RemedyGS: Defend 3D Gaussian Splatting against Computation Cost Attacks

Supplementary Material

7. Additional Experimental Results

7.1. Defend Against Black-box Attack

In Section 5, we validate that our defense framework achieves a favorable trade-off between safety and utility against white-box attacks. In this subsection, we further evaluate its effectiveness against black-box attacks. Similarly, we use the official implementation of Poison-splat [3] to simulate attacker behavior in black-box settings and the official implementations of Scaffold-GS [4] to simulate victim behavior. As reported in Tables 6 and 7, our defense framework consistently outperforms naive defenses, yielding improvements in utility by more than 1 dB while maintaining comparable safety. The results indicate that our proposed framework demonstrates significant effectiveness in different settings and provides comprehensive safeguarding for real-world 3DGS systems.

7.2. Defend Against Attacks of Different Strengths

We evaluate our defense framework under attacks of different strengths in the white-box setting. For the detector, we consider $\epsilon = 50/255$, $\epsilon = 100/255$, and $\epsilon = 150/255$, and report accuracy, F1 score, and recall metrics on mixed poisoned and clean datasets. As shown in Table 8, the detector consistently achieves a strong discrimination capability, demonstrating notable robustness against different settings. We then assess the purifier with $\epsilon = 26/255$, $\epsilon = 35/255$, and $\epsilon = 40/255$, as detailed in Table 9 and Table 10. Baselines such as image smoothing and limiting the number of Gaussians demonstrate weaker robustness and incur rapidly increasing computational cost as attack strengths are intensified, while our method remains effective even under stronger adversaries. In terms of utility, our approach also delivers superior 3D reconstruction. With increasing ϵ, naive defenses degrade performance by up to 3 dB, whereas our method suffers far less degradation and consistently outperforms baselines. Overall, our framework achieves a more favorable trade-off between safety and utility across different attack strengths.

7.3. Defend Against Adaptive Attacks

In this subsection, we further evaluate the defensive capabilities of our approach against adaptive computation cost attacks. In general adaptive attacks [1, 2, 5, 6], the attackers are aware of the defense strategy and actively adapt their attack strategy to more effectively exploit the system vulnerabilities in response to the deployed defense. Therefore, we assess the robustness of our defense method when confronted with stronger adversaries that explicitly leverage the knowledge of the defense to enable more challenging attacks. Notably, prior work [3] has not explored adaptive attacks that target the computation cost in 3DGS systems. To the best of our knowledge, we are the first to study the adaptive attack variant tailored for this topic, along with a comprehensive evaluation of our defense framework against such adaptive computation cost attack.

First, we develop an adaptive attack formulation that incorporates the knowledge of our defense mechanism. Our defense relies on detecting distributional discrepancies between attacked and clean images, followed by a purifier that removes the negative influence introduced by the adversary and aligns the resulting distribution with the source distribution. Being aware of this defense strategy, the adversary can augment its attack optimization process with additional constraints that enforce the adversarial images to remain close to the clean distribution, thereby undermining the defense framework and inducing more severe vulnerabilities. Specifically, we formulate the adaptive attack objective by adding a new weighted term, the mutual information between the poisoned images and the clean images, to the objective of the computation cost attack, enabling the attacker to produce more subtle and harder-to-defend adversarial perturbations. Mathematically, it is expressed as:

$$\max_{\mathcal{V}_{\text{poi}}} \mathcal{L}_{\text{ada}} = \max_{\mathcal{V}_{\text{poi}}} \left\{ \mathcal{S}_{TV}(\mathcal{V}_{\text{poi}}) + \beta \cdot I(\boldsymbol{V}_{\text{cln}}; \boldsymbol{V}_{\text{poi}}) \right\}, \quad (15)$$

where $\mathcal{S}_{TV}$, $I(\boldsymbol{V}_{\text{cln}}; \boldsymbol{V}_{\text{poi}})$, and β denote the TV score, mutual information between the poisoned images and the clean images, and the weight of mutual information, respectively. This formulation allows the attacker to simulate stronger and more stealthy attacks.

For a comprehensive evaluation of defense performance, we evaluate the robustness of our defense method against this adaptive computation cost attack, which presents stronger vulnerabilities. Specifically, we set β as 1, 5, and 20 to perform adaptive computation cost attack and obtain the adaptively attacked images. Then, the attacked images are utilized to evaluate our defense strategy. Experimental results are reported in Table 11 and Table 12. Our method consistently demonstrates strong defense capabilities against adaptive attacks and recover the clean images from the attacked images.

