EFFGS: EFFICIENT AND HIGH-FIDELITY GAUSSIAN SPLATTING

Anonymous authors
Paper under double-blind review

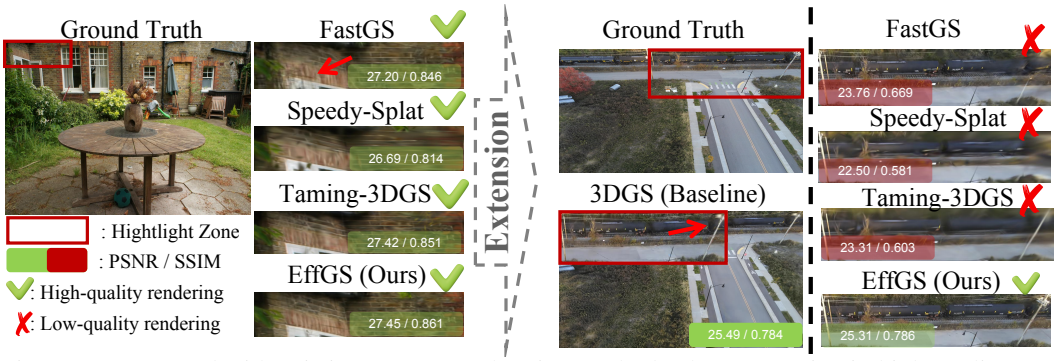


Figure 1: Compared with existing 3DGS acceleration methods, they can maintain high-quality rendering in small-scale scenes but suffer from degraded rendering quality when scaled up to city-level scenarios. In contrast, EffGS sustains high-fidelity rendering consistently, demonstrating superior scalability.

ABSTRACT

3D Gaussian Splatting (3DGS) enables real-time novel view synthesis, but existing general-purpose acceleration methods suffer severe rendering quality degradation when extended to more complex, large-scale scenes. To address this issue, we propose EffGS, a more general acceleration framework that improves training and rendering efficiency while maintaining reconstruction quality comparable to or better than vanilla 3DGS across bounded and large-scale scenes. EffGS combines frequency-aware guidance, localized density control, and adaptive primitive scale modulation. First, an importance scoring mechanism combines pixel-wise reconstruction errors with a difference-of-Gaussians mask scheduled over training to provide stage-dependent spatial guidance. Second, localized densification and pruning restricts density modifications to Gaussians with valid projected footprints in the sampled views. Third, learnable per-Gaussian scale modulation adjusts effective primitive extent during optimization while retaining the Compact Box rasterization rule. Extensive experiments on bounded and large-scale scene datasets demonstrate a favorable balance between reconstruction quality, training time, and primitive count. Component ablations and matched-primitive-budget comparisons further support the effectiveness of the framework.

1 INTRODUCTION

Neural Radiance Fields (NeRF) (Mildenhall et al., 2020) advanced novel view synthesis (NVS) (Reiser et al., 2023), but their computational demands motivated more efficient scene representations. 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) uses explicit Gaussian primitives and a tile-based rasterizer to achieve real-time, photorealistic rendering. However, adaptive density control can generate redundant primitives, increasing optimization and rasterization costs (Fan et al., 2024). These costs become particularly important when reconstructing large-scale environments. Recent acceleration methods improve density control through budget constraints (Mallick et al., 2024; Chen et al., 2025), targeted pruning (Hanson et al., 2025a), and multi-view evaluation (Ren et al., 2025). These approaches reduce computational cost, but preserving rendering quality under

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.

aggressive primitive reduction remains challenging. Our city-scale comparisons (Tab. 2 and Fig. 6) illustrate this trade-off. We focus on three aspects of the optimization pipeline: the spatial information used to score primitives, the sampled views supporting density-control decisions, and the effective extent of each Gaussian.

1. **spatial detail should inform primitive allocation**. Reconstruction errors identify regions that remain difficult to render, while scale-dependent image responses provide complementary information about local structure. Combining these signals allows density control to emphasize selected image structures at different training stages. The spectral magnitude comparisons and spatial visualizations in Fig. 2 further motivate evaluating detail preservation alongside standard reconstruction metrics.

2. **density-control decisions should account for sampled-view coverage**. In large-scale scenes, a small set of sampled cameras typically covers only part of the Gaussian representation. FastGS (Ren et al., 2025) evaluates primitives using reconstruction errors from randomly sampled views. When applying view-based scores, primitives outside the sampled coverage should be distinguished from those with valid projected footprints. Restricting density modifications to the latter localizes the operation to the currently covered portion of the scene. The FastGS+LDP comparison in Tab. 11 supports the benefit of this restriction.

3. **primitive extent affects rasterization cost**. Vanilla 3DGS (Kerbl et al., 2023) uses a three-sigma extent to construct rasterization bounds. Speedy-Splat (Hanson et al., 2025a) reduces redundant Gaussian–tile pairs through precise tile intersection, and FastGS (Ren et al., 2025) further restricts support using its Compact Box mechanism. These culling rules operate on projected Gaussian geometry. Adapting effective scales during optimization therefore complements tile culling by changing the spatial support to which the rule is applied.

We introduce **EffGS**, a framework that combines frequency-aware scoring, localized density control, and adaptive primitive compactness. Our scoring mechanism couples pixel-wise reconstruction errors with a scheduled difference-of-Gaussians mask. The mask provides two-stage, scale-dependent spatial guidance for importance scoring and reconstruction supervision. Localized Densification and Pruning (LDP) restricts density operations to Gaussians with valid projected footprints in the sampled views, excluding primitives outside the current active set. Finally, a learnable scalar γ_i uniformly modulates each Gaussian's three base-scale components. Joint optimization of this factor and the base scales provides an additional pathway for adapting effective primitive extent while preserving the Compact Box tile-culling rule. Experiments on bounded and urban-scale datasets show that EffGS combines high reconstruction quality with efficient training and real-time rendering. Component ablations support the contribution of each module, while matched-primitive-budget comparisons evaluate the framework under controlled Gaussian counts. We additionally report band-wise spectral magnitude errors and high-frequency response discrepancies to complement standard rendering metrics (Sec. A.4). Our main contributions are summarized as follows:

- **Frequency-Aware Optimization:** We combine reconstruction errors with scheduled difference-of-Gaussians masks to provide stage-dependent spatial guidance for primitive allocation and reconstruction supervision.
- **Localized Density Control (LDP):** We restrict densification and pruning to Gaussians with valid projected footprints in sampled views, improving reconstruction quality in the evaluated large-scale scenes.
- **Adaptive Primitive Compactness:** We introduce learnable per-Gaussian scale modulation that interacts with optimization and density control to improve the quality–efficiency trade-off while retaining the Compact Box rasterization rule.

2 Related Work

Neural Rendering. Novel View Synthesis has attracted significant attention in computer vision and graphics (Zhu et al., 2025). Neural Radiance Fields (NeRF) Mildenhall et al. (2020) implicitly parameterize scene information via a multi-layer perceptron network and achieve high-quality image synthesis through volume rendering, greatly advancing NVS (Turki et al., 2022a; Mi & Xu, 2023). However, due to its slow rendering efficiency, 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) was subsequently introduced. 3DGS explicitly represents scenes with Gaussian primitives and employs

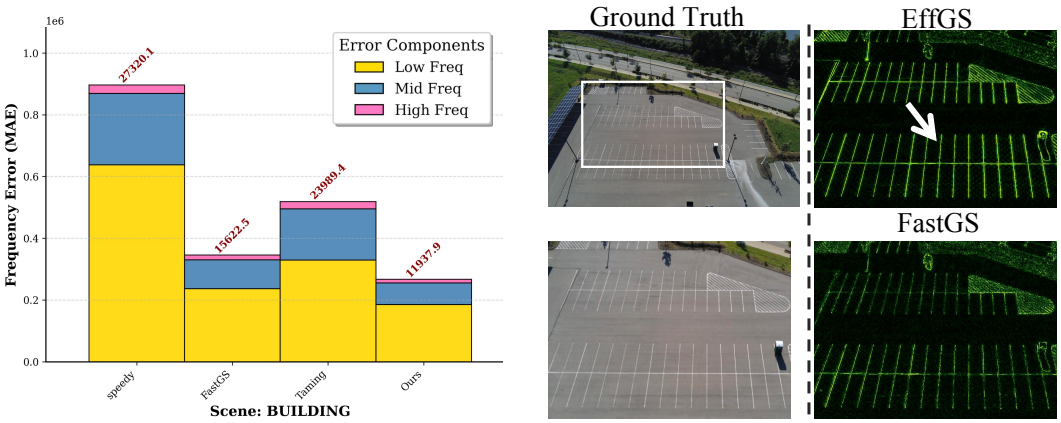


Figure 2: Spectral magnitude errors (left) and high-frequency discrepancy visualizations with ground-truth references (right) on Building (Turki et al., 2022b); see Sec. A.4.

GPU-friendly rasterization with alpha compositing for image rendering, thereby significantly accelerating optimization and enabling real-time rendering. Subsequently, a series of methods based on 3DGS have been proposed to enhance rendering quality (Yu et al., 2024a; Li et al., 2025; Chen et al., 2024; Yu et al., 2024b). Although 3DGS (Kerbl et al., 2023) achieves impressive rendering results, the redundancy of Gaussian primitives considerably reduces its efficiency (Fan et al., 2024; Li et al., 2025; Hamdi et al., 2024; Hanson et al., 2025b; Höllein et al., 2024), especially when extending to city-scale scenes where multi-GPU training is often used to improve training efficiency (Li et al., 2026; Liu et al., 2024a; Gao et al., 2025; Liu et al., 2024b; Lin et al., 2024), leading to increased training time. To accelerate 3DGS, many recent works have made substantial efforts.

3DGS Acceleration. Density control is a major direction for accelerating 3DGS (Ren et al., 2025). For densification, recent works (Chen et al., 2025; Mallick et al., 2024; Kim et al., 2024; Bulò et al., 2024) refine primitive allocation through mechanisms such as Gaussian budgets (Mallick et al., 2024) and resolution-guided scheduling (Chen et al., 2025). For pruning, methods reduce the representation using importance estimates and selection strategies (Fan et al., 2024; Li et al., 2025; Fang & Wang, 2024; Hanson et al., 2025a;b), including Gaussian attributes, sampling-based reduction (Fang & Wang, 2024), and approximate second-order information (Hanson et al., 2025b;a). FastGS (Ren et al., 2025) evaluates pixel-wise reconstruction errors from randomly sampled views to guide densification and pruning. These methods offer different trade-offs between primitive count, computational cost, and rendering quality. EffGS builds on error-based scoring and Compact Box rasterization by incorporating scale-dependent spatial guidance, restricting density operations to the sampled-view active set, and introducing learnable per-Gaussian scale modulation. The combined framework targets efficient training with high reconstruction fidelity across scene scales.

3 METHOD

We first revisit the basics of 3DGS (Sec. 3.1). We then introduce frequency-aware importance scoring, which combines pixel-wise reconstruction errors with scheduled difference-of-Gaussians masks (Sec. 3.2). Localized densification and pruning restricts density modifications to Gaussians with valid projected footprints in the sampled views (Sec. 3.3). Next, adaptive primitive compactness uses learnable per-Gaussian scale modulation to adjust effective spatial support during optimization (Sec. 3.4). Finally, we describe the training objective that combines photometric, structural, mask-weighted, and scale-factor regularization terms (Sec. 3.5). Additional frequency-domain analysis, including spectral decomposition and band-wise reconstruction-error measurements, is provided in the supplementary material (Sec. A.4.2).

3.1 PRELIMINARY

We briefly review 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023). A scene is represented by a set of 3D Gaussians $\{G_k\}$. Each Gaussian has position $\mathbf{p}_k$, opacity o_k, rotation R_k, base-scale matrix

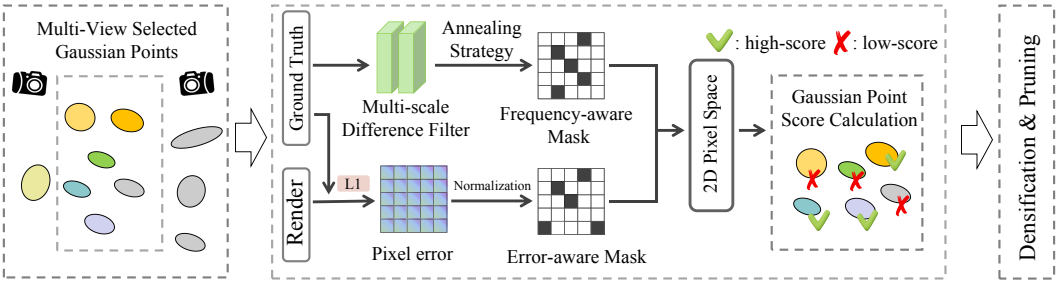


Figure 3: **Overview of the proposed unified optimization pipeline.** Given a set of sampled views, we first identify the visible Gaussians. We then compute a unified 2D supervision signal by fusing a pixel-wise error-aware mask with a frequency-aware mask (extracted via an annealing schedule). Finally, we project the visible Gaussians to 2D to accumulate importance scores from the unified mask, guiding localized densification and pruning.

$S_k = \mathrm{diag}(s_k^x, s_k^y, s_k^z)$, and covariance $\Sigma_k = R_k S_k S_k^\top R_k^\top$. Its spherical-harmonic coefficients determine the view-dependent RGB color $\mathbf{c}_k$.

