SafeguardGS: 3D Gaussian Primitive Pruning While Avoiding Catastrophic Scene Destruction

Supplementary Material

6. Score functions

The rendering equation Eq. (1) gives good hints for scoring the importance of Gaussian primitives since it directly relates to pixel colors. From the factors and intermediate values of Eq. (1) (i.e., $c_i, \boldsymbol{\sigma}_i, \alpha_i, \mathcal{G}_i, T_i$) , we design multiple candidates of effective score function. Because of $\alpha_i = \boldsymbol{\sigma}_i \mathcal{G}_i$, rather than we use directly $\mathcal{G}_i$, we choose the distance between the center of $\mathcal{G}_i$ and pixel position. Furthermore, to ensure the factors doesn't dominate in the score calculation, we properly map (squash) their values into $[0, 1]$ by using L1-norm, L2-norm, and cosine similarity function. The squashing functions are:

$$
\begin{aligned}
PC(X_r, X_i) &= \exp(-||X_r - X_i||_2), \\
CS_1(\mathbf{c}_{r,gt}, c_i) &= (1 - \frac{1}{\mathcal{C}}(||\mathbf{c}_{r,gt} - c_i||_1)), \\
CS_2(\mathbf{c}_{r,gt}, c_i) &= \exp(-\frac{1}{\mathcal{C}}(||\mathbf{c}_{r,gt} - c_i||_1)), \\
CS_3(\mathbf{c}_{r,gt}, c_i) &= (\mathbf{c}_{r,gt} \cdot c_i)/(||\mathbf{c}_{r,gt}||_2 * ||c_i||_2),
\end{aligned}
$$

where X_r and $\mathbf{c}_{r,gt}$ are, respectively, the pixel coordinates and color of the ray r, X_i is center of $\mathcal{G}_i$, $\cdot$ is vector inner product, and $*$ is scalar multiplication.

Tab. 5 lists the full set of functions. Note that the number id in the function name $Score_{SG\_id}$ is nothing but the score function id.

7. Preliminary Search of Effective Score Function Candidates

Given the full set of functions from Tab. 5, we performed a pilot study to reduce the number of candidates for the top-k parameter sweep experiment. In the pilot study, we selected two datasets (i.e., Synthetic NeRF [30] and MipNeRF360 [3]), as shown in Figs. 8 and 9. We trained 3DGS for 30k iterations and performed pruning once at 20k with the top-1 setting for each score function. Other hyperparameters were kept the same as in the vanilla 3DGS. To make the plot legible, we divided the points into two groups. Based on $Score_{SG\_3}$ and $Score_{SG\_4}$, we narrowed down the candidates, resulting in Tab. 1.

The functions listed in Tab. 5 are indexed according to a bit-masking rule. For example, 0x00–0x0f corresponds to key combinations of rendering factors; 0x10 multiplies CS_3; 0x20 multiplies CS_1; 0x30 multiplies CS_2; 0x40 adds CS_1; and 0x50 adds CS_2. The score functions that follow the bit-masking rule but are not listed in Tab. 5 were filtered out during the pilot study because they exhibited significantly degraded performance in the top-k=1 setting on the bicycle scene.