7.4. Impact of Undetected Poisoned Images

In addition to investigating the impact of misclassified clean images on the system's utility performance in Table 4, we further investigate the effect of undetected poisoned images on system safety. Specifically, we assess how varying pro-

Table 6. Comparison of the *image smoothing* baseline with our RemedyGS framework against black-box attacks, evaluated in terms of computational cost metrics including the number of Gaussians, peak GPU memory usage, training time, and rendering speed on the Mip-NeRF360 dataset. Compared with the baseline, our RemedyGS demonstrates the strongest capability to defend against computation cost attacks.

Metric	Number of Gaussians			Peak GPU memory [MB]			Training time [minutes]			Rendering speed [FPS]		
Setting / Scene	clean	image smoothing	RemedyGS (Ours)	clean	image smoothing	RemedyGS (Ours)	clean	image smoothing	RemedyGS (Ours)	clean	image smoothing	RemedyGS (Ours)
MIP-bonsai	4.396 M	3.833 M	3.947 M	9454	8685	8944	25.70	28.15	24.47	172	94	184
MIP-counter	2.843 M	2.439 M	2.584 M	10586	8814	9106	30.42	31.63	30.68	41	43	44
MIP-kitchen	3.395 M	3.125 M	3.190 M	11320	10513	10116	31.56	34.27	29.92	42	42	38
MIP-room	2.924 M	2.229 M	2.454 M	11823	10097	10655	28.92	29.51	25.04	46	49	39

Table 7. Comparison of the *image smoothing* baseline with our RemedyGS framework against black-box attacks, evaluated in terms of utility metrics including PSNR, LPIPS, and SSIM on the Mip-NeRF360 dataset. Our RemedyGS demonstrates superior performance by effectively recovering clean images from poisoned inputs manipulated by black-box attacks.

Metric	PSNR ↑			LPIPS ↓			SSIM ↑		
Setting / Scene	clean (Ground truth)	image smoothing	RemedyGS (Ours)	clean (Ground truth)	image smoothing	RemedyGS (Ours)	clean (Ground truth)	image smoothing	RemedyGS (Ours)
MIP-bonsai	32.884	30.289	**31.554**	0.182	0.253	**0.228**	0.948	0.883	**0.925**
MIP-counter	29.479	28.318	**28.926**	0.184	0.266	**0.233**	0.917	0.853	**0.884**
MIP-kitchen	31.426	29.268	**30.607**	0.119	0.216	**0.158**	0.932	0.861	**0.899**
MIP-room	32.085	30.167	**31.311**	0.190	0.281	**0.243**	0.931	0.847	**0.903**

Table 8. Performance of the detector in discriminating clean data from poisoned data across varying attack strengths on the Mip-NeRF360 dataset. The detector in our RemedyGS accurately distinguishes poisoned data of different attack strengths.

Scene	Metric	Setting (ϵ)		
		50/255	100/255	150/255
MIP-bonsai	Accuracy	0.9919	0.9939	0.9959
	F1	0.9919	0.9939	0.9959
	Recall	0.9919	0.9939	0.9959
MIP-all	Accuracy	0.9905	0.9929	0.9962
	F1	0.9905	0.9929	0.9962
	Recall	0.9905	0.9929	0.9962

portions of poisoned images that remain undetected and unpurified influence the system's safety performance. We combine 1%, 5%, 10% poisoned images with the remaining purified poisoned images in a scene and test the safety performance of the 3DGS system. The results are detailed in Table 13. The computational cost remains nearly identical to that observed for clean images across different mixed ratios, indicating that the presence of up to 10% poisoned images undetected in a scene almost has negligible influence on system safety. Furthermore, our detector demonstrates discriminative capability, achieving nearly 100% accuracy in identifying poisoned images. Even in cases where a small fraction of poisoned images are misclassified and remain undetected, such errors have minimal to no effect on the overall safety of the system.

7.5. Defense Performance in terms of Training Time and Rendering Speed Across Multiple Datasets.

We provide additional evaluation results on the computational cost in terms of training time and rendering speed, as an extension of Table 1. As shown in Table 14, the results indicate that our method mitigates the increase in computational overhead, achieving the same level of computational cost comparable to that of clean images.