For rendering, the covariance is projected to the image plane as

$$\Sigma_k^{\mathrm{2D}} = JW\Sigma_k W^\top J^\top, \tag{1}$$

where $W \in \mathbb{R}^{3\times3}$ is the rotational part of the world-to-camera transform and J is the projection Jacobian evaluated at $\mathbf{p}_k$. For image-plane coordinate $\mathbf{u} \in \mathbb{R}^2$, let $\mathbf{p}_k^{\mathrm{2D}}$ denote the projected Gaussian center. The rendered color is obtained by front-to-back alpha compositing:

$$\hat{\mathbf{C}}(\mathbf{u}) = \sum_k \mathbf{c}_k \alpha_k(\mathbf{u}) \prod_{\ell<k} (1 - \alpha_\ell(\mathbf{u})), \tag{2}$$

$$\alpha_k(\mathbf{u}) = o_k \exp\left(-\frac{1}{2}(\mathbf{u} - \mathbf{p}_k^{\mathrm{2D}})^\top (\Sigma_k^{\mathrm{2D}})^{-1}(\mathbf{u} - \mathbf{p}_k^{\mathrm{2D}})\right). \tag{3}$$

3DGS uses adaptive density control: Gaussians with large positional gradients are cloned or split, while those with low opacity or excessive scale are pruned. The standard objective is

$$\mathcal{L} = (1 - \lambda_{\mathrm{dssim}})\mathcal{L}_1 + \lambda_{\mathrm{dssim}}(1 - \mathcal{L}_{\mathrm{SSIM}}), \tag{4}$$

where $\lambda_{\mathrm{dssim}} = 0.2$.

3.2 Frequency-Aware Importance Scoring

We combine reconstruction errors with scheduled image responses to guide primitive allocation (Fig. 3).

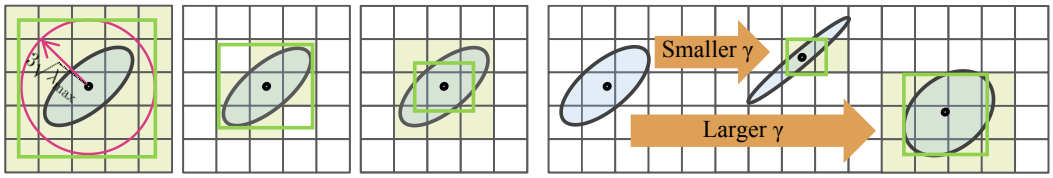
Frequency mask generation. We convert each ground-truth image to grayscale and downsample it to obtain I_{down}. Let $\mathcal{G}_\sigma$ denote a normalized Gaussian kernel with standard deviation σ. We compute

$$H = |I_{\mathrm{down}} * \mathcal{G}_{\sigma_1} - I_{\mathrm{down}} * \mathcal{G}_{\sigma_2}|, \qquad \sigma_2 = 2\sigma_1, \tag{5}$$

where $*$ denotes convolution. The bandwidth follows the piecewise schedule in Sec. C.2, controlled by $f(t) = 100t/T_{\mathrm{total}}$, where t is the current training iteration and T_{total} is the total number of iterations. After upsampling and normalization, we threshold the complemented response for $f < 50$ and the direct response for $f \geq 50$ to obtain the frequency-aware mask M_{freq}. This provides stage-dependent spatial guidance, with the decreasing bandwidth in the second stage emphasizing finer structures.

Composite error map. For sampled view j with pixel grid $\mathcal{P}_j$, let I^j_{rend} and I^j_{gt} denote the rendered and ground-truth images, respectively, and let M^j_{freq} be the corresponding frequency-aware mask. The raw RGB reconstruction error is

$$e^j(\mathbf{u}) = \frac{1}{3}\sum_{c=1}^{3} \left| I^j_{\mathrm{rend},c}(\mathbf{u}) - I^j_{\mathrm{gt},c}(\mathbf{u}) \right|, \qquad \bar{e}^j = \mathcal{N}_{\mathcal{P}_j}[e^j],$$



(a) 3DGS (b) Speedy-Splat (c) FastGS (d) Adaptive Primitive Compactness (Ours)

Figure 4: **Learnable primitive compactness.** Compared with vanilla 3DGS (Kerbl et al., 2023), Speedy-Splat (Hanson et al., 2025a), and FastGS (Ren et al., 2025), our method learns per-Gaussian compactness, reducing redundant pairs and improving fidelity.

where $\mathcal{N}_{\mathcal{S}}$ denotes min–max normalization over index set $\mathcal{S}$, with constant inputs mapped to zero. The composite mask is

$$M^j_{\text{err}}(\mathbf{u}) = \mathbb{I}\Big[\bar{e}^j(\mathbf{u}) > \tau \ \vee \ \Big(M^j_{\text{freq}}(\mathbf{u}) = 1 \ \wedge \ \bar{e}^j(\mathbf{u}) > 0.5\tau\Big)\Big], \tag{6}$$

where $\mathbb{I}[\cdot]$ is the indicator function and $\tau \in (0, 1)$ is the normalized error threshold.

Densification scores. Let $\mathcal{V}_i$ contain the sampled views in which Gaussian i has a valid projected footprint, and let $\mathcal{A}_t$ denote the active primitive set at the current density update (Sec. 3.3). For each view, Ω^j_i contains the pixels at which Gaussian i is processed and contributes during compositing. Define

$$n^j_i = \sum_{\mathbf{u}\in\Omega^j_i} M^j_{\text{err}}(\mathbf{u}).$$

The valid-view average and densification score are

$$C_i = \frac{1}{|\mathcal{V}_i|} \sum_{j\in\mathcal{V}_i} n^j_i, \qquad i \in \mathcal{A}_t, \tag{7}$$

$$s^i_d(t) = \omega(t) C_i, \tag{8}$$

where $\omega(t)$ is a positive, nondecreasing clipped-linear schedule over the densification interval. Exact normalization, pixel-support, and scheduling definitions are provided in Sec. C.3.

Pruning scores. Following FastGS (Ren et al., 2025), we weight each per-view count by the corresponding image-level photometric loss:

$$E^j_{\text{photo}} = (1 - \lambda_{\text{dssim}})\mathcal{L}^j_1 + \lambda_{\text{dssim}} \left(1 - \text{SSIM}(I^j_{\text{rend}}, I^j_{\text{gt}})\right), \qquad \mathcal{L}^j_1 = \frac{1}{|\mathcal{P}_j|} \sum_{\mathbf{u}\in\mathcal{P}_j} e^j(\mathbf{u}).$$

The pruning score is

$$Q_i = \sum_{j\in\mathcal{V}_i} n^j_i E^j_{\text{photo}}, \qquad s^i_p = \mathcal{N}_{\mathcal{A}_t}[Q](i). \tag{9}$$

Thus, densification uses a valid-view average, whereas pruning retains summed, loss-weighted counts. E^j_{photo} is used only for scoring; parameter optimization follows Sec. 3.5.

3.3 View-Visible Local Densification and Pruning

We restrict density modifications to the portion of the representation covered by the sampled views.

Active primitives. At each scheduled update, we sample K distinct training views uniformly without replacement. Let $\mathcal{G}_t$ denote the index set of Gaussian primitives before the update, and let r^j_i be the projected radius returned by the rasterizer. We define

$$\mathcal{V}_i = \{j : r^j_i > 0\}, \qquad |\mathcal{V}_i| = \sum_{j=1}^{K} \mathbb{I}[r^j_i > 0], \qquad \mathcal{A}_t = \{i \in \mathcal{G}_t : |\mathcal{V}_i| > 0\}. \tag{10}$$

This criterion defines projected-footprint eligibility rather than explicit occlusion visibility; actual counted contributions are determined by Ω^j_i. Density control is skipped when $\mathcal{A}_t = \varnothing$.

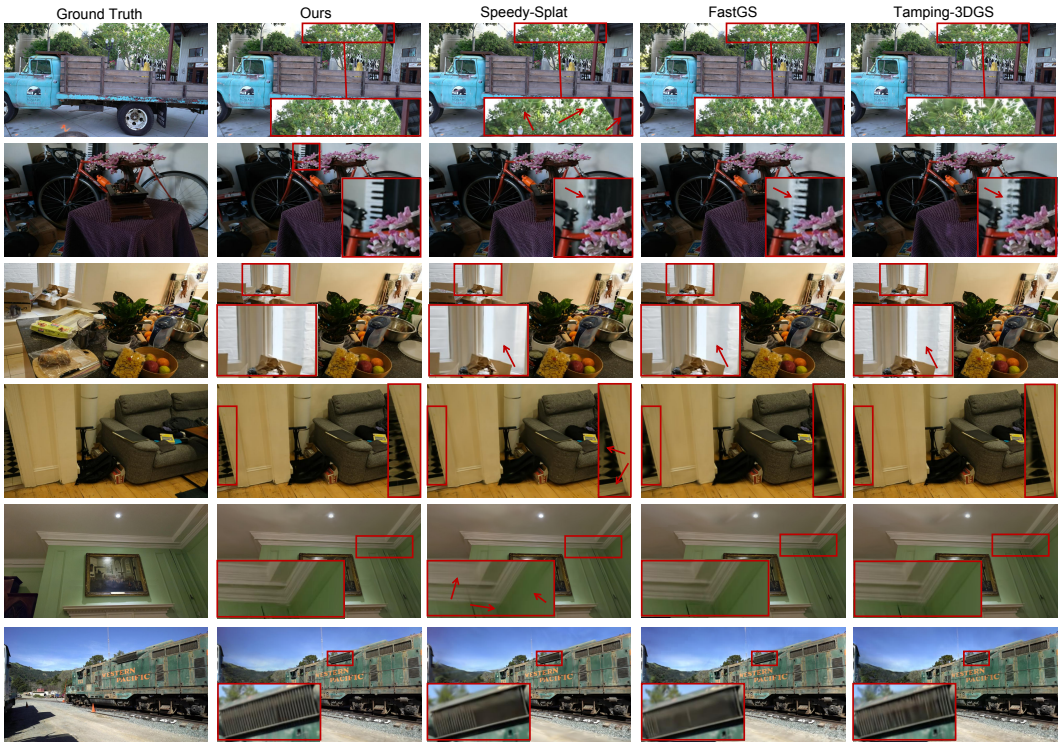


Figure 5: Qualitative results of ours and other methods in image rendering on Deep Blending (Park et al., 2019), Mip-NeRF 360 (Barron et al., 2022) and Tanks & Temples datasets (Knapitsch et al., 2017).

Pruning. At a pruning step, mandatory removals are

$$\mathcal{H}_t = \{i \in \mathcal{A}_t : o_i < 0.1 \ \vee \ \gamma_i < 0.01\},$$

where γ_i is the learnable scale-modulation factor introduced in Sec. 3.4. The remaining candidates are

$$\mathcal{P}_t = \left\{i \in \mathcal{A}_t \setminus \mathcal{H}_t : s_p^i > \tau_p\right\}, \qquad 0 \le \tau_p < 1.$$

We sample $b_t = \lfloor \rho |\mathcal{P}_t| \rfloor$ candidates without replacement using weights proportional to s_p^i, where $\rho \in [0, 1]$. The removal set $\mathcal{R}_t$ is their union with $\mathcal{H}_t$.

Densification and execution. At a densification step, surviving active primitives satisfying $s_d^i(t) > \tau_d$ are selected, where $\tau_d \ge 0$ is the densification-score threshold. A primitive is cloned and shifted along its positional gradient if its largest base-scale component is below $\tau_s > 0$; otherwise, it is replaced by two smaller Gaussians. New primitives inherit the parent's unconstrained modulation parameter $\beta_i \in \mathbb{R}$, with $\gamma_i = 2\,\mathrm{sigmoid}(\beta_i) \in (0, 2)$, while their remaining attributes follow the underlying clone/split updates. All scores are computed from the same pre-update representation. When pruning and densification coincide, pruning is applied first; new primitives are scored only at the next density update. Primitives outside $\mathcal{A}_t$ undergo neither operation. The complete execution protocol is given in Sec. C.3.

3.4 Adaptive Primitive Compactness

During rasterization, the projected extent of each Gaussian determines the tiles that require processing. Vanilla 3DGS (Kerbl et al., 2023) uses a three-sigma extent, while Speedy-Splat (Hanson et al., 2025a) and FastGS (Ren et al., 2025) reduce redundant Gaussian–tile evaluations through more restrictive culling rules. We retain the Compact Box mechanism of FastGS and introduce a learnable per-Gaussian scale modulation. For Gaussian i,

$$\widetilde{S}_i = \gamma_i S_i, \qquad \widetilde{\Sigma}_i = \gamma_i^2 \Sigma_i. \tag{11}$$

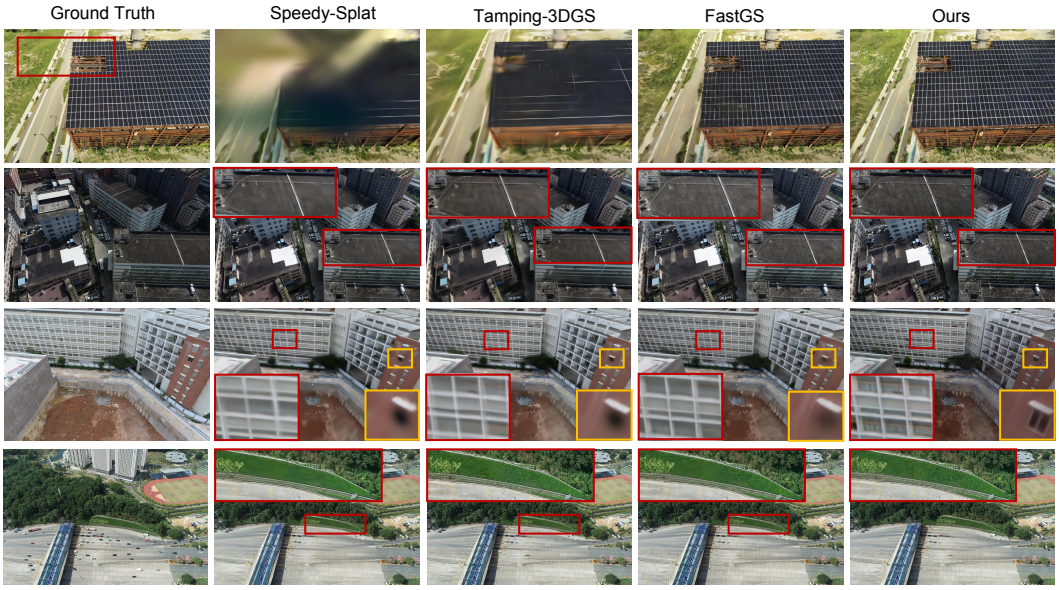


Figure 6: Qualitative results of ours and other methods in image rendering on Mill-19 (Turki et al., 2022b), Urbanscene3D (Lin et al., 2022) and GauU-Scene datasets (Xiong et al., 2024).