We further present the top-k parameter sweep results

Table 5. The full list of score functions

$Score_{SG\_1}(\mathbf{G}_i, r) = \boldsymbol{\sigma}_i$
$Score_{SG\_2}(\mathbf{G}_i, r) = \alpha_i$
$Score_{SG\_3}(\mathbf{G}_i, r) = \boldsymbol{\sigma}_i * T_i$
$Score_{SG\_4}(\mathbf{G}_i, r) = \alpha_i * T_i$
$Score_{SG\_5}(\mathbf{G}_i, r) = PC(X_r, X_i)$
$Score_{SG\_6}(\mathbf{G}_i, r) = PC(X_r, X_i) * \boldsymbol{\sigma}_i$
$Score_{SG\_7}(\mathbf{G}_i, r) = PC(X_r, X_i) * \alpha_i$
$Score_{SG\_8}(\mathbf{G}_i, r) = PC(X_r, X_i) * \boldsymbol{\sigma}_i * T_i$
$Score_{SG\_9}(\mathbf{G}_i, r) = PC(X_r, X_i) * \alpha_i * T_i$
$Score_{SG\_11}(\mathbf{G}_i, r) = \alpha_i + T_i$
$Score_{SG\_16}(\mathbf{G}_i, r) = CS_3(\mathbf{c}_{r,gt}, c_i)$
$Score_{SG\_17}(\mathbf{G}_i, r) = CS_3(\mathbf{c}_{r,gt}, c_i) * \boldsymbol{\sigma}_i$
$Score_{SG\_18}(\mathbf{G}_i, r) = CS_3(\mathbf{c}_{r,gt}, c_i) * \alpha_i$
$Score_{SG\_19}(\mathbf{G}_i, r) = CS_3(\mathbf{c}_{r,gt}, c_i) * \boldsymbol{\sigma}_i * T_i$
$Score_{SG\_20}(\mathbf{G}_i, r) = CS_3(\mathbf{c}_{r,gt}, c_i) * \alpha_i * T_i$
$Score_{SG\_21}(\mathbf{G}_i, r) = CS_3(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i)$
$Score_{SG\_22}(\mathbf{G}_i, r) = CS_3(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i) * \boldsymbol{\sigma}_i$
$Score_{SG\_23}(\mathbf{G}_i, r) = CS_3(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i) * \alpha_i$
$Score_{SG\_24}(\mathbf{G}_i, r) = CS_3(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i) * \boldsymbol{\sigma}_i * T_i$
$Score_{SG\_25}(\mathbf{G}_i, r) = CS_3(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i) * \alpha_i * T_i$
$Score_{SG\_32}(\mathbf{G}_i, r) = CS_1(\mathbf{c}_{r,gt}, c_i)$
$Score_{SG\_33}(\mathbf{G}_i, r) = CS_1(\mathbf{c}_{r,gt}, c_i) * \boldsymbol{\sigma}_i$
$Score_{SG\_34}(\mathbf{G}_i, r) = CS_1(\mathbf{c}_{r,gt}, c_i) * \alpha_i$
$Score_{SG\_35}(\mathbf{G}_i, r) = CS_1(\mathbf{c}_{r,gt}, c_i) * \boldsymbol{\sigma}_i * T_i$
$Score_{SG\_36}(\mathbf{G}_i, r) = CS_1(\mathbf{c}_{r,gt}, c_i) * \alpha_i * T_i$
$Score_{SG\_37}(\mathbf{G}_i, r) = CS_1(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i)$
$Score_{SG\_38}(\mathbf{G}_i, r) = CS_1(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i) * \boldsymbol{\sigma}_i$
$Score_{SG\_39}(\mathbf{G}_i, r) = CS_1(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i) * \alpha_i$
$Score_{SG\_40}(\mathbf{G}_i, r) = CS_1(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i) * \boldsymbol{\sigma}_i * T_i$
$Score_{SG\_41}(\mathbf{G}_i, r) = CS_1(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i) * \alpha_i * T_i$
$Score_{SG\_48}(\mathbf{G}_i, r) = CS_2(\mathbf{c}_{r,gt}, c_i)$
$Score_{SG\_49}(\mathbf{G}_i, r) = CS_2(\mathbf{c}_{r,gt}, c_i) * \boldsymbol{\sigma}_i$
$Score_{SG\_50}(\mathbf{G}_i, r) = CS_2(\mathbf{c}_{r,gt}, c_i) * \alpha_i$
$Score_{SG\_51}(\mathbf{G}_i, r) = CS_2(\mathbf{c}_{r,gt}, c_i) * \boldsymbol{\sigma}_i * T_i$
$Score_{SG\_52}(\mathbf{G}_i, r) = CS_2(\mathbf{c}_{r,gt}, c_i) * \alpha_i * T_i$
$Score_{SG\_53}(\mathbf{G}_i, r) = CS_2(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i)$
$Score_{SG\_54}(\mathbf{G}_i, r) = CS_2(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i) * \boldsymbol{\sigma}_i$
$Score_{SG\_55}(\mathbf{G}_i, r) = CS_2(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i) * \alpha_i$
$Score_{SG\_56}(\mathbf{G}_i, r) = CS_2(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i) * \boldsymbol{\sigma}_i * T_i$
$Score_{SG\_57}(\mathbf{G}_i, r) = CS_2(\mathbf{c}_{r,gt}, c_i) * PC(X_r, X_i) * \alpha_i * T_i$
$Score_{SG\_68}(\mathbf{G}_i, r) = CS_1(\mathbf{c}_{r,gt}, c_i) + \alpha_i * T_i$
$Score_{SG\_75}(\mathbf{G}_i, r) = CS_1(\mathbf{c}_{r,gt}, c_i) + \alpha_i + T_i$
$Score_{SG\_84}(\mathbf{G}_i, r) = CS_2(\mathbf{c}_{r,gt}, c_i) + \alpha_i * T_i$
$Score_{SG\_91}(\mathbf{G}_i, r) = CS_2(\mathbf{c}_{r,gt}, c_i) + \alpha_i + T_i$

in terms of PSNR, SSIM, and LPIPS metrics across four datasets: Synthetic NeRF [30], MipNeRF360 [3], TanksAndTemples [21], and Deep Blending [18], as shown in Figs. 10 to 13.

8. Color Similarity Distribution of Primitives

Across all scenes, we consistently observe that the color similarity distribution is pushed toward 1, as marked by the lower 1% of primitives' color similarity values through Figs. 14 to 17. Moreover, we observe that the number of primitives remaining after pruning with $Score_{SG\_36}$ is generally smaller than that with $Score_{SG\_4}$. This phenomenon implies that $Score_{SG\_36}$ more effectively selects primitives crucial for maintaining scene rendering quality.