7.6. Additional Visualization Results

We provide additional qualitative results of our method in Figure 5. The results demonstrate that although limiting the number of Gaussians imposes an upper bound on the computational cost, sharpened regions force reallocation of Gaussians from unsharpened areas, leaving certain regions without Gaussian representation and consequently diminishing local details. In contrast, our method preserves nearly the same level of detail as the clean images.

Additional visualizations of the recovered images by different methods are shown in Figure 6. It confirms that adversarial training effectively alleviates local over-smoothing issues, thereby enhancing perceptual fidelity and enabling higher-quality 3D representations.

8. Implementation Details

Detector. We train the detector on a curated dataset that consists of paired benign and poisoned images. During the training process of the detector, the images are resized to 960×528. The detector is trained with a learning rate of

Table 9. Comparison of baselines: *image smoothing* and *limiting the number of Gaussians* with the purifier in our RemedyGS framework on poisoned data under varying and stronger attack strengths. The evaluation is conducted on the Mip-NeRF360 dataset using utility metrics. Our RemedyGS demonstrates superior capability in recovering the original clean data from the poisoned data, even under stronger attack scenarios.

Metric	PSNR ↑			LPIPS ↓			SSIM ↑		
Setting / Scene	image smoothing	limiting Gaussian number	RemedyGS (Ours)	image smoothing	limiting Gaussian number	RemedyGS (Ours)	image smoothing	limiting Gaussian number	RemedyGS (Ours)
Constrained Poison-splat attack with $\epsilon = 26/255$									
MIP-bonsai	28.398	22.680	**30.212**	0.287	0.476	**0.254**	0.832	0.656	**0.891**
MIP-counter	27.069	20.064	**27.952**	0.300	0.509	**0.264**	0.808	0.577	**0.853**
MIP-kitchen	28.641	20.402	**29.088**	0.231	0.446	**0.186**	0.844	0.593	**0.871**
MIP-room	27.963	21.221	**29.662**	0.331	0.486	**0.283**	0.785	0.563	**0.855**
Constrained Poison-splat attack with $\epsilon = 35/255$									
MIP-bonsai	26.750	21.706	**29.574**	0.340	0.498	**0.278**	0.778	0.597	**0.881**
MIP-counter	25.709	19.112	**27.296**	0.340	0.533	**0.283**	0.760	0.533	**0.832**
MIP-kitchen	27.544	19.637	**28.388**	0.263	0.477	**0.212**	0.812	0.532	**0.849**
MIP-room	26.005	20.129	**29.045**	0.375	0.506	**0.304**	0.719	0.507	**0.839**
Constrained Poison-splat attack with $\epsilon = 40/255$									
MIP-bonsai	25.739	20.830	**28.871**	0.367	0.509	**0.291**	0.742	0.569	**0.859**
MIP-counter	24.929	18.633	**26.764**	0.361	0.542	**0.294**	0.731	0.514	**0.813**
MIP-kitchen	26.910	19.105	**27.664**	0.282	0.490	**0.227**	0.792	0.505	**0.831**
MIP-room	25.078	19.751	**28.517**	0.402	0.517	**0.313**	0.676	0.480	**0.821**

1×10^{-4} for 50 epochs using a batch size of 16. During the evaluation, images from the Mip-NeRF360 dataset with a resolution of 1600×1066 are randomly cropped to the same resolution as the training data. Due to the locally injected poisoned textures, the detector can effectively identify poisoned images based on these randomly cropped samples.

Purifier. We also train the purifier on the curated dataset that consists of paired benign and poisoned images. During training, we resize all input images to a fixed resolution of 960×544 to guarantee that the input images can be faithfully reconstructed with a consistent spatial resolution after passing through the encoder–decoder pipeline. The purifier is trained with a learning rate of 1×10^{-4} for 50 epochs using a batch size of 16. During the evaluation, images from the Mip-NeRF360 dataset are resized to 1600×1056.