Table 1: Quantitative comparison on Mip-NeRF 360 (Barron et al., 2022), Deep Blending (Hedman et al., 2018), and Tanks & Temples (Knapitsch et al., 2017).

Method	Mip-NeRF 360						Deep Blending						Tanks & Temples					
	Time ↓	PSNR ↑	SSIM ↑	LPIPS ↓	N_{GS} ↓	FPS ↑	Time ↓	PSNR ↑	SSIM ↑	LPIPS ↓	N_{GS} ↓	FPS ↑	Time ↓	PSNR ↑	SSIM ↑	LPIPS ↓	N_{GS} ↓	FPS ↑
3DGS	20.93	27.53	0.812	0.221	2.63M	146	19.77	29.71	0.903	0.241	2.46M	158	11.34	23.71	0.850	0.170	1.57M	195
3DGS-LM	10.37	27.42	0.810	0.222	3.36M	220	9.83	29.61	0.905	0.248	2.71M	248	6.15	23.53	0.842	0.184	1.81M	289
Mini-Splatting	14.67	27.32	0.821	0.217	0.53M	567	13.35	29.99	0.907	0.244	0.56M	624	9.06	23.46	0.844	0.181	0.30M	756
Speedy-Splat	13.38	26.91	0.781	0.295	0.30M	552	10.75	29.42	0.898	0.272	0.25M	664	6.32	23.38	0.816	0.242	0.18M	691
Taming-3DGS	5.14	27.48	0.794	0.261	0.68M	221	3.06	29.50	0.894	0.278	0.29M	352	2.71	23.89	0.833	0.214	0.32M	379
DashGaussian	6.35	27.73	0.817	0.218	2.40M	155	4.16	29.65	0.906	0.246	1.94M	208	4.28	24.00	0.853	0.178	1.21M	240
FastGS	1.93	27.56	0.797	0.261	0.40M	579	1.28	30.03	0.901	0.270	0.22M	714	1.32	24.15	0.839	0.210	0.24M	655
Ours	3.07	27.77	0.826	0.207	0.48M	478	2.94	30.01	0.910	0.238	0.25M	632	1.94	24.22	0.861	0.166	0.44M	572

With the base scales fixed, γ_i changes the overall spatial extent while preserving anisotropic axis ratios. Because both S_i and γ_i remain learnable, the modulation acts as an additional optimization and density-control variable rather than enlarging the family of representable covariances. It therefore provides an additional pathway for adapting effective primitive support while retaining the Compact Box tile-culling rule. Further details are provided in Sec. C.1.

3.5 Optimization

We further use M_{freq} to spatially weight reconstruction errors during training. For the currently sampled view, let $\mathcal{P}_{\text{img}}$ denote its pixel grid and $C_{\text{rgb}} = 3$. Using I_{rend} and I_{gt} for the rendered and ground-truth images,

$$\mathcal{L}_1 = \frac{1}{C_{\text{rgb}}|\mathcal{P}_{\text{img}}|} \sum_{\mathbf{u}\in\mathcal{P}_{\text{img}}} \sum_{c=1}^{C_{\text{rgb}}} |I_{\text{rend},c}(\mathbf{u}) - I_{\text{gt},c}(\mathbf{u})| \,.$$

The frequency-guided loss is

$$\mathcal{L}_{\text{freq}} = \frac{1}{C_{\text{rgb}}|\mathcal{P}_{\text{img}}|} \sum_{\mathbf{u}\in\mathcal{P}_{\text{img}}} \sum_{c=1}^{C_{\text{rgb}}} M_{\text{freq}}(\mathbf{u}) \, |I_{\text{rend},c}(\mathbf{u}) - I_{\text{gt},c}(\mathbf{u})| \,. \tag{12}$$

The mask is shared across color channels and uses the full-image denominator, so an empty mask yields zero masked loss without a separate normalization rule. Let $\mathcal{L}_{\text{SSIM}} = \text{SSIM}(I_{\text{rend}}, I_{\text{gt}})$. The complete objective is

$$\begin{aligned} \mathcal{L} = (1 - \lambda_{\text{dssim}} - \lambda_{\text{freq}})\mathcal{L}_1 + \lambda_{\text{dssim}}(1 - \mathcal{L}_{\text{SSIM}}) \\ + \lambda_{\text{freq}}\mathcal{L}_{\text{freq}} + \mathcal{R}_\gamma, \end{aligned} \tag{13}$$

Table 2: Quantitative comparison of the averaged metrics on Mill-19 (Turki et al., 2022b), UrbanScene3D (Lin et al., 2022), and GauU-Scene (Xiong et al., 2024).

Method	Mill-19						UrbanScene3D						GauU-Scene					
	Time (m) ↓	SSIM ↑	PSNR ↑	LPIPS ↓	N_{GS} ↓	FPS ↑	Time (m) ↓	SSIM ↑	PSNR ↑	LPIPS ↓	N_{GS} ↓	FPS ↑	Time (m) ↓	SSIM ↑	PSNR ↑	LPIPS ↓	N_{GS} ↓	FPS ↑
3DGS	196	0.735	23.06	0.292	10.76	< 50	180	0.763	21.69	0.252	6.12	< 50	176	0.736	23.66	0.268	6.58	< 50
PGSR	212	0.603	20.12	0.447	12.41	< 50	177	0.780	20.01	0.285	5.87	< 50	187	0.593	20.57	0.421	5.98	< 50
Mip-Splatting	187	0.703	22.35	0.334	18.44	< 50	192	0.775	21.37	0.266	13.07	< 50	184	0.660	21.40	0.341	14.50	< 50
Taming-3DGS	21	0.551	21.22	0.510	0.75	88	23	0.633	19.74	0.460	0.48	73	29	0.714	23.77	0.308	5.92	84
Speedy-Splat	45	0.522	19.70	0.547	0.54	204	54	0.658	19.52	0.432	0.56	181	45	0.646	22.42	0.401	0.73	185
FastGS	12	0.662	22.32	0.391	1.60	251	11	0.708	20.24	0.351	1.28	228	13	0.707	23.47	0.329	2.34	236
EffGS	20	0.755	23.68	0.279	3.76	145	22	0.778	21.50	0.281	3.05	148	21	0.756	24.23	0.273	4.23	124

where $\lambda_{\text{dssim}} = 0.2$ and $\lambda_{\text{freq}} = 0.1$. Let $\mathcal{G}$ denote the current primitive index set and $N = |\mathcal{G}|$. The scale-factor regularizer is

$$\mathcal{R}_\gamma = \begin{cases} \dfrac{\lambda_\gamma}{N} \displaystyle\sum_{i \in \mathcal{G}} \gamma_i^2, & N > 0, \\ 0, & N = 0, \end{cases} \qquad \lambda_\gamma > 0.$$

The mean reduction prevents its scale from increasing solely with the primitive count. The regularizer introduces a shrinkage preference on the modulation factors, but is not a direct penalty on effective Gaussian volume.

4 Experiment

We first describe the datasets and implementation details in Sec. 4.1. We then evaluate EffGS on both bounded and city-scale scenes and compare it with representative Gaussian Splatting methods in Sec. 4.2. Finally, we analyze the contribution of each component through ablation studies in Sec. 4.3.

4.1 Experimental Setup

Datasets. To comprehensively evaluate the rendering fidelity, computational efficiency, and scalability of our proposed method, we conduct experiments across a diverse range of scene configurations. For object-centric and bounded indoor/outdoor environments, we utilize the Mip-NeRF 360 (Barron et al., 2022), Deep-Blending (Park et al., 2019), and Tanks and Temples (Knapitsch et al., 2017) datasets. Furthermore, to rigorously assess the performance and robustness of our method in large-scale, complex, and unbounded scenarios, we evaluate it on the Mill-19 (Turki et al., 2022b), UrbanScene3D (Lin et al., 2022), and GauU-Scene (Xiong et al., 2024) datasets.

Implementation Details. EffGS and all baselines are implemented in PyTorch. EffGS, Speedy-Splat (Hanson et al., 2025a), and FastGS (Ren et al., 2025) are evaluated on identical 24GB GPUs, while baseline configurations that exceed this memory capacity on large-scale scenes are evaluated using NVIDIA A800 GPUs. Additional implementation details and extended experiments are provided in the supplementary material (see Sec. A.1).

4.2 Results Analysis

In this section, we first compare EffGS with representative Gaussian Splatting methods on bounded scenes, and then evaluate its scalability against efficient large-scene reconstruction methods on city-scale datasets. In the supplementary material, we further extend EffGS to multi-GPU training and compare it with several city-scale multi-GPU reconstruction methods, together with a detailed analysis of GPU memory consumption. See Sec. A for more details.

Performance on Bounded Scenes. EffGS achieves a favorable balance between rendering quality and training efficiency (Tab. 1). Among the methods compared in this table, it achieves the highest PSNR on Mip-NeRF 360 and Tanks & Temples, reaching 27.77 and 24.22, respectively. Across all three benchmarks, EffGS improves PSNR, SSIM, and LPIPS over vanilla 3DGS (Kerbl et al., 2023) while substantially reducing training time. Compared with FastGS, it achieves higher SSIM and lower LPIPS on all three datasets, with additional training cost. These results demonstrate the quality–efficiency benefits of the complete framework.

Table 3: Component ablations on GauU-Scene (Xiong et al., 2024), evaluating reconstruction quality and computational cost.

Ablation Item	Rendering Quality			Efficiency Metrics			
	SSIM ↑	PSNR ↑	LPIPS ↓	Time (Min) ↓	Size (GB) ↓	GS (M) ↓	Mem (G) ↓
(a) w/o Frequency-aware Mask	0.752	24.12	0.278	22	1.15	4.51	15.6
(b) w/o Error-aware Mask	0.748	23.98	0.284	21	1.23	4.42	14.8
(c) w/o Learnable Compactness (γ_i)	0.745	23.85	0.288	28	1.25	4.98	16.5
(d) w/o Local Densification and Pruning (LDP)	0.741	23.75	0.295	**19**	**0.98**	**3.85**	**14.2**
(e) w/o Frequency-aware Loss ($\mathcal{L}_{freq}$)	0.753	24.16	0.276	**19**	1.05	4.20	15.3
Full (Ours)	**0.756**	**24.23**	**0.273**	21	1.07	4.23	14.8

Scalability to City-Scale Scenes. As shown in Tab. 2, EffGS outperforms Taming-3DGS, Speedy-Splat, and FastGS in PSNR, SSIM, and LPIPS across all three city-scale datasets. Among the methods in this table, it achieves the best PSNR and SSIM on Mill-19 and GauU-Scene, while maintaining competitive quality on UrbanScene3D. With training times of 20–22 minutes and rendering speeds of 124–148 FPS, EffGS combines efficient training with real-time rendering in large-scale scenes.

4.3 Ablation Study

We evaluate the contribution of each component on the GauU-Scene dataset (Tab. 3). Additional measurements of the frequency extraction module's memory usage are provided in Sec. A.3. We further conduct matched-Gaussian-budget comparisons on both bounded and large-scale scenes to control for differences in primitive count (Tabs. 13 and 14).

Frequency and Error-Aware Guidance. Removing the frequency-aware mask (row a), the error-aware mask (row b), or the frequency-guided loss $\mathcal{L}_{\text{freq}}$ (row e) reduces PSNR and SSIM and increases LPIPS. These results support combining reconstruction errors with scale-dependent spatial guidance for density control and reconstruction supervision.

Adaptive Learnable Compactness (γ_i). Removing the learnable scale-modulation component (row c) increases the Gaussian count to 4.98M, training time to 28 minutes, and peak memory usage to 16.5GB, while reducing PSNR to 23.85. These results support the contribution of scale reparameterization and its interaction with density control to both reconstruction quality and efficiency. The extended evaluation in Tab. 15 further confirms this trend across representative bounded and large-scale scenes, where learnable compactness consistently reduces frequency-specific reconstruction errors while improving rendering quality with fewer Gaussian primitives.

Local Densification and Pruning (LDP). Removing LDP (row d) reduces the Gaussian count, training time, and memory usage, but yields the lowest reconstruction quality among the evaluated variants. Retaining LDP improves all three quality metrics by restricting density modifications to primitives with valid projected footprints in the sampled views. The results support this local restriction as a useful component of the framework's quality–efficiency trade-off. As shown in Tab. 11, adding LDP to FastGS consistently improves reconstruction quality on the Russian scene, increasing SSIM from 0.741 to 0.758 and PSNR from 23.48 to 23.89, while reducing LPIPS from 0.294 to 0.257.

5 Conclusion

We presented EffGS, a framework for efficient, high-fidelity Gaussian splatting across bounded and city-scale scenes. Frequency-aware scoring combines reconstruction errors with scale-dependent spatial masks, while localized density control restricts densification and pruning to primitives with valid projected footprints in sampled views. Learnable per-Gaussian scale modulation provides an additional optimization pathway for adapting effective spatial support. Together, these components achieve a favorable balance between reconstruction quality, training time, and primitive count. Component ablations and matched-primitive-budget comparisons support the effectiveness of the framework, with spectral magnitude analysis complementing standard rendering metrics.

AI USE STATEMENT

Generative AI tools were used only for language polishing, grammar correction, and improving the clarity of the manuscript. They were not used to generate experimental results, perform data analysis, or make scientific claims. All technical content, experiments, and conclusions were produced and verified by the authors.

