or a clean fallback. The numerical acceptance tolerance is 10^{-6} in 8-bit MAE units ($10^{-6}/255$ in normalized units). The final $\hat{\alpha}$ is between 0.96 and 0.98 for the three historical variants. This is a verified feasible scale on a finite release, not a certified maximum over all scales or an all-view/time guarantee. Table 7 summarizes this protocol.

Procedure overview. Figure 8 summarizes one protection update and the separate recipient evaluation, and Figure 9 gives implementation-aligned pseudocode of the protection construction shared by all protocols. Algorithm 1 in Section 3.4 states the procedure in the notation of the main paper.

Table 7: Hyperparameters of the original Chickchicken asset experiment.

Parameter	Value
Per-frame budget ϵ / pixel threshold κ	$0.35/255$ / $8/255$
Loss weights λ_{fid} / λ_{bg} / λ_∞ / λ_{f}	8 / 4 / 0.05 / 0.5
Prompt loss weight λ_{p}	4
SH and opacity regularization (each) / selection threshold ρ	3×10^{-4} / 0.7
Learning rate (SH DC / SH rest / opacity)	3.75×10^{-3} / 7.5×10^{-4} / 3×10^{-3}
Iterations $\times$ time steps per iteration	1,360 $\times$ 2 (80 visits per frame)
Number of temporal controls K	4 (default), 1
Transformation set $\mathcal{T}$ (`4DGuard`-ST)	{id, JPEG50, Avg2, Avg2$\rightarrow$JPEG50}
Feasibility-search steps (training / final)	14 / 14
Prompt box margin	8 px

B Additional Quantitative Results

The results below retain the original Chickchicken protocols and their fixed checkpoints. Frame counts are correlated observations within one sequence, not independent scene repetitions.

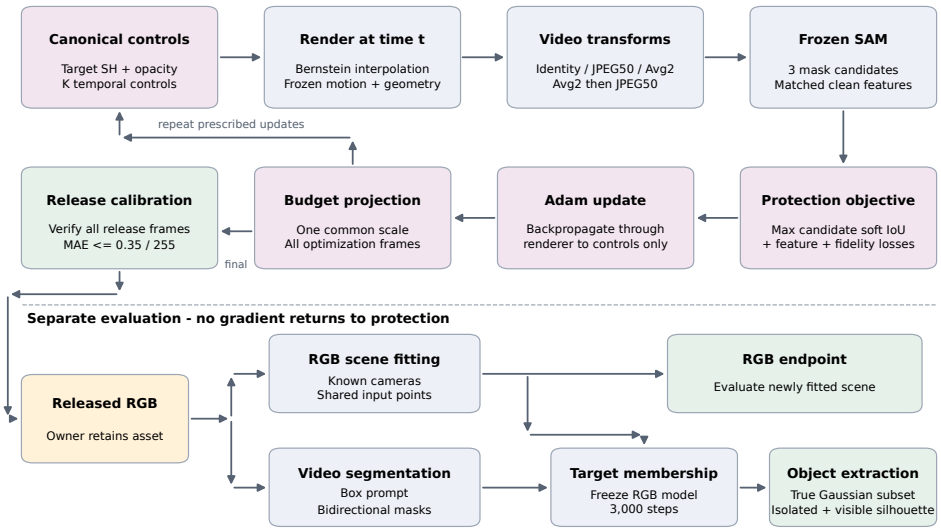


Figure 8: Optimization loop of `4DGuard` (top) and the separate recipient evaluation (bottom). Only the target appearance and opacity controls are optimized; geometry, motion, and SAM remain frozen. After every Adam update, one common scale projects the controls onto the budget; the final calibration verifies all released frames. The recipient's RGB fitting and video segmentation branches receive only the released RGB, and no gradient flows from the evaluation back to protection.

Input. Clean dynamic Gaussian asset G^0, optimization frames $\mathcal{A}$ with target masks and boxes, finite release views/times $\mathcal{F}$, frozen SAM, prescribed update schedule, $K \in \{1, 4\}$, and $\epsilon = 0.35/255$.

Project moving Gaussian centers into the annotated optimization frames. Select target support using in-mask ratio ≥ 0.7 and at least two valid projections.
Initialize target SH and raw-opacity controls $\theta_{i,k} = 0$. Keep canonical geometry, motion weights, all other asset parameters, and SAM fixed.
Cache quantized clean renders and matching transformed clean encoder features.
For each of the S prescribed Adam updates:
  Choose two centers and one transformation per center by the fixed cyclic schedule.
  For each required source time t, set $a_i(t) = a_i^0 + \sum_{k=0}^{K-1} B_k(u(t))\theta_{i,k}$ on the target support; render and quantize RGB through the frozen motion model.
  Apply the scheduled identity, JPEG50, Avg2, or Avg2 then JPEG50 transform. Avg2 uses actual adjacent release frames at their own cameras and times.
  Obtain SAM's three mask logits and encoder features. Compute the center objective L_t below using the matched clean input.
  Average the two center losses, backpropagate, and take one Adam update of θ only. Use straight-through derivatives for quantization, flooring, and JPEG.
  Set $\theta \leftarrow \alpha\theta$, where one common $\alpha = \mathrm{FeasibleScale}(\theta, \mathcal{A})$ is verified on every optimization frame. Retain Adam's moments.
End for. Use the endpoint prescribed by the experiment protocol.
Calibrate all controls with one $\widehat{\alpha} = \mathrm{FeasibleScale}(\theta, \mathcal{F})$ and set $\theta \leftarrow \widehat{\alpha}\theta$.
Independently re-render every release frame and verify the exact exported controls satisfy the image budget.
Output. Store the calibrated asset internally; publish its quantized RGB frames.