Adversarial Training. For adversarial training of the purifier, we first employ a warm-up model trained solely with the MSE loss in Eq. (10) to stabilize the optimization. We then fix the parameters of this warm-up purifier and train a discriminator to distinguish between the original clean images and the outputs produced by the warm-up purifier. During this stage, the discriminator is trained with a learning rate of 5×10^{-4} for 50 epochs. The discriminator and the warm-up purifier are subsequently used to initialize the adversarial training process, during which the discriminator and purifier are alternately updated. Specifically, the purifier is optimized with a weighted combination of three losses, as defined in Eq. (14), with weights set to $\alpha_1 = 0.23$, $\alpha_2 = 100$, and $\alpha_3 = 1$, respectively. The learning rates for both the purifier and the discriminator are set to 2.5×10^{-4} and the epoch is set to 40.

Table 10. Comparison of baselines: *image smoothing* and *limiting the number of Gaussians* with the purifier in our RemedyGS framework on poisoned data under varying and stronger attack strengths. The evaluation is conducted in terms of computation cost metrics on Mip-NeRF360. Our RemedyGS exhibits the strongest capability in defending against even more intensive computation cost attacks.

Metric	Number of Gaussians			Peak GPU memory [MB]			Training time [minutes]			Rendering speed [FPS]		
Scene \ Setting	image smoothing	limiting Gaussian number	RemedyGS (Ours)	image smoothing	limiting Gaussian number	RemedyGS (Ours)	image smoothing	limiting Gaussian number	ours	image smoothing	limiting Gaussian number	RemedyGS (Ours)
Constrained Poison-splat attack with $\epsilon = 26/255$												
MIP-bonsai	0.798 M	2.148 M	1.031 M	8912	11590	8731	17.04	19.58	17.77	307	168	260
MIP-counter	0.772 M	2.099 M	0.966 M	8693	10074	9133	19.92	20.38	20.79	247	194	201
MIP-kitchen	1.017 M	2.101 M	1.515 M	9861	10418	10521	22.79	21.97	23.08	199	146	149
MIP-room	0.961 M	2.150 M	1.102 M	11120	12185	10374	20.33	20.10	20.97	213	153	182
Constrained Poison-splat attack with $\epsilon = 35/255$												
MIP-bonsai	0.972 M	2.169 M	0.980 M	8659	10814	8603	17.80	20.03	17.59	272	163	264
MIP-counter	0.832 M	2.117 M	0.917 M	9190	9785	8870	20.55	20.00	20.26	218	204	215
MIP-kitchen	1.144 M	2.130 M	1.533 M	10056	10582	10340	23.44	22.44	23.14	180	146	148
MIP-room	1.289 M	2.167 M	1.015 M	13269	12652	10282	22.24	20.23	20.45	161	147	201
Constrained Poison-splat attack with $\epsilon = 40/255$												
MIP-bonsai	1.130 M	2.186 M	0.948 M	8850	11158	8586	18.28	18.19	17.56	137	169	261
MIP-counter	0.894 M	2.122 M	0.909 M	9548	9885	9237	20.72	19.26	17.43	207	200	205
MIP-kitchen	1.313 M	2.132 M	1.692 M	10508	10532	10790	24.09	20.15	22.26	229	148	137
MIP-room	1.625 M	2.159 M	0.986 M	15710	12514	10227	23.31	18.89	20.44	174	149	208

Table 11. Defense performance against adaptive computation cost attacks. The evaluation is conducted in terms of computation cost metrics on NeRF-Synthetic. Our RemedyGS also possesses the capability in defending against such more intensive computation cost attacks.

Metric	Number of Gaussians			Peak GPU memory [MB]			Training time [minutes]		
Scene \ Setting	clean (Ground truth)	Attack	RemedyGS (Ours)	clean (Ground truth)	Attack	RemedyGS (Ours)	clean (Ground truth)	Attack	RemedyGS (Ours)
Adaptive attack with $\beta = 1$									
NS-chair	0.495 M	0.901 M	0.485 M	4633	8410	4804	7.81	10.25	7.93
NS-ficus	0.267 M	0.285 M	0.220 M	4052	5984	4040	6.41	7.56	7.86
Adaptive attack with $\beta = 5$									
NS-chair	0.495 M	0.932 M	0.492 M	4633	8828	4892	7.81	10.78	8.66
NS-ficus	0.267 M	0.287 M	0.215 M	4052	6268	4028	6.41	7.90	6.33
Adaptive attack with $\beta = 20$									
NS-chair	0.495 M	0.931 M	0.492 M	4633	8772	4886	7.81	10.20	8.18
NS-ficus	0.267 M	0.298 M	0.213 M	4052	6302	4002	6.41	7.36	6.33

Table 12. Defense performance against adaptive computation cost attacks. The evaluation is conducted in terms of utility metrics on NeRF-Synthetic. The purifier of our RemedyGS still has the capability in recovering clean images even under more intensive computation cost attacks.