ETHICS STATEMENT

This work focuses on efficient 3D scene reconstruction and does not involve human subjects, personal data, or sensitive attributes. All experiments are conducted on publicly available research datasets following their intended academic use. We are not aware of any direct ethical concerns specific to the proposed method.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the model architecture, optimization objectives, training schedule, hyperparameters, datasets, and evaluation protocols in the main paper and supplementary material. All experiments use consistent settings unless otherwise specified. We plan to release the implementation and configuration files to facilitate reproduction of the reported results.

REFERENCES

Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 5460–5469. IEEE, 2022.

Samuel Rota Bulò, Lorenzo Porzi, and Peter Kontschieder. Revising densification in gaussian splatting. In Ales Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part LXIII*, Lecture Notes in Computer Science, pp. 347–362. Springer, 2024.

Danpeng Chen, Hai Li, Weicai Ye, Yifan Wang, Weijian Xie, Shangjin Zhai, Nan Wang, Haomin Liu, Hujun Bao, and Guofeng Zhang. PGSR: planar-based gaussian splatting for efficient and high-fidelity surface reconstruction. *CoRR*, abs/2406.06521, 2024.

Youyu Chen, Junjun Jiang, Kui Jiang, Xiao Tang, Zhihao Li, Xianming Liu, and Yinyu Nie. Dashgaussian: Optimizing 3d gaussian splatting in 200 seconds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025*, pp. 11146–11155. Computer Vision Foundation / IEEE, 2025.

Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, and Zhangyang Wang. Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ FPS. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

Guangchi Fang and Bing Wang. Mini-splatting: Representing scenes with a constrained number of gaussians. In Ales Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part LXXVII*, Lecture Notes in Computer Science, pp. 165–181. Springer, 2024.

Yuanyuan Gao, Hao Li, Jiaqi Chen, Zhengyu Zou, Zhihang Zhong, Dingwen Zhang, Xiao Sun, and Junwei Han. Citygs-x: A scalable architecture for efficient and geometrically accurate large-scale scene reconstruction. *CoRR*, abs/2503.23044, 2025.

Abdullah Hamdi, Luke Melas-Kyriazi, Jinjie Mai, Guocheng Qian, Ruoshi Liu, Carl Vondrick, Bernard Ghanem, and Andrea Vedaldi. GES: generalized exponential splatting for efficient radiance field rendering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*. IEEE, 2024.

Alex Hanson, Allen Tu, Geng Lin, Vasu Singla, Matthias Zwicker, and Tom Goldstein. Speedy-splat: Fast 3d gaussian splatting with sparse pixels and sparse primitives. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025*, pp. 21537–21546. Computer Vision Foundation / IEEE, 2025a.

Alex Hanson, Allen Tu, Vasu Singla, Mayuka Jayawardhana, Matthias Zwicker, and Tom Goldstein. PUP 3d-gs: Principled uncertainty pruning for 3d gaussian splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025*, pp. 5949–5958. Computer Vision Foundation / IEEE, 2025b.

Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel J. Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Trans. Graph.*, 37(6):257, 2018.

Lukas Höllein, Aljaz Bozic, Michael Zollhöfer, and Matthias Nießner. 3dgs-lm: Faster gaussian-splatting optimization with levenberg-marquardt. *CoRR*, abs/2409.12892, 2024.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139:1–139:14, 2023.

Sieun Kim, Kyungjin Lee, and Youngki Lee. Color-cued efficient densification method for 3d gaussian splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024 - Workshops, Seattle, WA, USA, June 17-18, 2024*, pp. 775–783. IEEE, 2024.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: benchmarking large-scale scene reconstruction. *ACM Trans. Graph.*, 36(4):78:1–78:13, 2017.

Changbai Li, Haodong Zhu, Hanlin Chen, Juan Zhang, Tongfei Chen, Shuo Yang, Shuwei Shao, Wenhao Dong, and Baochang Zhang. HRGS: hierarchical gaussian splatting for memory-efficient high-resolution 3d reconstruction. *CoRR*, abs/2506.14229, 2025.

Changbai Li, Haodong Zhu, Hanlin Chen, Xiuping Liang, Tongfei Chen, Shuwei Shao, Linlin Yang, Huobin Tan, and Baochang Zhang. Urbangs: A scalable and efficient architecture for geometrically accurate large-scene reconstruction. *CoRR*, abs/2602.02089, 2026.

Jiaqi Lin, Zhihao Li, Xiao Tang, Jianzhuang Liu, Shiyong Liu, Jiayue Liu, Yangdi Lu, Xiaofei Wu, Songcen Xu, Youliang Yan, and Wenming Yang. Vastgaussian: Vast 3d gaussians for large scene reconstruction. In *CVPR*, 2024.

Liqiang Lin, Yilin Liu, Yue Hu, Xingguang Yan, Ke Xie, and Hui Huang. Capturing, reconstructing, and simulating: The urbanscene3d dataset. In Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part VIII*, volume 13668 of *Lecture Notes in Computer Science*, pp. 93–109. Springer, 2022.

Yang Liu, Chuanchen Luo, Lue Fan, Naiyan Wang, Junran Peng, and Zhaoxiang Zhang. Citygaussian: Real-time high-quality large-scale scene rendering with gaussians. In *Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part XVI*, pp. 265–282, 2024a.

Yang Liu, Chuanchen Luo, Zhongkai Mao, Junran Peng, and Zhaoxiang Zhang. Citygaussianv2: Efficient and geometrically accurate reconstruction for large-scale scenes. *CoRR*, abs/2411.00771, 2024b.

Saswat Subhajyoti Mallick, Rahul Goel, Bernhard Kerbl, Markus Steinberger, Francisco Vicente Carrasco, and Fernando De la Torre. Taming 3dgs: High-quality radiance fields with limited resources. In Takeo Igarashi, Ariel Shamir, and Hao (Richard) Zhang (eds.), *SIGGRAPH Asia 2024 Conference Papers, SA 2024, Tokyo, Japan, December 3-6, 2024*, pp. 2:1–2:11. ACM, 2024.

Zhenxing Mi and Dan Xu. Switch-nerf: Learning scene decomposition with mixture of experts for large-scale neural radiance fields. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part I*, 2020.

Jeong Joon Park, Peter R. Florence, Julian Straub, Richard A. Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, June 16-20, 2019*, pp. 165–174. Computer Vision Foundation / IEEE, 2019.

Christian Reiser, Richard Szeliski, Dor Verbin, Pratul P. Srinivasan, Ben Mildenhall, Andreas Geiger, Jonathan T. Barron, and Peter Hedman. MERF: memory-efficient radiance fields for real-time view synthesis in unbounded scenes. *ACM Trans. Graph.*, 42(4):89:1–89:12, 2023.

Shiwei Ren, Tianci Wen, Yongchun Fang, and Biao Lu. Fastgs: Training 3d gaussian splatting in 100 seconds. *CoRR*, abs/2511.04283, 2025.

Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-nerf: Scalable construction of large-scale nerfs for virtual fly- throughs. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 12912–12921. IEEE, 2022a.

Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-nerf: Scalable construction of large-scale nerfs for virtual fly- throughs. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 12912–12921. IEEE, 2022b.

Butian Xiong, Nanjun Zheng, Junhua Liu, and Zhen Li. Gauu-scene V2: assessing the reliability of image-based metrics with expansive lidar image dataset using 3dgs and nerf. *CoRR*, abs/2404.04880, 2024.

Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pp. 19447–19456. IEEE, 2024a.

Zehao Yu, Torsten Sattler, and Andreas Geiger. Gaussian opacity fields: Efficient adaptive surface reconstruction in unbounded scenes. *ACM Trans. Graph.*, 43(6):271:1–271:13, 2024b.

Haodong Zhu, Changbai Li, Yangyang Ren, Zichao Feng, Xuhui Liu, Hanlin Chen, Xiantong Zhen, and Baochang Zhang. Surf3r: Rapid surface reconstruction from sparse RGB views in seconds. *CoRR*, abs/2508.04508, 2025.

Supplementary Material

This supplementary material is organized as follows:

- Appendix A provides experimental settings, extended quantitative and qualitative comparisons, and analyses of frequency-extraction memory usage and frequency-aware reconstruction errors.
- Appendix B presents additional baseline comparisons, matched-Gaussian-budget evaluations, and an extended ablation of adaptive primitive compactness.
- Appendix C details learnable per-Gaussian scale modulation and frequency-aware mask generation.
- Appendix D discusses the limitations of the proposed method.
- Appendix E discusses broader social impacts and potential applications.

A More Experiments

To further demonstrate the scalability of our EffGS, we extend it to multi-GPU parallel training. We adopt the tiling strategy from CityGS (Liu et al., 2024a) for parallelization and conduct an extended comparison with state-of-the-art city-scale 3DGS methods (Gao et al., 2025; Liu et al., 2024b;a) on the Mill-19 (Turki et al., 2022b), Urbanscene3D (Lin et al., 2022), and GauU-Scene (Xiong et al., 2024) datasets, as illustrated in Fig. 12. We provide detailed quantitative comparisons in Tabs. 5, 9, 10, and 12. We also report model size and GPU memory usage, as visualized in Fig. 9. In addition, we perform two supplementary ablation studies: Memory Footprint of Frequency Extraction (Sec. A.3) and Frequency Aware Error Analysis (Sec. A.4).

We also provide additional qualitative (see Fig. 13) and quantitative comparisons (see Fig. 14, Tab. 5,Tab. 4,Tab. 6) on the Deep Blending (Park et al., 2019), Mip-NeRF 360 (Barron et al., 2022), and Tanks & Temples (Knapitsch et al., 2017) datasets.

A.1 Experimental Setup and Implementation

EffGS and the comparative baselines are implemented in PyTorch. Shared optimization settings, including the learning rates for Gaussian positions and base scales, opacity reset intervals, and density-control frequency, follow the original 3DGS configuration. The density-control rules and additional optimization components of EffGS are described in Secs. 3.2–3.5.

For bounded indoor and outdoor benchmarks, including Mip-NeRF 360, Deep Blending, and Tanks & Temples, training and evaluation are conducted on a single NVIDIA GeForce RTX 4090 with 24GB VRAM. EffGS, Speedy-Splat (Hanson et al., 2025a), and FastGS (Ren et al., 2025) are evaluated under the same hardware setting.

For large-scale urban benchmarks, we maintain the same 24GB memory constraint whenever possible. However, several baseline configurations exceed this capacity. Specifically, 3DGS (Kerbl et al., 2023), PGSR (Chen et al., 2024), and Mip-Splatting (Yu et al., 2024a) encounter out-of-memory issues, with Mip-Splatting requiring more than 60GB of GPU memory in some scenes, and are therefore evaluated using NVIDIA A800 GPUs. Taming-3DGS similarly requires the A800 on several GauU-Scene scenes.

We further distinguish the single-GPU EffGS configuration from its distributed extension, denoted as EffGS-GPUs. The distributed experiments use NVIDIA TITAN RTX GPUs and adopt the tiling strategy described in Sec. A. Additional memory measurements for frequency extraction and frequency-specific error analysis are provided in Secs. A.3 and A.4, respectively.

For the initial Gaussian primitives, we initialize $\beta_i = 0$, corresponding to a scale-modulation factor $\gamma_i = 1$. Primitives created through cloning or splitting inherit their parent's β_i. The modulation parameters are optimized jointly with the other Gaussian attributes. The parameterization, regularization, and relation to projected support are detailed in Sec. C.1.

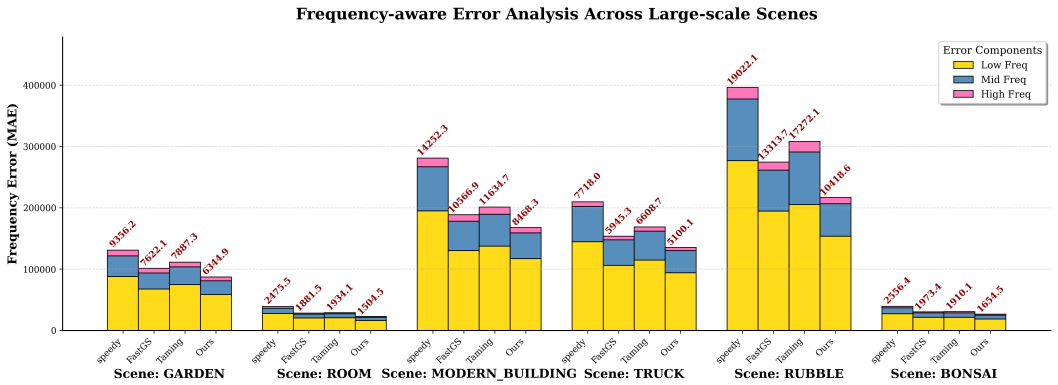


Figure 7: Low-, mid-, and high-frequency spectral magnitude errors across scenes.

A.2 Extended Results Analysis

Performance on Massive Urban Scenes. The extended comparisons (Tabs. 8–10) show that EffGS-GPUs achieves competitive reconstruction quality against specialized large-scale pipelines, including CityGaussian, CityGS-v2, and CityGS-X, with leading results across multiple scenes and metrics. Table 12 reports the training time, primitive count, and rendering speed of EffGS and the listed baselines. Fig. 9 additionally compares memory usage and model size, with multi-GPU configurations marked separately.

Fig. 8 tracks the band-wise spectral magnitude errors on the Rubble scene. The lower errors observed for EffGS across the three bands complement the standard image-quality metrics and provide additional evidence of improved spectral magnitude reconstruction.

We further evaluate LDP by incorporating it into FastGS (Tab. 11). On the Russian scene, FastGS+LDP improves PSNR from 23.48 to 23.89 and reduces LPIPS from 0.294 to 0.257. This comparison supports the benefit of restricting density operations to the sampled-view active set within the FastGS pipeline.