Center objective.

$$L_t = 4 \max_{j=1,2,3} \mathrm{sIoU}(\sigma(z_t^j), \overline{M}_t) + 0.5L_{\mathrm{feat}} + 8L_1 + 0.05L_{\mathrm{hinge}} + L_{\mathrm{reg}}.$$

L_{feat} is target-weighted cosine similarity of clean/protected encoder features after the same transform. For the *unprocessed* quantized center, $d = |x^\theta - x^0|$, $L_1 = \mathrm{mean}[(1 + 3(1 - M_t))d]$, and $L_{\mathrm{hinge}} = \mathrm{mean}([255d - 8]_+^2)$. L_{reg} is 3×10^{-4} times the sum of separate mean squares of interpolated SH DC, higher-order SH, and raw-opacity changes.

Temporal field. $u(t) = (t - t_{\min})/(t_{\max} - t_{\min})$ uses actual camera timestamps; $B_k(u) = \binom{K-1}{k}(1-u)^{K-1-k}u^k$. $K = 4$ is one cubic segment; $K = 1$ is constant.

FeasibleScale$(\theta, \mathcal{U})$**.** Verify scale 1 first. If needed, verify scale 0, then perform 14 bisection-style checks. Re-render the accepted state; halve its scale if verification fails, with a checked clean fallback. Each check requires per-frame full-image MAE $\leq \epsilon + 10^{-6}/255$ on all $\mathcal{U}$. This returns a verified feasible scale, without a largest-scale or all-view guarantee.

Separate recipient evaluation (no gradient to protection). Released RGB supplies two branches: fresh RGB scene fitting, and SAM 2.1 video masks from a box on the first annotated frame with propagation in both directions. Evaluate the fitted RGB model at the fixed test views. For object extraction, freeze that fitted model, learn Gaussian membership from the video masks for 3,000 Adam steps, select membership > 0.5, and evaluate the actual Gaussian subset. Known cameras and a shared initialization point cloud are provided in these controlled tests.

Figure 9: Implementation-aligned pseudocode of the `4DGuard` protection construction with a verified release budget, followed by the separate recipient evaluation. S counts actual Adam updates.

Ablation tables. Table 8 uses 46 centers and 3,680 single-center steps (80 visits each). Its Gaussian arms use the historical per-frame joint14 parameterization, largest-50%/clone2 support, and learning rate 0.05; this differs from the canonical full-SH/opacity arms of Table 1. Single-head training uses `multimask_output=False` (head 0); max-of-three training uses `True` (heads 1–3). Both are evaluated by choosing the candidate with the highest ground-truth IoU among heads 1–3. The single-head row in (a) repeats the Gaussian JPEG-EOT arm in (b). The three executions use identical random-number settings, zero initialization, and the same cyclic schedule. Their population standard

Table 8: Historical Chickchicken diagnostics on all 46 annotated frames with per-frame controls (ΔIoU in pp, $\uparrow$). (a) Single-mask versus multimask objective (mean $\pm$ population standard deviation across three executions under identical settings). (b) Perturbation space and transformation set; JPEG75 is absent from optimization. All arms satisfy the same per-frame budget before processing.

(a) Parser-aligned objective (Gaussian, JPEG-EOT).

Objective	None	JPEG50	JPEG75
Single-head	5.00$\pm$0.10	2.51$\pm$0.05	2.54$\pm$0.14
Max-of-3 (ours)	**18.58**$\pm$0.25	**11.78**$\pm$0.06	**12.09**$\pm$0.06

(b) Perturbation space and EOT (single-head objective).

Space	EOT	None	JPEG50	JPEG75
Pixel	–	74.04	0.20	0.56
Pixel	JPEG	47.40	1.25	0.99
Gaussian	–	11.33	0.36	0.54
Gaussian	JPEG	4.80	**2.64**	**2.43**

deviations describe execution repeatability and use denominator three; variation under other random-number settings was not evaluated.

Distribution of the effect over frames. Table 2 complements the frame-averaged results of Table 1 with the maximum reduction and the number of frames whose IoU drops by more than 10 pp. The protection is concentrated on the frames where the parser is most vulnerable: on optimized frames, the default `4DGuard` lowers the IoU by more than 10 pp on 10 of 34 frames directly and on 6 frames after JPEG50 or JPEG75, with maxima above 23 pp, and on unoptimized time steps it reaches 21.5 pp directly and 13.1 pp after JPEG50.