Metric	PSNR			LPIPS			SSIM		
Scene \ Setting	clean (Ground truth)	Attack	RemedyGS (Ours)	clean (Ground truth)	Attack	RemedyGS (Ours)	clean (Ground truth)	Attack	RemedyGS (Ours)
Adaptive attack with $\beta = 1$									
NS-chair	35.776	28.402	33.343	0.010	0.218	0.032	0.988	0.240	0.621
NS-ficus	35.542	28.488	34.295	0.012	0.323	0.017	0.987	0.218	0.610
Adaptive attack with $\beta = 5$									
NS-chair	35.776	29.223	33.256	0.010	0.287	0.033	0.988	0.278	0.584
NS-ficus	35.542	29.673	34.179	0.012	0.320	0.017	0.987	0.271	0.571
Adaptive attack with $\beta = 20$									
NS-chair	35.776	29.616	32.582	0.010	0.277	0.033	0.988	0.309	0.514
NS-ficus	35.542	30.062	33.633	0.012	0.351	0.022	0.987	0.296	0.487

Table 13. Impact of different ratios of undetected poisoned data in a scene on safety performance in term of computation cost metrics including the number of Gaussians, peak GPU memory, training time, and rendering speed.

Metric	Number of Gaussians			Peak GPU memory [MB]			Training time [minutes]			Rendering speed [FPS]		
Scene \ Setting	clean	attack	Mixed	clean	attack	Mixed	clean	attack	Mixed	clean	attack	Mixed
Mixed ratio: 1%												
NS-chair	0.495 M	0.942 M	0.492 M	4633	9378	4834	7.81	12.41	9.63	345	170	354
NS-ficus	0.267 M	0.288 M	0.220 M	4052	5447	4057	6.41	10.90	8.66	606	447	647
MIP-room	1.536 M	7.368 M	1.177 M	11370	47721	10879	22.14	48.11	20.71	154	31	191
Mixed ratio: 5%												
NS-chair	0.495 M	0.942 M	0.503 M	4633	9378	4922	7.81	12.41	8.69	345	170	338
NS-ficus	0.267 M	0.288 M	0.220 M	4052	5447	4050	6.41	10.90	8.02	606	447	615
MIP-room	1.536 M	7.368 M	1.178 M	11370	47721	10954	22.14	48.11	20.67	154	31	199
Mixed ratio: 10%												
NS-chair	0.495 M	0.942 M	0.520 M	4633	9378	4906	7.81	12.41	9.79	345	170	314
NS-ficus	0.267 M	0.288 M	0.223 M	4052	5447	4060	6.41	10.90	8.89	606	447	616
MIP-room	1.536 M	7.368 M	1.165 M	11370	47721	10977	22.14	48.11	20.47	154	31	196

Table 14. Comparison of baselines: *image smoothing* and *limiting the number of Gaussians* with our RemedyGS framework on poisoned data, evaluated in terms of computational cost metrics including training time and rendering speed across NeRF-Synthetic, Mip-NeRF360 and Tanks-and-Temples datasets. Our RemedyGS effectively mitigates the negative impact of computation cost attacks.

Metric	Training time [minutes]					Rendering speed [FPS]				
Scene \ Setting	clean	poisoned	image smoothing	limiting Gaussian number	ours	clean	poisoned	image smoothing	limiting Gaussian number	ours
NS-chair	7.81	12.41	9.71	8.61	10.07	345	170	645	371	348
NS-ficus	6.41	10.90	9.08	9.16	9.48	606	447	797	471	701
NS-hotdog	7.68	16.23	9.31	9.98	10.85	655	100	852	356	374
NS-lego	7.36	13.77	9.96	9.52	10.34	483	194	702	403	445
NS-Avg	7.30	12.87	9.66	9.61	10.17	566	236	786	373	489
MIP-bonsai	16.66	33.72	13.98	18.11	18.63	222	60	290	178	277
MIP-counter	22.21	35.46	17.79	20.67	23.7	175	55	240	192	213
MIP-kitchen	22.84	41.99	19.56	20.05	23.64	138	46	205	157	168
MIP-room	22.14	48.11	17.26	19.45	23.86	154	31	225	147	185
MIP-Avg	25.37	40.62	18.94	24.94	24.59	128	51	197	129	157
TT-Barn	13.54	17.03	12.19	10.52	11.61	265	135	481	388	343
TT-Francis	10.19	13.47	10.47	9.71	10.15	300	159	497	349	477
TT-M60	13.85	19.32	11.68	14.45	10.88	188	103	331	194	301
TT-Panther	14.68	20.58	9.63	14.56	10.07	179	95	390	205	309
TT-Avg	15.27	19.60	11.71	14.63	12.59	194	122	347	231	288