Performance on Bounded Scenes. The per-scene results in Tabs. 4, 5, and 6 provide a more detailed view of reconstruction quality across bounded scenes. Together with the training-time comparison in Tab. 1, they support the effectiveness of EffGS across indoor and outdoor environments. The qualitative examples in Fig. 13 show clearer edges and finer texture details than FastGS and Speedy-Splat in the illustrated regions, complementing the quantitative comparisons.

A.3 Memory Footprint of Frequency Extraction

The frequency-aware mask is computed from two Gaussian-filtered versions of a downsampled grayscale image. The absolute difference is upsampled, normalized, and thresholded according to the two-stage schedule in Sec. C.2.

Fig. 11 evaluates memory usage during sequential processing with varying input resolutions and a fixed batch size of 10 on the Russian scene. After the initial increase, peak VRAM usage fluctuates within a bounded range without a sustained upward trend over the tested sequence. The step-wise changes reflect these fluctuations and support the memory stability of frequency extraction under the evaluated settings.

A.4 Frequency-Aware Error Analysis

We visualize the spatial masks used for training guidance and measure spectral magnitude discrepancies in separate frequency bands. These analyses complement PSNR, SSIM, LPIPS, and qualitative renderings by characterizing mask selection and frequency-specific differences.

Table 4: **Quantitative PSNR comparison on Mip-NeRF 360 (Barron et al., 2022).** The best, second-best, and third-best results are indicated by light red, light orange, and light yellow backgrounds, respectively.

Method	Bicycle	Flowers	Garden	Stump	Treehill	Room	Counter	Kitchen	Bonsai
3DGS	25.14	21.30	27.34	26.64	22.59	31.71	29.16	31.54	32.37
Mini-Splatting	25.23	21.43	27.36	26.80	22.76	31.48	28.65	31.05	31.24
Speedy-Splat	24.79	21.21	26.69	26.67	22.48	30.83	28.22	30.09	31.16
Taming-3DGS	24.72	21.10	27.42	26.05	22.92	31.64	29.20	31.84	32.40
DashGaussian	25.31	21.78	27.57	27.17	22.94	31.81	29.11	31.69	32.15
FastGS	24.84	21.21	27.20	26.65	22.94	31.98	29.15	31.87	32.19
Ours	25.25	22.18	27.45	27.21	22.34	32.03	29.15	31.91	32.39

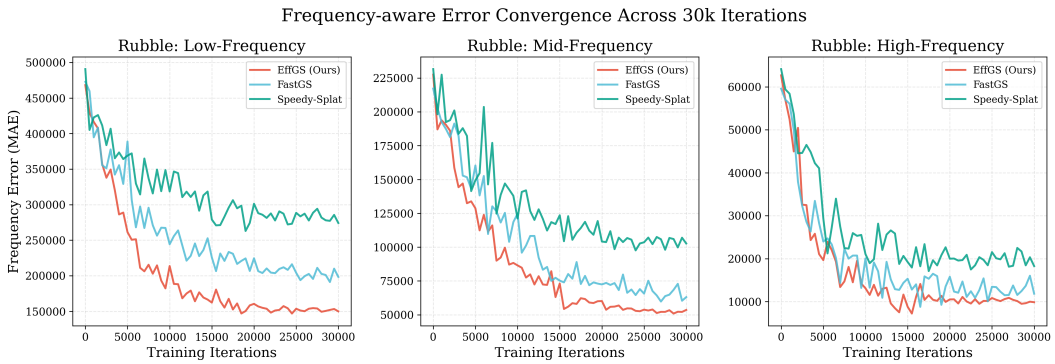


Figure 8: Band-wise spectral magnitude errors over 30,000 training iterations on Rubble.

A.4.1 Frequency-Aware Mask Visualization

Fig. 10 shows continuous selection scores, their overlays, and binary masks on a fixed image at different schedule parameters f. For $f \geq 50$, increasing f reduces the Gaussian filter scales, producing responses around progressively finer structures. For $f < 50$, the normalized response is complemented before thresholding. The heatmaps represent spatial selection scores, and the binary masks identify pixels selected at the threshold of 0.5. Because scores are normalized independently for each setting, their colors indicate relative selection strength within each image.

A.4.2 Spectral Decomposition and Error Formulation

We compute the centered Fourier spectra of the grayscale ground-truth and rendered images. For $I \in \{I_{\text{gt}}, I_{\text{render}}\}$,

$$\widehat{I} = \text{fftshift}(\mathcal{F}(I)), \qquad A_I = |\widehat{I}|, \tag{14}$$

where $\mathcal{F}$ denotes the 2D discrete Fourier transform, $\widehat{I}$ is the centered complex spectrum, and A_I is its magnitude spectrum.

Let (ξ, η) denote a frequency-grid coordinate and $D(\xi, \eta)$ its Euclidean radial distance from the centered zero-frequency location. We partition the frequency coordinates into three bands:

$$\begin{aligned} \Omega_{\text{low}} &= \{(\xi,\eta) : D(\xi,\eta) \leq 30\}, \\ \Omega_{\text{mid}} &= \{(\xi,\eta) : 30 < D(\xi,\eta) \leq 80\}, \\ \Omega_{\text{high}} &= \{(\xi,\eta) : D(\xi,\eta) > 80\}. \end{aligned} \tag{15}$$

The thresholds are measured in frequency-grid pixels and therefore define bands relative to the image resolution used in each evaluation. For each band $b \in \{\text{low}, \text{mid}, \text{high}\}$, we compute the mean absolute spectral magnitude error:

$$E_b = \frac{1}{|\Omega_b|} \sum_{(\xi,\eta)\in\Omega_b} \left| A_{I_{\text{render}}}(\xi,\eta) - A_{I_{\text{gt}}}(\xi,\eta) \right|. \tag{16}$$

Table 5: **Quantitative SSIM comparison on Mip-NeRF 360 (Barron et al., 2022).** The best, second-best, and third-best results are indicated by light red, light orange, and light yellow backgrounds, respectively.

Method	Bicycle	Flowers	Garden	Stump	Treehill	Room	Counter	Kitchen	Bonsai
3DGS	0.748	0.586	0.857	0.768	0.636	0.927	0.915	0.932	0.946
Mini-Splatting	0.764	0.614	0.806	0.839	0.656	0.928	0.911	0.930	0.943
Speedy-Splat	0.704	0.560	0.814	0.765	0.590	0.903	0.876	0.895	0.925
Taming-3DGS	0.693	0.552	0.851	0.729	0.628	0.917	0.909	0.929	0.942
DashGaussian	0.763	0.604	0.857	0.783	0.640	0.924	0.911	0.927	0.945
FastGS	0.714	0.560	0.836	0.756	0.612	0.920	0.907	0.929	0.942
Ours	0.773	0.610	0.861	0.825	0.642	0.931	0.913	0.934	0.944

Table 6: **Quantitative LPIPS comparison on Mip-NeRF 360 (Barron et al., 2022).** The best, second-best, and third-best results are indicated by light red, light orange, and light yellow backgrounds, respectively.

Method	Bicycle	Flowers	Garden	Stump	Treehill	Room	Counter	Kitchen	Bonsai
3DGS	0.242	0.360	0.122	0.244	0.347	0.197	0.183	0.116	0.180
Mini-Splatting	0.241	0.341	0.215	0.161	0.326	0.190	0.181	0.120	0.177
Speedy-Splat	0.333	0.418	0.214	0.288	0.462	0.258	0.259	0.195	0.228
Taming-3DGS	0.332	0.416	0.138	0.324	0.395	0.227	0.200	0.128	0.193
DashGaussian	0.222	0.341	0.131	0.229	0.333	0.205	0.191	0.129	0.180
FastGS	0.310	0.406	0.174	0.297	0.429	0.217	0.204	0.127	0.191
Ours	0.240	0.353	0.162	0.243	0.335	0.104	0.171	0.106	0.151

This metric measures differences in Fourier magnitude. Since it does not capture phase differences, we interpret it together with the standard image-quality metrics and qualitative results.

A.4.3 Spatial Visualization and Quantitative Comparison

To visualize high-frequency response discrepancies, we define the binary spectral mask

$$M_{\text{high}}(\xi, \eta) = \mathbb{I}[(\xi, \eta) \in \Omega_{\text{high}}], \tag{17}$$

where $\mathbb{I}[\cdot]$ denotes the indicator function. We apply this mask to the complex spectrum and compute the magnitude of the reconstructed high-pass response:

$$S_{\text{hf}}(I) = \left| \mathcal{F}^{-1}\left(\text{ifftshift}\left(\widehat{I} \odot M_{\text{high}} \right) \right) \right|, \tag{18}$$

where $\mathcal{F}^{-1}$ denotes the inverse 2D discrete Fourier transform and $\odot$ denotes element-wise multiplication.

We visualize

$$\Delta S_{\text{hf}} = |S_{\text{hf}}(I_{\text{gt}}) - S_{\text{hf}}(I_{\text{render}})| \tag{19}$$

Table 7: **Quantitative results (Time) on Mip-NeRF 360 (Barron et al., 2022).**

Method	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai
3DGS	27.97	18.98	26.78	21.77	20.33	18.78	17.58	21.30	14.90
Mini-Splatting	16.17	17.22	15.97	16.52	17.05	18.00	9.83	10.23	11.02
Speedy-Splat	15.87	13.38	15.73	13.77	12.90	12.05	12.10	13.25	11.37
Taming-3DGS	5.65	4.97	9.82	3.93	5.37	3.88	4.60	3.48	4.60
DashGaussian	9.93	7.05	8.27	6.57	8.20	4.00	3.95	5.52	3.95
FastGS	1.92	1.95	2.47	1.72	1.72	1.62	1.83	2.42	1.83
Ours	3.14	2.45	3.23	4.15	2.56	2.52	2.93	3.22	3.43

Table 8: Quantitative SSIM comparison (↑) across different datasets. Building and Rubble are from Mill-19 (Turki et al., 2022b); Residence and Sci-Art are from UrbanScene3D (Lin et al., 2022); Russian Building, Residence+, and Modern Building are from GauU-Scene (Xiong et al., 2024). The best, second-best, and third-best results are indicated by light red, light orange, and light yellow backgrounds, respectively. † denotes results obtained without decoupled appearance encoding, while * denotes results obtained without depth supervision.

Method	Mill-19		UrbanScene3D		GauU-Scene		
	Building	Rubble	Residence	Sci-Art	Russian Building	Residence+	Modern Building
VastGaussian†	0.725	0.745	0.712	0.765	0.781	0.738	0.789
CityGaussian	0.776	0.814	0.810	0.835	0.801	0.755	0.791
CityGS-v2	0.661	0.724	0.771	0.808	0.792	0.741	0.762
CityGS-X*	0.771	0.803	0.802	0.829	0.793	0.742	0.774
3DGS	0.722	0.748	0.782	0.743	0.770	0.686	0.751
PGSR	0.482	0.723	0.758	0.802	0.759	0.315	0.705
Mip-Splatting	0.683	0.722	0.749	0.806	0.751	0.518	0.710
Taming-3DGS	0.498	0.603	0.560	0.706	0.738	0.661	0.744
Speedy-Splat	0.462	0.581	0.649	0.667	0.697	0.569	0.671
FastGS	0.654	0.669	0.688	0.728	0.741	0.659	0.722
EffGS	0.728	0.783	0.777	0.778	0.768	0.747	0.754
EffGS-GPUs	0.782	0.811	0.813	0.844	0.803	0.786	0.795

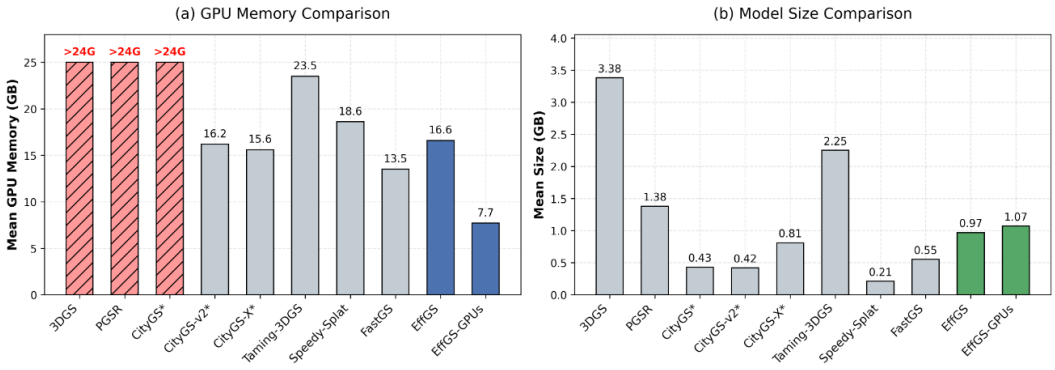


Figure 9: Quantitative comparison of computational overhead on the Mill-19 (Turki et al., 2022b), Urbanscene3D (Lin et al., 2022), and GauU-Scene datasets (Xiong et al., 2024). * indicates multi-GPU training.

after normalization by its 99th percentile. The resulting map shows spatial differences in high-pass response magnitude, rather than squared-energy error. Percentile normalization emphasizes the spatial distribution of these discrepancies.

We also present the three band-wise errors as stacked bars, allowing comparisons between methods within each frequency band. The stacked height is the sum of the three band-wise mean errors; it is neither the overall pixel-space error nor a decomposition of total error into frequency-band proportions.

B Additional Comparisons and Controlled Experiments

We compare EffGS with EDGS and 3DGS-MCMC under reported and matched-budget settings, and further evaluate adaptive primitive compactness.

B.1 Reference and Matched-Budget Comparisons

Tab. 13 compares reconstruction quality and training time on Deep Blending and Tanks & Temples. Under reported settings, EffGS has lower reported training times on both datasets and achieves the

Table 9: Quantitative PSNR comparison (↑) across different datasets. Building and Rubble are from Mill-19 (Turki et al., 2022b); Residence and Sci-Art are from UrbanScene3D (Lin et al., 2022); Russian Building, Residence+, and Modern Building are from GauU-Scene (Xiong et al., 2024). The best, second-best, and third-best results are indicated by light red, light orange, and light yellow backgrounds, respectively. † denotes results obtained without decoupled appearance encoding, while * denotes results obtained without depth supervision.