Absolute parsing quality. For reference, the absolute IoU* of the default `4DGuard` on optimized frames is 0.884 (None), 0.925 (JPEG50), and 0.921 (JPEG75), against 0.956, 0.959, and 0.959 for the unprotected renders; on unoptimized time steps it is 0.916, 0.942, and 0.936 against 0.955, 0.956, and 0.956. The per-frame Gaussian baseline reaches 0.765 (None) and 0.834 (JPEG50) on optimized frames only.

Optimization dynamics. The curves in Figure 5 use identical support, full-SH/opacity channels, and max-of-three JPEG-EOT objectives. There are 2,720 single-center visits: the shared optimizer updates every visit, while each independent per-frame optimizer updates 80 times. After 20/40/80 visits per frame, JPEG50 effects are 4.89/9.77/12.54 pp for per-frame controls and 3.78/4.47/4.48 pp for shared controls. The latter endpoint exceeds the budget on 13 of 456 frames (maximum $0.36570/255$); it is a training-frame diagnostic, not the calibrated K1-ST release in Table 1. Its finite-budget plateau does not establish convergence.

C Additional Qualitative Results

Figure 11 compares the historical Chickchicken renders with the three `4DGuard` variants under JPEG50 on an optimized frame at the beginning of the sequence (frame 79), the unoptimized frame 239, and an optimized frame near the end (frame 399). Figure 6 shows the default variant on the same frames under direct evaluation, JPEG50, Avg2, and Avg2$\rightarrow$JPEG50. Of these examples, the largest parsing change occurs when the container is open and its interior is visible (frame 239), where the best candidate excludes part of the interior. The reductions are smaller for the closed container in frames 79 and 399. These observations do not isolate the effect of object configuration from prompt or rendering differences. Temporal averaging restores most of the parsing quality on these examples, consistent with Table 1.

D Discussions

Difference from static 3DGS protection. `AdLift` (Hong et al., 2025) studies protection of a static Gaussian asset against editing by lifting image perturbations into a set of safeguard Gaussians. Our release is a sequence rendered from a dynamic asset. We attach appearance controls to canonical Gaussians, evaluate a temporal control field through their existing motion, and include temporal averaging in the parser objective. Final calibration checks one common scale over the finite release sequence. These choices define and constrain the perturbation at the release times; whether its parsing effect transfers to unoptimized times or to reconstruction is assessed by the corresponding experiments.

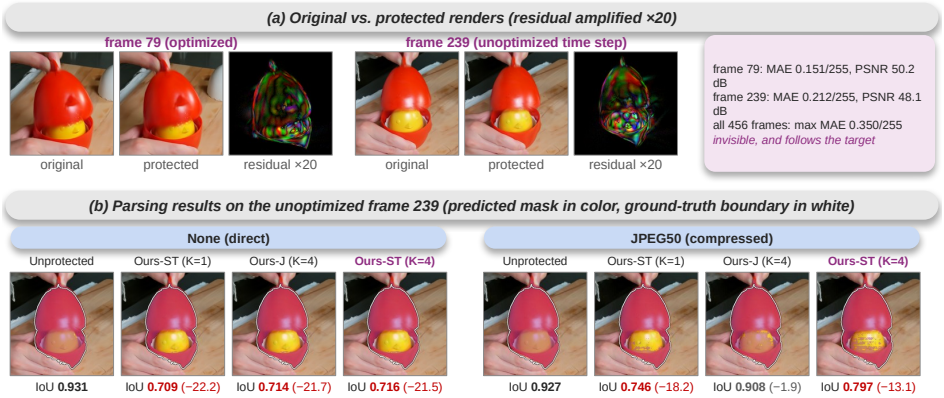


Figure 10: Additional Chickchicken examples. (a) Original and protected renders at an optimized frame and at an unoptimized time step, with the residual amplified $20\times$. (b) Parsing on the unoptimized frame 239 for unprotected renders and three `4DGuard` variants under direct evaluation and after JPEG50 (predicted mask in color, ground-truth boundary in white, best-candidate IoU below each panel).