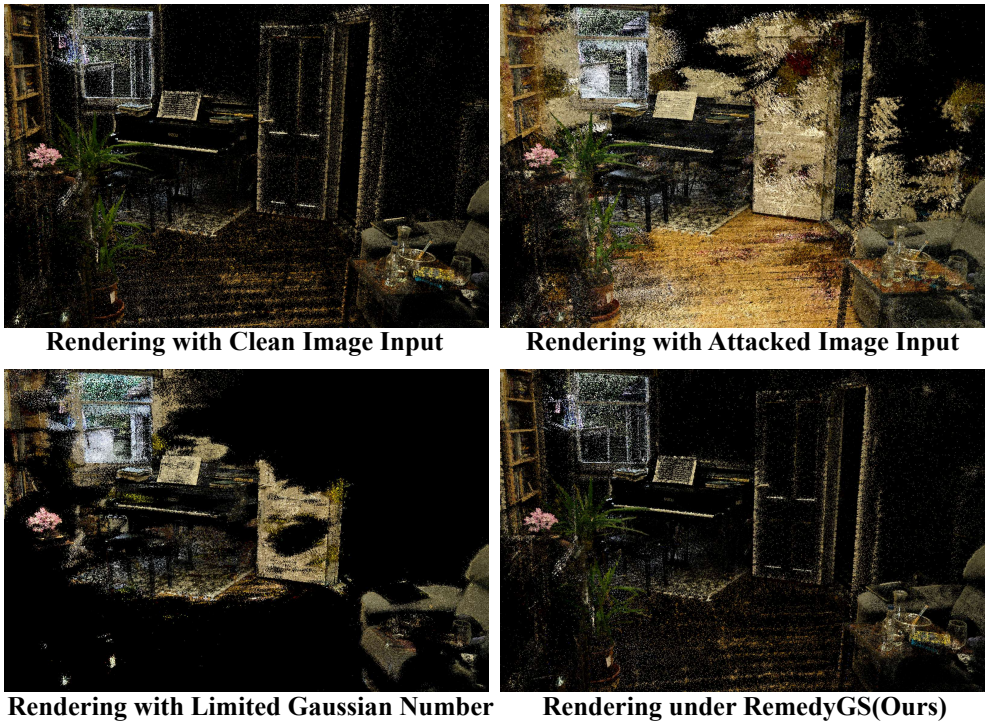

Figure 5. 3D Gaussian point cloud visualization of rendering results from different input images. Top row: (Left) Point cloud visualization of rendering from clean image input. (Right) Point cloud visualization of rendering from attacked image input. Bottom row: (Left) Point cloud visualization of rendering from the limiting Gaussian number defense method. (Right) Point cloud visualization of rendering from our RemedyGS method.