Method	Mill-19		UrbanScene3D		GauU-Scene		
	Building	Rubble	Residence	Sci-Art	Russian Building	Residence+	Modern Building
VastGaussian†	21.83	25.24	21.06	22.59	23.98	23.41	25.53
CityGaussian	21.56	25.79	22.03	22.42	24.11	23.65	26.01
CityGS-v2	19.85	24.02	21.23	20.71	24.04	23.43	25.78
CityGS-X*	21.78	25.45	22.11	22.32	24.18	23.91	25.95
3DGS	20.62	25.49	21.45	21.92	23.74	22.10	25.15
PGSR	17.01	23.23	20.58	19.43	23.30	14.65	23.77
Mip-Splatting	20.65	24.05	20.98	21.75	22.48	17.97	23.76
Taming-3DGS	19.13	23.31	19.01	20.47	23.51	22.34	25.46
Speedy-Splat	16.90	22.50	20.04	18.99	22.87	20.32	24.08
FastGS	20.88	23.76	20.13	20.34	23.48	22.12	24.81
EffGS	22.05	25.31	21.23	21.77	24.05	23.28	25.35
EffGS-GPUs	22.34	25.83	21.89	22.73	24.34	23.89	26.14

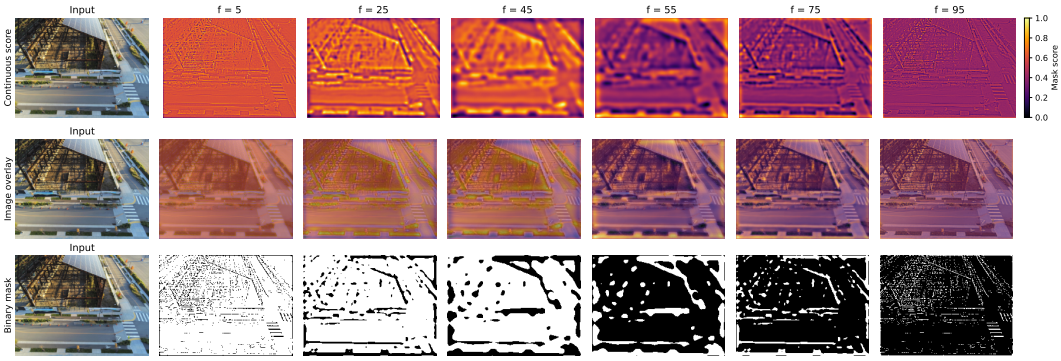


Figure 10: Visualization of the frequency-aware mask at different schedule parameters f on a fixed input image. Top: continuous mask scores before thresholding. Middle: score heatmaps overlaid on the input. Bottom: binary masks obtained with a threshold of 0.5, where white indicates selected pixels. Scores are normalized independently for each setting.

highest SSIM and PSNR on Deep Blending, while EDGS achieves the lowest LPIPS on both datasets and the highest SSIM and PSNR on Tanks & Temples. Under Gaussian budgets matched to EffGS—0.25M on Deep Blending and 0.44M on Tanks & Temples—EffGS achieves the best PSNR, SSIM, LPIPS, and training time on both datasets. These results show that its quality advantages persist under matched budgets rather than relying on more Gaussian primitives.

B.2 Matched-Budget Comparisons on Large-Scale Scenes

We further evaluate matched-budget performance on Mill-19 and UrbanScene3D. For Taming-3DGS, we set its target Gaussian budget to match EffGS. For the EDGS variant, denoted as EDGS+3DGS, we disable densification and initialize optimization with the same Gaussian budget. These budget-constrained configurations are evaluated separately from the methods' original operating settings.

Tab. 14 summarizes the results. At the reported matched budgets of 3.76M Gaussians on Mill-19 and 3.05M Gaussians on UrbanScene3D, EffGS achieves the best reconstruction metrics and the shortest training time among the evaluated methods. On Mill-19, EffGS achieves 23.68 dB PSNR

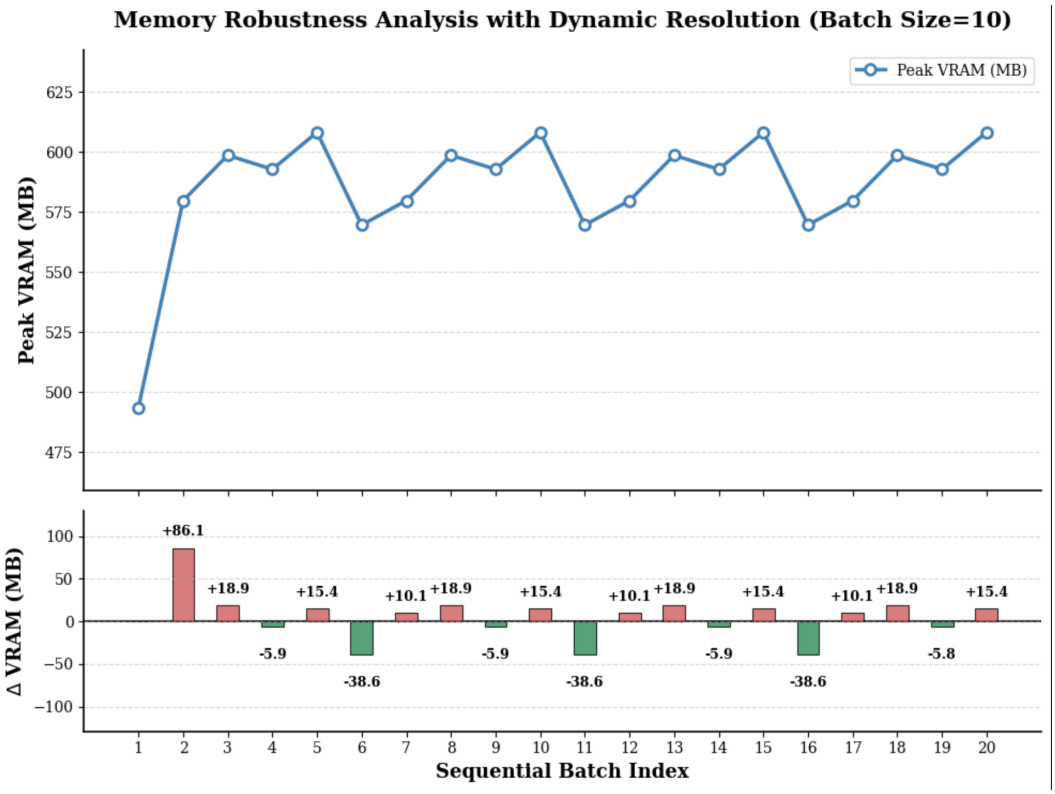


Figure 11: Peak VRAM usage (top) and step-wise changes (bottom) during frequency extraction with varying resolutions and batch size 10 on Russian (Xiong et al., 2024).

Table 10: Quantitative LPIPS comparison (↓) across different datasets. Building and Rubble are from Mill-19 (Turki et al., 2022b); Residence and Sci-Art are from UrbanScene3D (Lin et al., 2022); Russian Building, Residence+, and Modern Building are from GauU-Scene (Xiong et al., 2024). The best, second-best, and third-best results are indicated by light red, light orange, and light yellow backgrounds, respectively. † denotes results obtained without decoupled appearance encoding, while * denotes results obtained without depth supervision.

Method	Mill-19		UrbanScene3D		GauU-Scene		
	Building	Rubble	Residence	Sci-Art	Russian Building	Residence+	Modern Building
VastGaussian†	0.271	0.268	0.263	0.259	0.251	0.301	0.245
CityGaussian	0.267	0.226	0.213	0.232	0.223	0.289	0.253
CityGS-v2	0.382	0.311	0.235	0.265	0.234	0.294	0.254
CityGS-X*	0.256	0.224	0.221	0.241	0.220	0.284	0.234
3DGS	0.304	0.279	0.238	0.265	0.234	0.321	0.249
PGSR	0.541	0.352	0.292	0.277	0.278	0.657	0.327
Mip-Splatting	0.344	0.324	0.272	0.259	0.252	0.482	0.290
Taming-3DGS	0.542	0.477	0.509	0.410	0.291	0.352	0.282
Speedy-Splat	0.593	0.501	0.411	0.452	0.352	0.469	0.382
FastGS	0.380	0.402	0.351	0.351	0.294	0.377	0.315
EffGS	0.276	0.282	0.265	0.297	0.254	0.291	0.273
EffGS-GPUs	0.253	0.217	0.208	0.224	0.236	0.281	0.256

and 0.279 LPIPS, compared with 22.57 dB and 0.372 for EDGS+3DGS. On UrbanScene3D, EffGS achieves 21.50 dB PSNR and 0.281 LPIPS, compared with 21.17 dB and 0.330 for EDGS+3DGS. Together with the results on Deep Blending and Tanks & Temples, these comparisons support the effectiveness of EffGS under matched primitive budgets across different scene scales. They do not, by themselves, isolate the contribution of an individual component of the full framework.

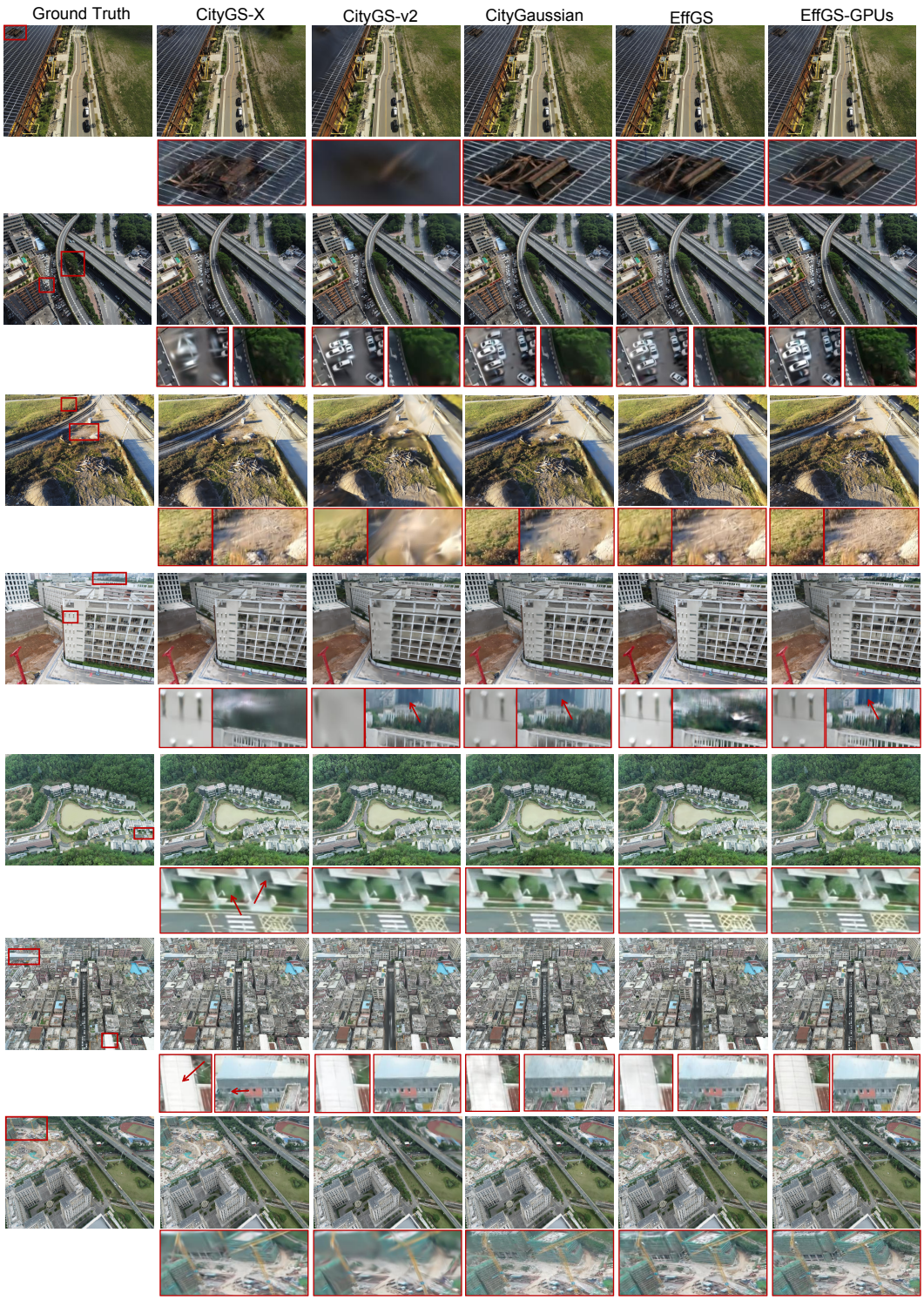


Figure 12: Qualitative comparisons on the Mill-19, Urbanscene3D, and GauU-Scene datasets (Turki et al., 2022b; Lin et al., 2022; Xiong et al., 2024).