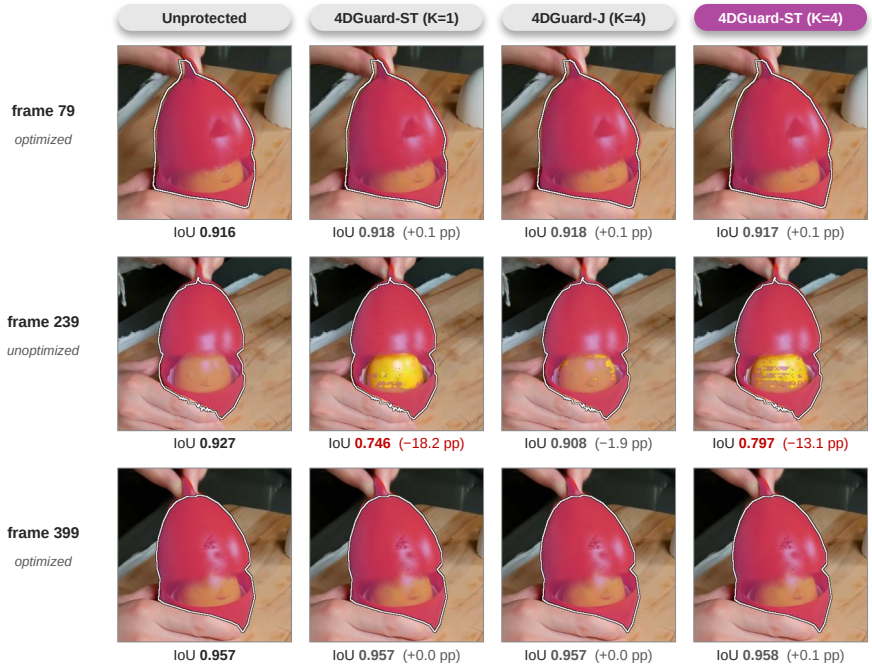


Figure 11: Parsing results under JPEG50 for unprotected renders and three `4DGuard` variants on frames 79 (optimized), 239 (unoptimized), and 399 (optimized). The predicted mask of the best candidate is overlaid in color and the ground-truth boundary in white; the best-candidate IoU is printed below each panel.

Effect of temporal averaging. Unwarped averaging is a temporal box filter at fixed image coordinates. For intuition, consider an idealized perturbation on an object translating by u pixels per frame, with sinusoidal profile $\delta_t(x) = a\cos(2\pi k(x - ut))$ and spatial frequency k. Averaging two consecutive frames scales its amplitude by $|\cos(\pi ku)|$, which vanishes at $k = 1/4$ cycles per pixel for $u = 2$ pixels per frame. This example illustrates how motion can attenuate an object-attached signal even when its material coordinates are consistent. It is not a spectral diagnosis of the learned perturbations or a causal explanation of the reported SAM scores. Spatio-temporal EOT optimizes the processed parsing objective, and the historical Avg3 results remain weak. Adding longer windows would require a new optimization and evaluation, with no established gain from that change alone.

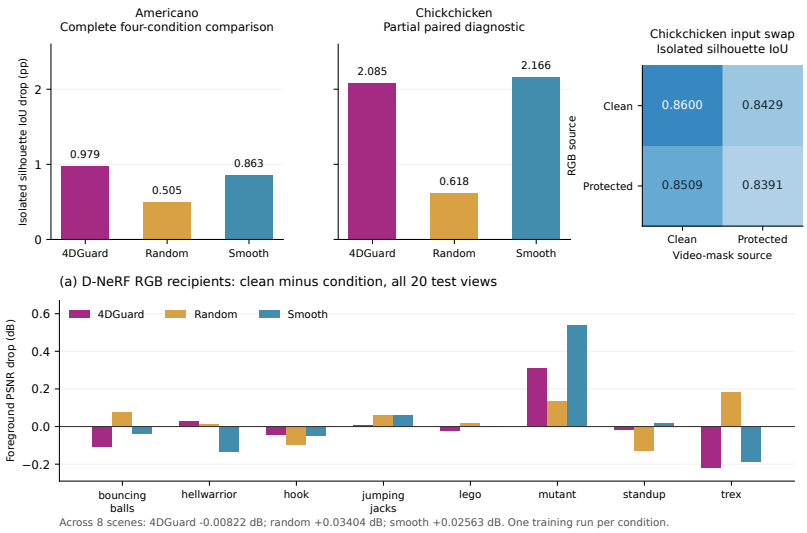


Figure 12: Recipient-side analysis. Top: extraction IoU drop relative to the clean release and the Chickchicken input swap (IoU for each RGB/mask source). Bottom: per-scene foreground PSNR drop of RGB-only recipients on D-NeRF.

Choice of the budget. We adopt a full-frame MAE budget of $0.35/255$. PSNR exceeds 48 dB on the sampled historical renders, but this observation is not a perceptual guarantee or a consequence of an MAE bound alone. MAE constrains average pixel change, whereas the ℓ_∞ budgets of $8/255$ to $16/255$ used in image-protection settings (Madry et al., 2018; Salman et al., 2023; Hong et al., 2025) constrain the largest change. Their numerical thresholds therefore cannot be directly ranked by strictness. An MAE bound also does not enforce equal distortion inside targets of different sizes. A different budget would require new calibration and reporting of both release distortion and task outcomes; allowing more distortion does not establish stronger protection.

Best-of-three evaluation. The best-candidate score uses the fixed reference mask and measures the upper envelope of the three returned candidates for the specified prompt. It avoids crediting an attack when another candidate succeeds. A practical recipient may use a different selection rule, prompt, segmenter, or direct RGB reconstruction, so this protocol does not bound the disruption of an arbitrary reconstruction pipeline.

E Limitations and Future Directions

Scenes and parsers. The historical tables above concern Chickchicken and SAM ViT-H. Later completed experiments include five-model parsing evaluations on Americano and Split-cookie, and fresh RGB reconstruction on all eight D-NeRF scenes (Section F). These results broaden the evidence beyond the original single-scene study, but do not establish effectiveness for arbitrary scenes, prompts, or reconstruction architectures. The five-model evaluation includes the generating surrogate and should not be described as five unseen-model tests.