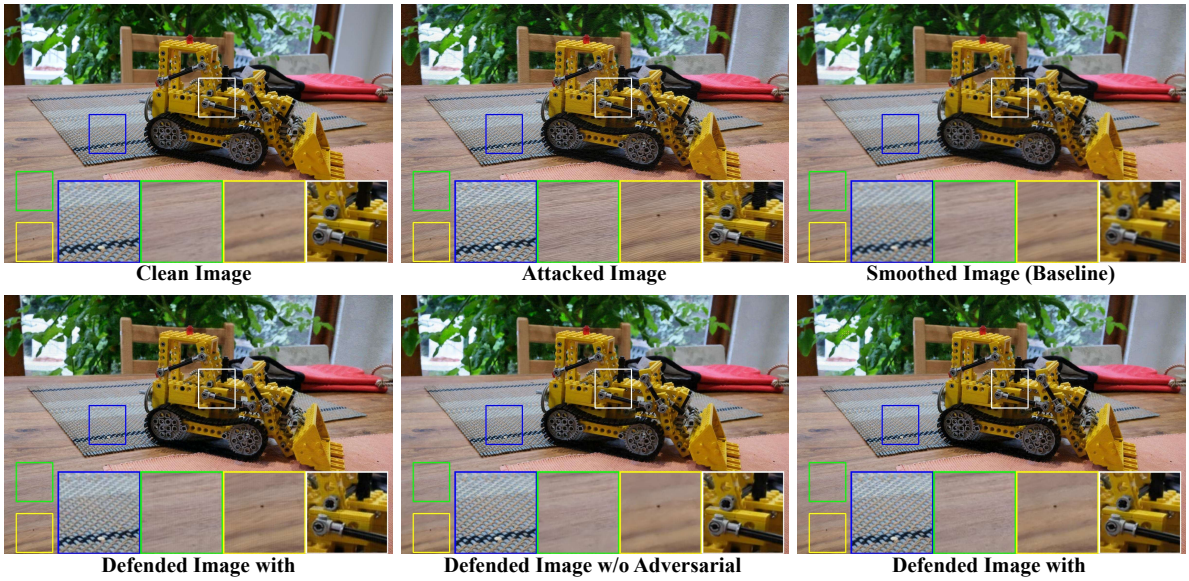

Figure 6. Qualitative results for the kitchen scene in the Mip-NeRF360 dataset. Top row: (left) clean input image, (middle) attacked input image, (right) defended input image using the image smoothing baseline. Bottom row: (left) purified input image using the purifier with concatenated skip connections, (middle) purified input image using the purifier with added skip connections but without adversarial training, (right) purified input image using the purifier in our RemedyGS framework. Our method achieves the best recovery performance among all baselines and effectively mitigates the issue of local over-smoothing compared with methods without adversarial training.

9. Derivation for the Optimal Discriminator

In this section, we rigorously derive the expression of the optimal discriminator in Eq. (12). Note that

$$\mathcal{L}_{\mathcal{F}} = \min_{\mathcal{F}} -\mathbb{E}_{p_{\text{cln}}} \log(\mathcal{F}(\boldsymbol{V}|\boldsymbol{c})) - \mathbb{E}_{p_{\text{rec}}} \log(1 - \mathcal{F}(\boldsymbol{V}|\boldsymbol{c})) \tag{16}$$

$$= \min_{\mathcal{F}} \int_{\boldsymbol{V}} \big\{ - p_{cln}(\boldsymbol{V}) \log(\mathcal{F}(\boldsymbol{V}|c)) - p_{rec}(\boldsymbol{V}) \log(1 - \mathcal{F}(\boldsymbol{V}|c)) \big\} \mathrm{d}\boldsymbol{V}. \tag{17}$$

For any $\alpha \in \mathbb{R} \backslash \{0\}$ and $\beta \in \mathbb{R} \backslash \{0\}$, considering a function $f(x) = \alpha \log(x) + \beta \log(1 - x)$ with $x \in [0, 1]$, $f(x)$ achieves its maximum at $x = \frac{\alpha}{\alpha+\beta}$. Therefore, given a fixed purifier, the optimal discriminator $\mathcal{F}^*$ is given by

$$\mathcal{F}^* = \frac{p_{\text{cln}}(\boldsymbol{V})}{p_{\text{cln}}(\boldsymbol{V}) + p_{\text{rec}}(\boldsymbol{V})}. \tag{18}$$

10. Derivation for the Maximum Adversarial Training Objective

In this section, we provide the detailed derivation of the conclusion that Eq. (13) is maximized if and only if the reconstructed distribution matches the clean distribution. We have

$$\mathcal{L}_G = \max_{\phi,\theta} \{-\mathbb{E}_{p_{\text{cln}}} \log(\mathcal{F}^*(\boldsymbol{V}|\boldsymbol{c})) - \mathbb{E}_{p_{\text{rec}}} \log(1 - \mathcal{F}^*(\boldsymbol{V}|\boldsymbol{c}))\} \tag{19}$$