B.3 Extended Analysis of Adaptive Primitive Compactness

The learnable factor γ_i provides a dedicated optimization pathway for adaptive primitive compactness. Although a multiplicative scale modulation does not expand the static representational space

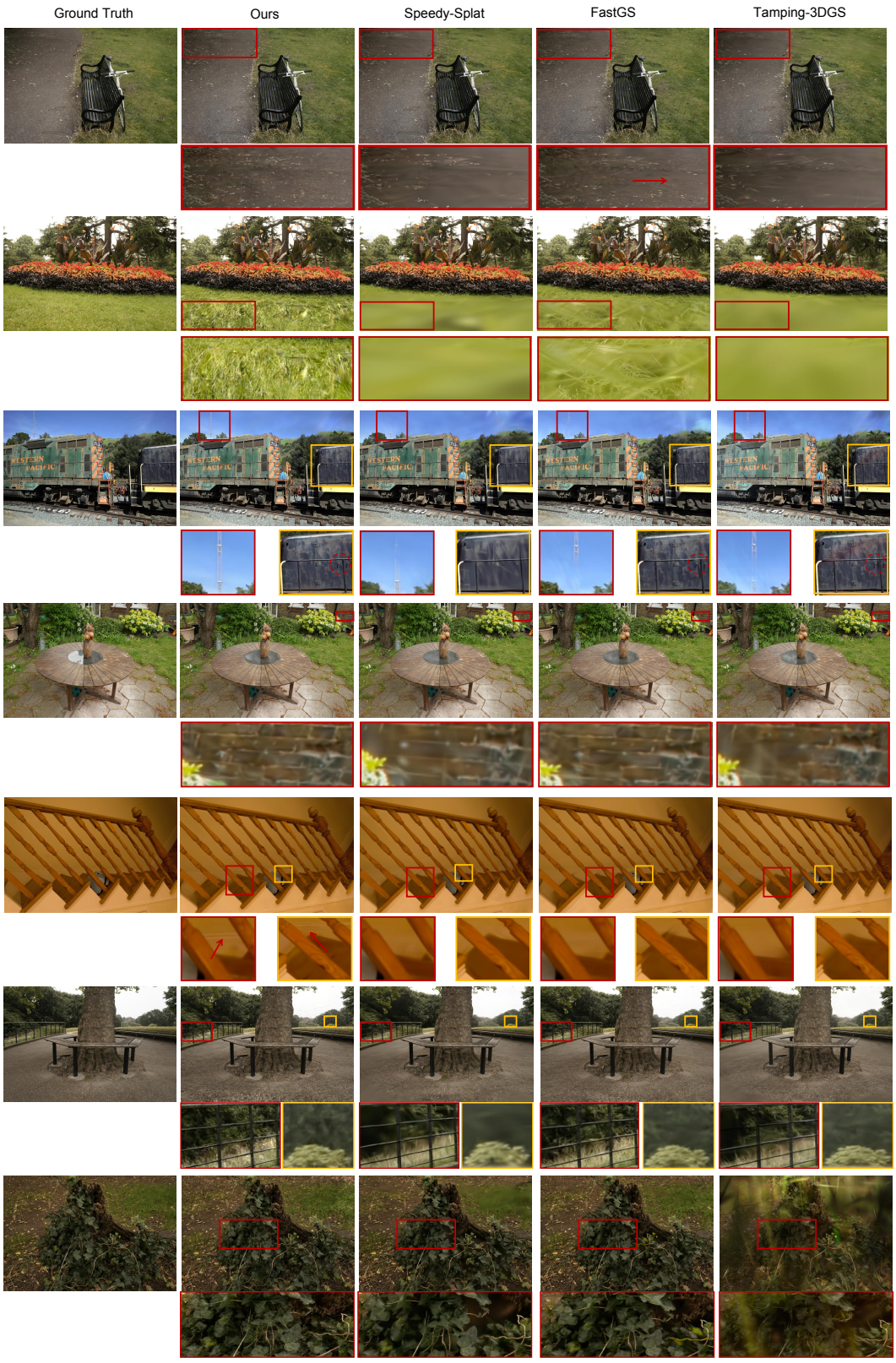


Figure 13: Qualitative comparisons on the Deep Blending, Mip-NeRF 360, and Tanks & Temples datasets (Park et al., 2019; Barron et al., 2022; Knapitsch et al., 2017).

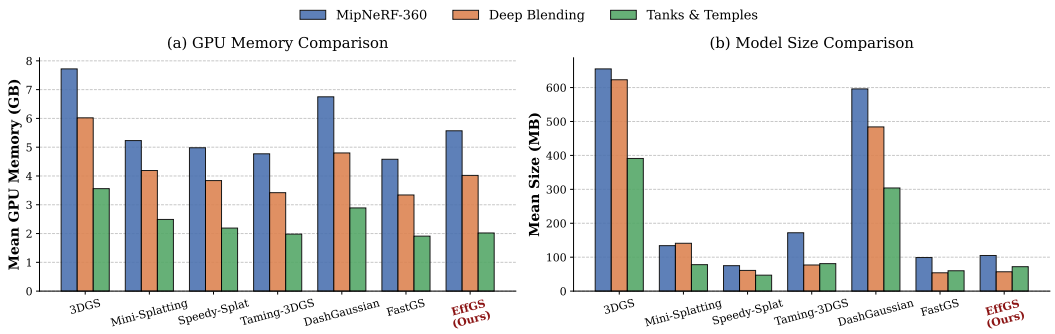


Figure 14: Quantitative comparison of computational overhead on the Deep Blending (Park et al., 2019), Mip-NeRF 360 (Barron et al., 2022), and Tanks & Temples datasets (Knapitsch et al., 2017).

Table 11: Ablation study on the Russian scene from the GauU-Scene dataset (Xiong et al., 2024).

Method	SSIM ↑	PSNR ↑	LPIPS ↓
FastGS	0.741	23.48	0.294
FastGS+LDP	0.758	23.89	0.257

of a Gaussian, it changes the optimization parameterization and participates in compactness-aware primitive control.

We evaluate its contribution on five representative scenes: Garden, Room, and Bonsai from Mip-NeRF 360, Rubble from Mill-19, and Modern Building from GauU-Scene. Tab. 15 reports the aggregated results across these scenes. Compared with the variant without learnable compactness, the full model reduces the average errors in all three frequency bands, improves PSNR from 26.99 to 28.51 dB and SSIM from 0.810 to 0.855, and reduces LPIPS from 0.264 to 0.194. The average Gaussian count also decreases from 2.30M to 2.02M. Thus, the observed improvement in reconstruction quality is accompanied by a more compact representation, rather than an increase in primitive count.

Summary. The matched-budget comparisons show that EffGS achieves better reconstruction quality than the evaluated baselines at the tested Gaussian budgets, while also reporting shorter training times. The extended compactness ablation further shows that the full model improves average reconstruction quality and frequency-specific errors while using fewer primitives. These results provide complementary evidence for the quality–efficiency benefits of the proposed framework.

C Methodological Details

C.1 Learnable Scale Modulation for Primitive Compactness

Building upon the Compact Box mechanism in FastGS (Ren et al., 2025), we introduce a **learnable per-Gaussian scale factor** γ_i. This factor reparameterizes the effective Gaussian scales, providing an additional optimization pathway for adapting primitive extent while retaining the existing tile-culling rule.

Covariance modulation. In vanilla 3DGS (Kerbl et al., 2023), the covariance of Gaussian i is constructed from its base-scale matrix $S_i = \mathrm{diag}(s_i^x, s_i^y, s_i^z)$, with $s_i^x, s_i^y, s_i^z > 0$, and rotation matrix R_i:

$$\Sigma_i = R_i S_i S_i^\top R_i^\top . \tag{20}$$

We augment each Gaussian with a learnable scalar $\beta_i \in \mathbb{R}$, initialized to zero for the initial Gaussian primitives. Its corresponding positive scale multiplier is

$$\gamma_i = 2\ \mathrm{sigmoid}(\beta_i). \tag{21}$$

For finite β_i, we have $\gamma_i \in (0, 2)$, with $\gamma_i = 1$ at initialization.

Table 12: Efficiency comparison on Residence+, Russian Building, and Modern Building scenes from the GauU-Scene dataset (Xiong et al., 2024).

Method	Residence+			Russian Building			Modern Building		
	Time (min)	GS (M)	FPS	Time (min)	GS (M)	FPS	Time (min)	GS (M)	FPS
CityGaussian	261	8.14	68	214	7.02	55	217	7.90	57
CityGS-v2	203	8.04	46	182	6.97	33	188	7.90	35
Speedy-splat	50	0.50	195	43	0.82	189	43	0.87	170
FastGS	14	2.91	252	12	1.89	231	12	2.23	224
EffGS	19	4.86	131	21	3.45	127	23	4.37	115

Table 13: Comparison on Deep Blending and Tanks & Temples. Top: reported settings. Bottom: Gaussian budgets matched to EffGS. Time is in minutes and N_{GS} in millions; – denotes unavailable counts. Best quality and timing results within each block and dataset are bold.

Method	Deep Blending					Tanks & Temples				
	Time ↓	PSNR ↑	SSIM ↑	LPIPS ↓	N_{GS}	Time ↓	PSNR ↑	SSIM ↑	LPIPS ↓	N_{GS}
Uncontrolled Gaussian-point count										
EDGS	30.00	29.81	0.904	**0.223**	–	23.00	**24.28**	**0.868**	**0.132**	–
3DGS-MCMC	19.00	29.56	0.902	0.244	–	13.00	24.22	0.863	0.158	–
EffGS (Ours)	**2.94**	**30.01**	**0.910**	0.238	0.25	**1.94**	24.22	0.861	0.166	0.44
Matched Gaussian budgets: controlled comparison										
EDGS	7.00	28.83	0.877	0.309	0.25	6.00	22.19	0.841	0.194	0.44
3DGS-MCMC	5.00	28.67	0.874	0.294	0.25	4.00	22.24	0.837	0.207	0.44
EffGS (Ours)	**2.94**	**30.01**	**0.910**	**0.238**	0.25	**1.94**	**24.22**	**0.861**	**0.166**	0.44

The effective scale matrix is

$$\widetilde{S}_i = \mathrm{diag}(\gamma_i s_i^x, \gamma_i s_i^y, \gamma_i s_i^z) = \gamma_i S_i. \tag{22}$$

Consequently,

$$\widetilde{\Sigma}_i = R_i \widetilde{S}_i \widetilde{S}_i^\top R_i^\top = \gamma_i^2 \Sigma_i. \tag{23}$$

The scalar factor changes the overall extent, while anisotropic axis ratios remain governed by the base scales.

Let $\widetilde{\Sigma}_i^{2\mathrm{D}}$ denote the projected covariance and $\mathbf{p}_i^{2\mathrm{D}}$ the projected center, following the notation in the main paper. For an image-plane coordinate $\mathbf{u} \in \mathbb{R}^2$, the squared Mahalanobis distance is

$$m_i(\mathbf{u}) = (\mathbf{u} - \mathbf{p}_i^{2\mathrm{D}})^\top (\widetilde{\Sigma}_i^{2\mathrm{D}})^{-1} (\mathbf{u} - \mathbf{p}_i^{2\mathrm{D}}). \tag{24}$$

Compact Box uses this quantity to determine the spatial support relevant to tile culling.

Gradient flow. We optimize β_i jointly with the other Gaussian attributes using Adam. Its gradient follows the chain rule:

$$\frac{\partial \mathcal{L}}{\partial \beta_i} = \frac{\partial \mathcal{L}}{\partial \gamma_i} \frac{\partial \gamma_i}{\partial \beta_i}. \tag{25}$$

From Eq. equation 21,

$$\frac{\partial \gamma_i}{\partial \beta_i} = 2\ \mathrm{sigmoid}(\beta_i)\left(1 - \mathrm{sigmoid}(\beta_i)\right). \tag{26}$$

This differentiable parameterization allows the training objective to update the modulation factor through the effective Gaussian scales.

Impact on projected support. Holding the Gaussian center and camera fixed, let $\Pi_i = J_i W$, where W is the rotational part of the world-to-camera transformation and J_i is the projection Jacobian at the center of Gaussian i. Under the first-order covariance projection,

$$\widetilde{\Sigma}_i^{2\mathrm{D}} = \Pi_i \widetilde{\Sigma}_i \Pi_i^\top = \gamma_i^2 \Pi_i \Sigma_i \Pi_i^\top = \gamma_i^2 \Sigma_i^{2\mathrm{D}}. \tag{27}$$

Table 14: Matched-budget comparison on Mill-19 and UrbanScene3D. EDGS+3DGS uses matched initialization with densification disabled. Time is in minutes and N_{GS} in millions. Best quality and timing results are bold.

Method	Mill-19					UrbanScene3D				
	Time ↓	PSNR ↑	SSIM ↑	LPIPS ↓	N_{GS}	Time ↓	PSNR ↑	SSIM ↑	LPIPS ↓	N_{GS}
Taming-3DGS	57	22.52	0.693	0.384	3.76	61	21.14	0.719	0.332	3.05
EDGS+3DGS	83	22.57	0.716	0.372	3.76	91	21.17	0.716	0.330	3.05
EffGS (Ours)	**20**	**23.68**	**0.755**	**0.279**	3.76	**22**	**21.50**	**0.778**	**0.281**	3.05

Table 15: Extended ablation of learnable primitive compactness, averaged over Garden, Room, Bonsai, Rubble, and Modern Building. N_{GS} is reported in millions. The best results are shown in bold.

Method	Frequency Error			Reconstruction Quality			N_{GS} ↓
	Low ↓	Mid ↓	High ↓	PSNR ↑	SSIM ↑	LPIPS ↓	
w/o Learnable Compactness (γ_i)	86726.7	30836.8	7073.6	26.99	0.810	0.264	2.30
EffGS (Ours)	**72713.2**	**25773.5**	**5678.1**	**28.51**	**0.855**	**0.194**	**2.02**

For a fixed squared Mahalanobis level $q > 0$, let $\ell_{i,\kappa}$ denote the κ-th eigenvalue of Σ_i^{2D}, where $\kappa \in \{1, 2\}$ indexes the two principal axes. The corresponding ellipse semi-axis length is

$$\rho_{i,\kappa} = \sqrt{q\gamma_i^2 \ell_{i,\kappa}} = \gamma_i \sqrt{q\ell_{i,\kappa}}. \tag{28}$$

Thus, with the base covariance held fixed, a smaller factor contracts the ideal projected support and can reduce the number of intersected tiles.