Stronger purification. Three-frame averaging removes much of the parsing effect in the historical Chickchicken evaluation. Those tables do not evaluate motion-compensated denoising, nor do parser scores after processing establish the outcome of retraining a recipient on processed video. Longer averaging windows and motion-compensated filters are candidate evaluation and optimization settings whose effects remain to be tested under the relevant endpoint.

Optimization budget. The original Chickchicken controls receive one update per frame visit under the 1,360-iteration schedule. Later runs use consecutive inner updates under separate protocols; they do not replace the historical assets or checkpoints reported above. A plateau in a training-surrogate score is neither a convergence proof nor an efficacy criterion, and additional inner updates need not improve fresh reconstruction outcomes.

End-to-end reconstruction evaluation. Fresh reconstruction and object-extraction evaluations must be distinguished from parser tests (Sections F and G). RGB-only recipients optimize reconstruction

without SAM; the complete eight-scene D-NeRF comparison shows no stable advantage over ordinary perturbations. The completed Americano mask-guided comparison also falls below its degradation criterion, while Chickchicken's main comparison and Split-cookie extraction remain incomplete. Their references and target definitions differ from official D-NeRF foreground evaluation and cannot be pooled as one protocol. Directly optimizing the recipient's learning process remains a proposed method change, not an established property of the current SAM objective.

Multi-view extension. We have begun extending the evaluation to the multi-view Neu3D benchmark. Figure 13 shows render probes of the clean *sear_steak* asset at three saved checkpoints; the protected release and its recipient evaluation have not been run yet, so this dataset enters no conclusion of this paper.



Figure 13: Neu3D *sear_steak*: clean-asset render probes at three saved checkpoints for the same three training view–time pairs. Reload agreement verifies serialization, not novel-view quality; no protected result is shown.

Time coverage and visual quality. The temporal field defines controls throughout its time interval, while the reported parsing scores cover only annotated frames. The historical unoptimized block also has a development history, and the full release enters final quality calibration. Neither its scores nor the finite release budget establish protection or perceptual invisibility at arbitrary new times and viewpoints.

F Additional Transfer and Fresh-Reconstruction Evidence

The following experiments were completed after the original Chickchicken diagnostic. They use separately specified protection schedules and references; their results are not pooled with the earlier tables.

F.1 Additional Scenes and Unseen Segmenters

The additional HyperNeRF scenes are Americano (526 published frames) and Split-cookie (539 frames). Each uses 38 annotated optimization centers, a four-control temporal representation, and ten consecutive updates per center-pair visit ($K_{\text{in}} = 10$). The training-selected endpoints are outer pass 16 for Americano (3,040 Adam updates, training-objective stability criterion met) and outer pass 80 for Split-cookie (15,200 updates, budget exhausted). Stability of the training objective is not a claim of global convergence. The unoptimized-time sets contain 11 and 12 annotated frames, respectively. The final release assets satisfy the per-frame full-image MAE budget of $0.35/255$; their mean MAE is $0.204/255$ and $0.245/255$.

Table 9 evaluates the same protected release against SAM ViT-H and four segmenters not used in protection optimization. All clean/protected pairs are scored within one fixed evaluation environment. The score selects the best candidate by ground-truth IoU; SAM 2.1 is used as an image predictor,

Table 9: Additional HyperNeRF segmentation results. Entries are clean-minus-protected differences in the highest candidate IoU against annotations, in percentage points. "Unopt." denotes time steps excluded from protection optimization, not from recipient RGB training.

		Optimized frames		Unopt. times	
Scene	Segmenter	None	JPEG50	None	JPEG50
Americano	SAM ViT-H (surrogate)	0.883	0.949	0.121	0.281
	SAM ViT-L	0.616	0.720	0.169	0.237
	SAM ViT-B	0.850	0.774	0.114	0.321
	SAM 2.1 Hiera-L	0.472	0.576	0.154	0.276
	MobileSAM	2.071	1.056	0.048	1.305
Split-cookie	SAM ViT-H (surrogate)	60.780	26.602	1.970	5.242
	SAM ViT-L	40.245	17.970	7.582	8.431
	SAM ViT-B	29.663	17.144	9.685	11.006
	SAM 2.1 Hiera-L	16.375	3.143	0.674	1.030
	MobileSAM	21.924	18.564	12.305	11.259

Table 10: Fresh D-NeRF reconstruction, all scenes, fixed 20k fine checkpoint. Foreground PSNR (dB, higher is better) against official GT. Negative clean-minus-protected differences are retained.

Scene	Clean	Protected	Random	Smooth	Clean−prot.
bouncingballs	33.93644	34.04744	33.85730	33.97312	-0.11100
hellwarrior	19.75504	19.72700	19.73942	19.88884	0.02804
hook	22.93612	22.98169	23.03364	22.98844	-0.04557
jumpingjacks	23.62980	23.62062	23.56636	23.56849	0.00918
lego	19.37046	19.39207	19.35120	19.36935	-0.02161
mutant	27.67484	27.36183	27.53955	27.13789	0.31301
standup	26.05146	26.06906	26.17975	26.03304	-0.01760
trex	23.16350	23.38368	22.97805	23.35345	-0.22018
Equal-scene mean	24.56471	24.57292	24.53066	24.53908	-0.00822

without temporal memory. Each scene uses one protection optimization run. These are segmentation-transfer results, not unseen-reconstruction-model results. They also show why optimization-frame effects should not be substituted for temporal generalization.