$$= \max_{\phi,\theta} \left\{ -\mathbb{E}_{p_{\text{cln}}} \log \left(\frac{p_{\text{cln}}(\boldsymbol{V})}{p_{\text{cln}}(\boldsymbol{V}) + p_{\text{rec}}(\boldsymbol{V})} \right) - \mathbb{E}_{p_{\text{rec}}} \log \left(\frac{p_{\text{rec}}(\boldsymbol{V})}{p_{\text{cln}}(\boldsymbol{V}) + p_{\text{rec}}(\boldsymbol{V})} \right) \right\} \tag{20}$$

$$= \max_{\phi,\theta} \left\{ -\mathbb{E}_{p_{\text{cln}}} \log \left(\frac{p_{\text{cln}}(\boldsymbol{V})}{2 \cdot \frac{p_{\text{cln}}(\boldsymbol{V}) + p_{\text{rec}}(\boldsymbol{V})}{2}} \right) - \mathbb{E}_{p_{\text{rec}}} \log \left(\frac{p_{\text{rec}}(\boldsymbol{V})}{2 \cdot \frac{p_{\text{cln}}(\boldsymbol{V}) + p_{\text{rec}}(\boldsymbol{V})}{2}} \right) \right\} \tag{21}$$

$$= \max_{\phi,\theta} \left\{ 2 \log 2 - \mathbb{E}_{p_{\text{cln}}} \log \left(\frac{p_{\text{cln}}(\boldsymbol{V})}{\frac{p_{\text{cln}}(\boldsymbol{V}) + p_{\text{rec}}(\boldsymbol{V})}{2}} \right) - \mathbb{E}_{p_{\text{rec}}} \log \left(\frac{p_{\text{rec}}(\boldsymbol{V})}{\frac{p_{\text{cln}}(\boldsymbol{V}) + p_{\text{rec}}(\boldsymbol{V})}{2}} \right) \right\} \tag{22}$$

$$= \max_{\phi,\theta} \left\{ 2 \log 2 - KL \left(p_{\text{cln}}(\boldsymbol{V}) || \frac{p_{\text{cln}}(\boldsymbol{V}) + p_{\text{rec}}(\boldsymbol{V})}{2} \right) - KL \left(p_{\text{rec}}(\boldsymbol{V}) || \frac{p_{\text{cln}}(\boldsymbol{V}) + p_{\text{rec}}(\boldsymbol{V})}{2} \right) \right\}, \tag{23}$$

where $KL(p_{\text{cln}}(\boldsymbol{V})||\frac{p_{\text{cln}}(\boldsymbol{V})+p_{\text{rec}}(\boldsymbol{V})}{2})$ and $KL(p_{\text{rec}}(\boldsymbol{V})||\frac{p_{\text{cln}}(\boldsymbol{V})+p_{\text{rec}}(\boldsymbol{V})}{2})$ denote the Kullback–Leibler divergence. According to the definition of the Jensen-Shannon divergence, we have

$$\mathcal{L}_G = \max_{\phi,\theta} 2 \log 2 - JSD(p_{\text{cln}}(\boldsymbol{V})||p_{\text{rec}}(\boldsymbol{V})), \tag{24}$$

where JSD denotes the Jensen-Shannon divergence. Since JSD is nonnegative, $\mathcal{L}_G$ achieves its maximum when $JSD(p_{\text{cln}}(\boldsymbol{V})||p_{\text{rec}}(\boldsymbol{V})) = 0$, which occurs if and only if $p_{\text{rec}}(\boldsymbol{V}) = p_{\text{cln}}(\boldsymbol{V})$. Therefore, $\mathcal{L}_G$ is maximized precisely when the reconstructed distribution matches the clean distribution.

References

[1] Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *International conference on machine learning*, pages 2196–2205. PMLR, 2020. 1

[2] Ye Liu, Yaya Cheng, Lianli Gao, Xianglong Liu, Qilong Zhang, and Jingkuan Song. Practical evaluation of adversarial robustness via adaptive auto attack. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15105–15114, 2022. 1

[3] Jiahao Lu, Yifan Zhang, Qiuhong Shen, Xinchao Wang, and Shuicheng Yan. Poison-splat: Computation cost attack on 3d gaussian splatting. *arXiv preprint arXiv:2410.08190*, 2024. 1

[4] Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20654–20664, 2024. 1

[5] Jeonghwan Park, Niall McLaughlin, and Ihsen Alouani. Mind

the gap: Detecting black-box adversarial attacks in the making through query update analysis. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10235–10243, 2025. 1

[6] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. *Advances in neural information processing systems*, 33:1633–1645, 2020. 1