If a fixed screen-space covariance $\varepsilon_{\mathrm{cov}}\mathbf{I}_2$ is added during rasterization, where $\varepsilon_{\mathrm{cov}} > 0$ and $\mathbf{I}_2$ denotes the 2×2 identity matrix, the projected eigenvalues become $\gamma_i^2\ell_{i,\kappa}+\varepsilon_{\mathrm{cov}}$. The corresponding semi-axis lengths are

$$\sqrt{q\,(\gamma_i^2 \ell_{i,\kappa} + \varepsilon_{\mathrm{cov}})}. \tag{29}$$

Discrete tile assignment further makes the realized tile count change in steps as the support varies.

During training, both the base scales and γ_i are optimized. The resulting effective support therefore depends on their product, while total rasterization cost also depends on the number and arrangement of primitives.

Training and regularization. For the initial Gaussian primitives, we set $\beta_i = 0$, giving $\gamma_i = 1$. Primitives created by cloning or splitting inherit their parent's β_i, as specified in Sec. 3.3. The modulation parameters are subsequently optimized with a dedicated learning rate. The L2 term $\mathcal{R}_\gamma$ is exactly the regularizer defined in Sec. 3.5, with the same coefficient and reduction. It introduces a shrinkage preference on the modulation factors. Within the active set used by local density control, primitives with $\gamma_i < 0.01$ are removed.

Since the effective scales are $\gamma_i S_i$, a penalty or threshold on γ_i acts on the chosen parameterization rather than directly constraining Gaussian volume. Indeed, compensating changes in the base scales can preserve the effective covariance. The factor therefore serves as an optimization and primitive-control variable.

Role of the reparameterization. The base scales remain learnable, so the modulation does not enlarge the static family of representable Gaussian covariances. Its contribution lies in the optimization parameterization and its interaction with density control. The reported ablations show improved reconstruction quality and efficiency when this component is included in the full framework.

C.2 Frequency-Aware Mask Generation

Given an input image, we first average its color channels to obtain a grayscale image and downsample it by a factor of $s = 0.5$. We convolve the downsampled image with two Gaussian kernels having standard deviations σ_1 and $\sigma_2 = 2\sigma_1$, then take the absolute difference of the filtered images. Each discrete Gaussian kernel is normalized to unit sum, and zero padding is used to preserve the spatial dimensions.

The bandwidth is controlled by the schedule parameter $f \in [0, 100]$, which increases linearly with the training iteration:

$$\sigma_1 = \begin{cases} 0.1 + (100 - f) \times 0.1, & \text{if } f \geq 50, \\ 0.1 + f \times 0.1, & \text{otherwise.} \end{cases} \tag{30}$$

The bandwidth increases during the first stage and decreases during the second stage.

We upsample the absolute response to the original resolution and apply min–max normalization separately to each image. The minima and maxima are taken over spatial pixels, independently of the other images in the batch. For $f < 50$, thresholding the complemented response selects locations with relatively weak responses at the current scale. For $f \geq 50$, thresholding the direct response selects strong-response locations, with progressively narrower filters emphasizing finer structures. The resulting binary mask, denoted consistently by M_{freq}, is used in both importance scoring and the mask-weighted reconstruction loss.

Algorithm 1 summarizes the procedure. The mask provides a stage-dependent spatial selection signal derived from image-frequency responses. A spatially constant response has $H_{\text{norm}} = 0$ under the stated stabilization rule, yielding an all-one mask for $f < 50$ and an all-zero mask for $f \geq 50$. Its participation in density control is determined by the operation sets in Sec. C.3; the mask is also used in the reconstruction objective throughout training.

Here, $*$ denotes 2D convolution and $\mathbb{I}[\cdot]$ denotes the indicator function.

C.3 Density-Control Definitions and Execution Details

This subsection specifies the normalization, counting support, schedules, and sampling procedure used in Secs. 3.2 and 3.3. We use G_t, A_t, V_i, $s_i^d(t)$, s_i^p, H_t, P_t, b_t, τ_d, τ_p, and τ_s following the definitions in Sec. 3.3. Let $t \in \{1, \ldots, T_{\text{total}}\}$ denote the current training iteration, where T_{total} is the total number of training iterations.

Normalization. For a nonempty finite index set $\mathcal{S}$, a scalar-valued collection $\{x_a\}_{a\in\mathcal{S}}$, and $a \in \mathcal{S}$, define

$$\mathcal{N}_{\mathcal{S}}[x](a) = \begin{cases} \dfrac{x_a - x_{\min}}{x_{\max} - x_{\min}}, & x_{\max} > x_{\min}, \\ 0, & x_{\max} = x_{\min}, \end{cases} \qquad x_{\min} = \min_{b\in\mathcal{S}} x_b, \quad x_{\max} = \max_{b\in\mathcal{S}} x_b. \tag{31}$$

Pixel errors are normalized separately for each image, whereas pruning statistics are normalized over the current active set. Constant inputs receive no relative priority. The unnormalized pixel errors are used for photometric losses. Density control is skipped when the active set is empty.

Pixel counting support. Let B_i^j denote the image pixels in tiles assigned to Gaussian i by Compact Box. Let $\alpha_i^j(\mathbf{u})$ be its per-pixel compositing opacity and $T_i^j(\mathbf{u})$ the transmittance immediately before processing it in front-to-back order. We define

$$\Omega_i^j = \left\{ \mathbf{u} \in B_i^j : \begin{array}{l} \text{the compositing loop reaches Gaussian } i, \\ \alpha_i^j(\mathbf{u}) \geq \varepsilon_\alpha, \\ T_i^j(\mathbf{u})\big(1 - \alpha_i^j(\mathbf{u})\big) \geq \varepsilon_T \end{array} \right\}, \tag{32}$$

where $\varepsilon_\alpha > 0$ and $0 < \varepsilon_T < 1$ are the renderer's opacity and early-termination tolerances. Thus, a pixel is counted only when Gaussian i is reached by the compositing loop, passes the opacity test, and is processed before the renderer's early-termination condition is triggered. We set $\Omega_i^j = \varnothing$ for $j \notin V_i$.

Algorithm 1 Frequency-Aware Mask Generation

Require: Input image batch $I \in \mathbb{R}^{B\times C\times H_{\text{img}}\times W_{\text{img}}}$, schedule parameter $f \in [0, 100]$, and scale factor s (e.g., $s = 0.5$), where B, C, H_{img}, and W_{img} denote the batch size, number of channels, image height, and image width, respectively

Ensure: Binary mask $M_{\text{freq}} \in \{0, 1\}^{B\times 1\times H_{\text{img}}\times W_{\text{img}}}$

1: Convert I to grayscale:

$$I_{\text{gray}} \leftarrow \text{mean}(I, \text{dim} = 1, \text{keepdim} = \text{true}), \qquad I_{\text{gray}} \in \mathbb{R}^{B\times 1\times H_{\text{img}}\times W_{\text{img}}}$$

2: Downsample the grayscale image:

$$I_{\text{down}} \leftarrow \text{BilinearInterp}(I_{\text{gray}}, s)$$

3: **if** $f \geq 50$ **then**
4: $\quad \sigma_1 \leftarrow 0.1 + (100 - f) \times 0.1$
5: **else**
6: $\quad \sigma_1 \leftarrow 0.1 + f \times 0.1$
7: **end if**
8: $\sigma_2 \leftarrow 2\sigma_1$
9: Compute the Gaussian kernel sizes:

$$k_1 \leftarrow 2\lceil 3\sigma_1\rceil + 1, \qquad k_2 \leftarrow 2\lceil 3\sigma_2\rceil + 1$$

10: Generate Gaussian kernels:

$$\mathcal{K}_1 \leftarrow \text{GaussianKernel}(k_1, \sigma_1), \qquad \mathcal{K}_2 \leftarrow \text{GaussianKernel}(k_2, \sigma_2)$$

11: Apply dimension-preserving zero-padded convolution:

$$I_1^{\text{blur}} \leftarrow \mathcal{K}_1 * I_{\text{down}}, \qquad I_2^{\text{blur}} \leftarrow \mathcal{K}_2 * I_{\text{down}}$$

12: Compute the absolute difference-of-Gaussians response:

$$H_{\text{freq}} \leftarrow \left|I_1^{\text{blur}} - I_2^{\text{blur}}\right|$$

13: Upsample the response:

$$H_{\text{up}} \leftarrow \text{BilinearInterp}\left(H_{\text{freq}}, (H_{\text{img}}, W_{\text{img}})\right)$$

14: Normalize each image independently over its spatial pixels, where $\varepsilon_{\text{norm}} > 0$ is a numerical stabilization constant:

$$H_{\text{norm}} \leftarrow \frac{H_{\text{up}} - \min(H_{\text{up}})}{\max(H_{\text{up}}) - \min(H_{\text{up}}) + \varepsilon_{\text{norm}}}$$

15: **if** $f \geq 50$ **then**
16: $\quad M_{\text{freq}} \leftarrow \mathbb{I}[H_{\text{norm}} \geq 0.5]$
17: **else**
18: $\quad M_{\text{freq}} \leftarrow \mathbb{I}[1 - H_{\text{norm}} \geq 0.5]$
19: **end if**
20: **return** M_{freq}

Densification weight. The temporal weight in Eq. equation 8 is

$$u_d(t) = \min\left\{1, \max\left\{0, \frac{t - t_d^{\text{start}}}{t_d^{\text{end}} - t_d^{\text{start}}}\right\}\right\}, \qquad \omega(t) = \omega_{\text{start}} + (\omega_{\text{end}} - \omega_{\text{start}})u_d(t), \tag{33}$$

with $0 < \omega_{\text{start}} \leq \omega_{\text{end}}$ and $t_d^{\text{start}} < t_d^{\text{end}}$. For a fixed count and threshold, increasing $\omega(t)$ makes the densification criterion easier to satisfy. The weight is used only at scheduled densification steps.

Operation schedules. Let $t_d^{\text{start}} < t_d^{\text{end}}$ and $t_p^{\text{start}} \le t_p^{\text{end}}$ be integer endpoints within $[1, T_{\text{total}}]$. For positive integer intervals Δ_d and Δ_p, define

$$\begin{aligned} \mathcal{I}_d &= \left\{ t \in \mathbb{Z} : t_d^{\text{start}} \le t \le t_d^{\text{end}}, \quad (t - t_d^{\text{start}}) \bmod \Delta_d = 0 \right\}, \\ \mathcal{I}_p &= \left\{ t \in \mathbb{Z} : t_p^{\text{start}} \le t \le t_p^{\text{end}}, \quad (t - t_p^{\text{start}}) \bmod \Delta_p = 0 \right\}. \end{aligned} \tag{34}$$

Density control occurs at $t \in \mathcal{I}_d \cup \mathcal{I}_p$. At each such update, we sample K distinct views uniformly without replacement, where N_{train} denotes the number of training views and $1 \le K \le N_{\text{train}}$.

The frequency schedule $f(t) = 100t/T_{\text{total}}$ is not restarted for density control. For either operation $o \in \{d, p\}$, the complemented and direct masks participate at

$$\mathcal{I}_o \cap \{t : t < T_{\text{total}}/2\}, \qquad \mathcal{I}_o \cap \{t : t \ge T_{\text{total}}/2\}, \tag{35}$$

respectively. Either intersection may be empty. Mask-weighted supervision follows the same frequency schedule throughout training, including iterations without density updates.

Weighted pruning without replacement. For $t \in \mathcal{I}_p$, form H_t, P_t, and b_t as defined in Sec. 3.3. Initialize $U_0 = P_t$. For $\nu = 1, \ldots, b_t$, sample

$$\Pr(i_\nu = i \mid U_{\nu-1}) = \frac{s_i^p}{\sum_{k \in U_{\nu-1}} s_k^p}, \qquad i \in U_{\nu-1}, \tag{36}$$

and update

$$U_\nu = U_{\nu-1} \setminus \{i_\nu\}. \tag{37}$$

All candidate weights are positive because $s_i^p > \tau_p \ge 0$. The removal set is

$$R_t = H_t \cup \{i_1, \ldots, i_{b_t}\}. \tag{38}$$

When $b_t = 0$, the sampled subset is empty. For $t \notin \mathcal{I}_p$, set $R_t = \varnothing$.

Densification and update order. Visibility sets and scores are computed from the same pre-update representation G_t. Pruning, when scheduled, is applied first. For $t \in \mathcal{I}_d$, densification candidates are

$$D_t = \left\{ i \in A_t \setminus R_t : s_i^d(t) > \tau_d \right\}, \qquad \tau_d \ge 0. \tag{39}$$

Let

$$s_{\max,i} = \max\{s_i^x, s_i^y, s_i^z\}. \tag{40}$$

For $s_{\max,i} < \tau_s$, with $\tau_s > 0$, we clone the primitive and shift the clone along the positional gradient. Otherwise, we replace the parent with two smaller Gaussians. The branch uses base scales, whereas rendering uses the effective scales $\gamma_i S_i$.

New primitives inherit the parent's modulation parameter β_i, and hence γ_i; their other attributes follow the underlying clone/split updates. They receive fresh visibility estimates and scores at the next scheduled density update and are not pruned during their creation step. Every density modification is restricted to A_t.

D Limitations

Our evaluation focuses on static-scene novel view synthesis across bounded and urban-scale benchmarks. The effectiveness of EffGS for time-varying scene reconstruction remains unevaluated. The reported results characterize the quality–efficiency trade-off under the evaluated training configurations. Although matched-primitive-budget comparisons control Gaussian counts, they do not characterize performance across all computational budgets or hardware settings. Further evaluation is needed to establish how these trade-offs extend beyond the tested configurations.

E Broader Social Impacts

This work introduces EffGS, an efficient framework for high-fidelity large-scale 3D scene reconstruction. While our approach positively contributes to environmental sustainability by significantly reducing computational overhead and enables beneficial applications in urban planning and simulation, we acknowledge the potential for misuse in privacy-sensitive contexts, such as unauthorized spatial reconstruction. To mitigate such risks, the deployment of our framework should be governed by clear ethical guidelines, rigorous dataset sanitization prior to training, and regulatory oversight.