F.2 Fresh RGB-Only Reconstruction as a Bypass

A recipient may bypass mask supervision by fitting RGB directly. These additional tests fit fresh models and evaluate their rendered outputs; the recipients do not use SAM or target masks in their reconstruction loss. They receive known cameras and the same initialization point cloud within a scene, including its colors, but not the owner's asset weights or protection controls. Thus, these are controlled fitting experiments rather than a pipeline that estimates every auxiliary input from video.

D-NeRF protocol and fixed endpoint. All eight D-NeRF scenes (Pumarola et al., 2021) use four release conditions: clean, protected, random K4, and smooth-color K4. The protected condition uses the fixed SAM objective, $K = 4$, $K_{\text{in}} = 10$, and the publication budget above. Each condition is trained once per scene for 3,000 coarse and 20,000 fine iterations under fixed initialization and sampling settings, yielding 32 fresh recipients and 24 clean/condition pairs. All 20 official test views per scene are evaluated; test RGB and alpha do not enter protection optimization or recipient training. The target is the complete official foreground ($\alpha > 0$), which can include a base, floor, or multiple objects rather than one manually annotated instance.

Table 10 reports foreground PSNR relative to official GT at the fixed 20k fine endpoint. Views are averaged within a scene, then the eight scenes receive equal weight. These 160 test views are not independent training repetitions. The protected condition has no positive mean PSNR degradation relative to clean and no mean advantage over either ordinary perturbation control. The foreground SSIM values are 0.849289 (clean), 0.847774 (protected), 0.846638 (random), and 0.849442 (smooth); the two metrics need not move in the same direction.

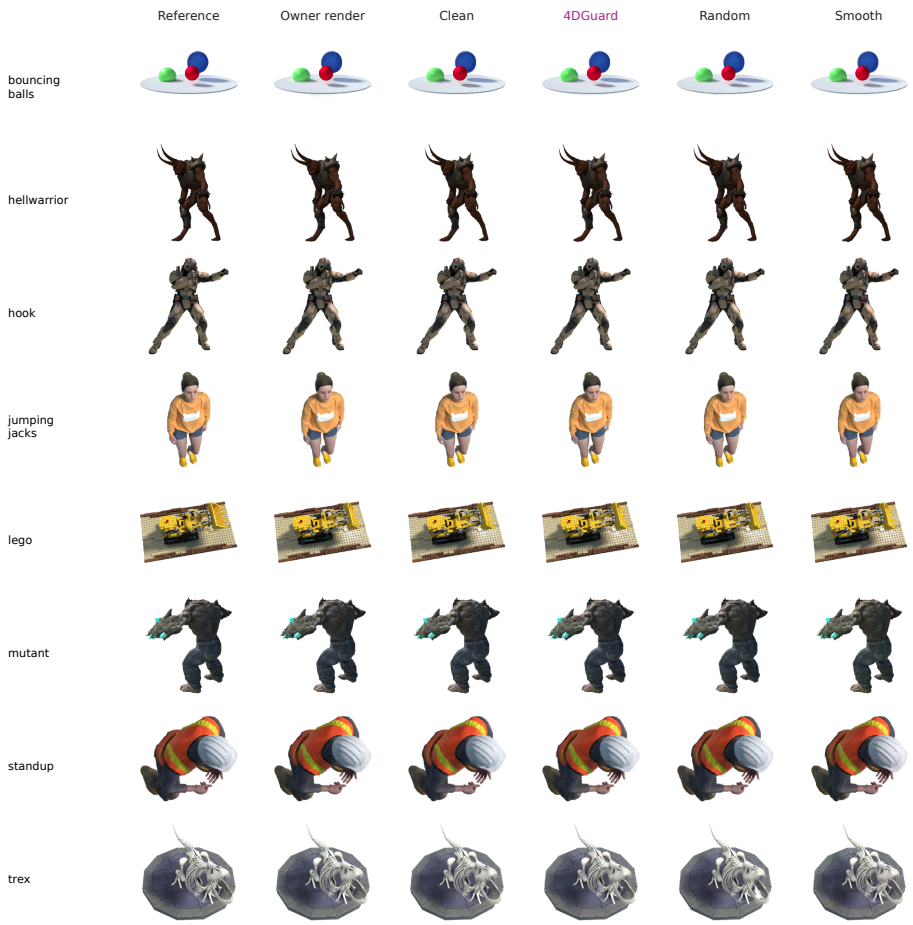


Figure 14: Fresh RGB-only recipients on all eight D-NeRF scenes at the fixed test view 10 and the 20k endpoint. Columns: official reference, owner render of the original asset, and recipients fitted to the clean, `4DGuard`, random, and smooth releases (one training run per condition). The view index was fixed for all scenes before inspection, and each row shares one reference-derived crop. The renders share the same reconstruction errors and look alike across conditions; Table 10 averages all 20 test views.

At 5k, 10k, and 20k, mean foreground PSNR drops are $+0.00297$, -0.01355, and -0.00822 dB, respectively. At 20k, the protected drop minus the random and smooth drops is -0.04226 and -0.03385 dB. The controls approximately match publication distortion: target RMS matching residuals range from -2.55% to $+3.81\%$ for random and -25.99% to $+7.93\%$ for smooth. The comparison does not establish strict equal-distortion equivalence or statistical significance. All eight protected releases pass their full-training-sequence image-budget checks. This fidelity result and the reconstruction result are separate outcomes.

Connecting segmentation to the same release. For seven D-NeRF scenes, archived image hashes connect the final calibrated release's segmentation evaluation to the exact RGB inputs used by the recipients above. The segmentation measurement uses the 38 optimization centers, SAM ViT-H with annotation-derived boxes, and the highest hard-mask IoU among three candidates. It is neither a held-out-time result nor a full-video segmentation average. Table 11 retains the weak front-end effects and the missing Hellwarrior front-end record; that scene's reconstruction evaluation is complete. These checks establish correspondence of saved artifacts, not a new segmentation inference run.

Mutant provides a same-asset example of substantial training-frame segmentation disruption without reconstruction degradation beyond the smooth control. Five of the seven available segmentation drops are below 1 pp. This distinction prevents attributing the eight-scene reconstruction result to a single unverified mechanism such as temporal correction of uniformly strong mask errors.

Table 11: Same-release D-NeRF comparison. Segmentation drops are measured after final calibration on optimization centers; reconstruction drops are at the fixed 20k endpoint against official foreground GT. Each drop is clean minus the indicated condition. Missing segmentation evidence is retained.

	SAM IoU drop	Foreground PSNR drop (dB)		
Scene	(pp)	Protected	Random	Smooth
bouncingballs	0.0078	-0.11100	0.07914	-0.03668
hellwarrior	missing	0.02804	0.01562	-0.13380
hook	0.7859	-0.04557	-0.09752	-0.05231
jumpingjacks	0.2809	0.00918	0.06344	0.06131
lego	0.2004	-0.02161	0.01926	0.00111
mutant	11.6659	0.31301	0.13529	0.53695
standup	0.6191	-0.01760	-0.12829	0.01842
trex	2.1255	-0.22018	0.18546	-0.18995

HyperNeRF endpoint observation. For Split-cookie's selected outer80 asset, a fresh R0 recipient at fine20k has target PSNR 27.89798 dB, compared with 32.55561 dB for the clean-trained recipient, a drop of 4.65763 dB. Background clean-minus-protected PSNR is -0.05567 dB. This uses 16 fixed teacher-rendered views and a teacher-SAM target ROI, not independent photographic GT. Existing random/smooth controls were distortion-matched to an earlier protection asset, so they cannot establish a strictly matched causal gain for this new endpoint. Numerical results are archived; review of the complete endpoint RGB set remains unfinished. These observations motivate separating publication appearance changes, receiver fitting error, and adversarial effects rather than treating any PSNR drop as proof that the target cannot be reconstructed.

G Mask-Guided Recipient and Object-Extraction Results

This experiment evaluates the recovered target submodel in a segmentation-dependent pipeline after fresh RGB reconstruction. It uses a separately specified recipient protocol and, for Chickchicken, a newly optimized publication asset; its results are not pooled with the historical segmentation diagnostics.

G.1 Recipient, References, and Evaluation

The recipient receives released RGB, fixed cameras, and a shared initialization point cloud including colors. RGB fitting does not use target masks. The main mask source is SAM 2.1 Hiera-L in video mode, prompted by a box on the first annotated frame and propagated in both temporal directions to cover the publication sequence. After RGB fitting, geometry, appearance, and deformation are frozen, and each Gaussian receives a trainable membership logit ℓ_i, initialized to zero, with $q_i = \sigma(\ell_i)$. The rendered target contribution is $O(v,t) = \sum_i T_i \alpha_i q_i$, where T_i and α_i are the scene's transmittance and opacity contributions. Binary cross-entropy against the predicted mask is combined with a squared membership-difference penalty over 16 nearest neighbors, weighted by 0.1. The frozen configuration uses Adam with learning rate 0.01 for 3,000 steps and two frames per step. Gaussians with $q_i > 0.5$ form the extracted submodel. We evaluate its accumulated alpha after removing the other Gaussians (E-iso), and separately the selected Gaussians' contribution while retaining the full scene's transmittance (E-vis). These two renderings need not have identical silhouettes.

The four publication conditions are clean, `4DGuard`, ordinary random perturbations, and spatially smooth perturbations. The ordinary controls share the target support and temporal-control family, with publication distortion approximately matched using target MAE, target RMS, and temporal-difference RMS; this does not establish equal perceptual distortion. All results below use unprocessed releases and the video-mask source. Americano has one RGB reconstruction per condition and three membership fits per reconstruction. Chickchicken planned three RGB reconstructions per condition, each with three membership fits. Metrics are averaged over membership fits, then RGB reconstructions, then the fixed evaluation views; views are not counted as independent training repetitions.

The primary views correspond to times excluded from protection optimization, but their RGB frames are available to the recipient. The eligible unoccluded subset contains two views for Chickchicken and

Table 12: Mask-guided extraction at unoptimized times. The first column identifies the publication conditions supplying RGB and video masks. Δ_{iso} is clean-minus-condition E-iso IoU in percentage points. Americano includes its complete planned main comparison. Chickchicken uses only the two RGB reconstructions completed in all four conditions, with three membership fits each; those rows and the crossed inputs are partial paired diagnostics. The two scene blocks use different references and are not pooled.

RGB / masks	E-iso IoU	E-vis IoU	Δ_{iso} (pp)
Americano: complete main comparison, 11 views			
Clean / clean	0.940115	0.934255	—
`4DGuard` / `4DGuard`	0.930328	0.936305	0.9787
Random / random	0.935065	0.935542	0.5050
Smooth / smooth	0.931481	0.935466	0.8634
Chickchicken: partial paired diagnostic, 10 views			
Clean / clean	0.859964	0.882574	—
`4DGuard` / `4DGuard`	0.839110	0.839910	2.0854
Random / random	0.853782	0.894976	0.6181
Smooth / smooth	0.838300	0.867469	2.1664
Chickchicken: crossed inputs, same paired reconstructions			
Clean / `4DGuard`	0.842902	0.816477	1.7061
`4DGuard` / clean	0.850879	0.891847	0.9084

none for Americano. Under the fixed fewer-than-three-view fallback rule, E-iso therefore uses all 10 and 11 unoptimized-time views, respectively, with a human-reviewed owner submodel as its reference. E-vis at these views uses the corresponding visible annotations. The contour thresholds, calibrated separately from membership selection, are 0.95 for Chickchicken and 0.55 for Americano. The owner submodel is a reviewed reference, not independent complete three-dimensional ground truth; Americano's repaired reference construction also used owner annotations from evaluation-source times before human review.

A rendering check originally required equality between full-scene target contribution and isolated-submodel alpha on reviewed unoccluded pixels. It failed on Chickchicken. Before the reported membership fits, the protocol was amended to check compositing, alpha-rendering, and full-subset consistency separately, retaining the original discrepancy as a diagnostic. The results here use that disclosed amendment; they are not an unmodified pass of the original protocol.

G.2 Completed Americano Main Comparison

All four Americano aggregates in Table 12 bind three completed membership-fit evaluations each. The E-iso reduction is 0.9787 pp for `4DGuard`, compared with 0.5050 pp for random and 0.8634 pp for smooth perturbations. The registered relative-effect rule requires the protected E-iso drop to exceed the largest nonnegative ordinary-control drop by at least 5 pp, reach at least twice that drop, and coincide with an E-vis decrease. These are per-scene components of the registered rule, which requires qualifying results on at least two scenes; a completed comparison in one scene cannot establish the full claim. Americano meets none of these three conditions: the additional E-iso drop over smooth is only 0.1152 pp, while E-vis IoU increases by 0.2050 pp. After averaging the membership fits, none of the 11 primary views has protected E-iso IoU or recall at or below 0.5; the minimum IoU is 0.901445. On the 16 fixed novel views, E-iso IoU is 0.937216, 0.931465, 0.943093, and 0.929318 for clean, `4DGuard`, random, and smooth, respectively. The protected drop is 0.5751 pp, below smooth's 0.7898 pp. Thus this complete scene comparison does not meet the specified extraction-damage criterion.

G.3 Chickchicken: Missing Main Result and Paired Diagnostics

For the planned three-reconstruction comparison, complete E-iso aggregates are 0.860308 for clean, 0.854236 for random, and 0.844206 for smooth. The protected aggregate is unavailable because one planned protected RGB reconstruction failed; that failure remains part of the result record. The Chickchicken rows in Table 12 instead compare the two RGB reconstructions available in every condition. This analysis was formed after the missing result was known, and the missingness may

depend on the experimental condition; it cannot replace the planned main comparison or establish a complete negative result. Within this paired diagnostic, the protected E-iso drop of 2.0854 pp does not exceed smooth's 2.1664 pp, and none of the 10 primary views has protected IoU or recall at or below 0.5 after averaging. Holding protected RGB fixed and replacing its masks with clean-video masks increases E-iso IoU from 0.839110 to 0.850879, a recovery of 1.1769 pp; the result remains 0.9084 pp below clean RGB with clean masks. Conversely, protected-video masks applied to clean RGB give IoU 0.842902, a 1.7061 pp decrease. These crossed inputs separate changing the RGB reconstruction input from changing the mask-prediction input within the same paired runs. They do not establish a unique error-correction mechanism or show that mask disruption reliably prevents extraction.

Split-cookie has no admissible main extraction comparison in this evidence set: the repaired reference lacked human acceptance, no eligible unoccluded calibration frame was available, and an original smooth-control RGB training failure is retained. Consequently, the full planned three-scene decision remains incomplete even though Americano's main comparison is complete.