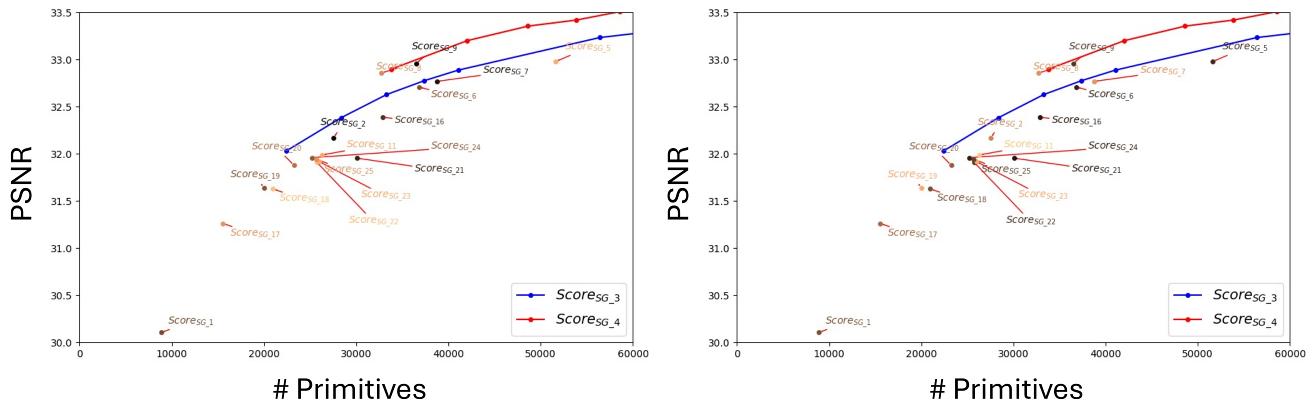
9. Rendering Result of different Pruning Techniques

Through Figs. 18 to 20, we present selected rendered images for each scene from MipNeRF360 [3], TanksAndTemples [21], and Deep Blending [18] to illustrate the visual quality of pruned scenes using the diverse pruning techniques: $Prune_{LG}$ [9], $Prune_{MS}$ [10], $Prune_{RS}$ [34], $Prune_{EG}$ [27], and $Prune_{SG}$.

10. Rendering Result of Application SafeguardGS to SoTA Pruning Pipeline

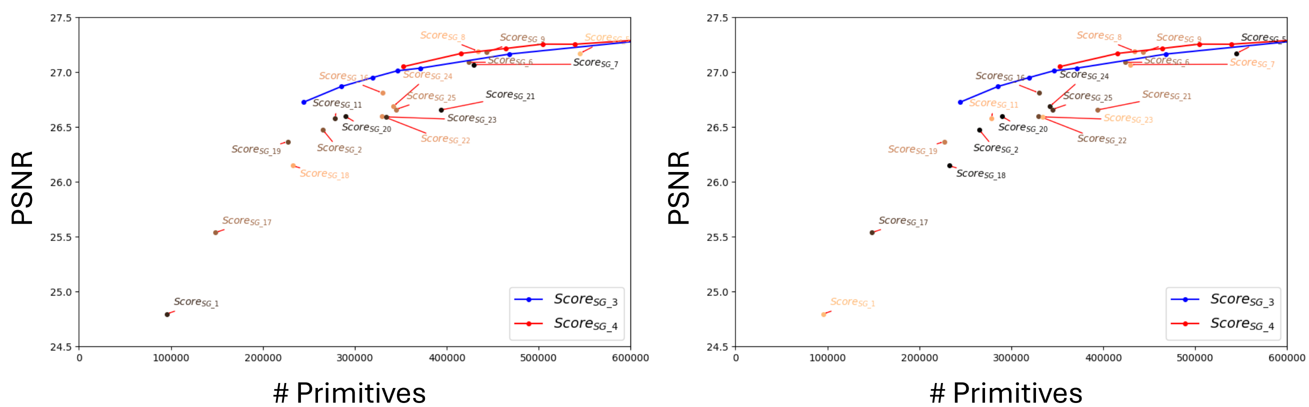
In this section, Figs. 21 to 23 provide selected rendered images for MipNeRF360 [3], TanksAndTemples [21], and Deep Blending [18] to illustrate the visual quality of pruned scenes using the diverse baselines and SafeguardGS-applied baselines: LightGaussian [9], LightGaussian-GS, PUP 3D-GS [17], PUP 3D-GS-SG.

We also provide point visualization of Gaussian primitives from Fig. 7 through Figs. 24 to 26. The point visualizations show SafeguardGS leaves more primitives at high detailed area, e.g., letters of '713' on `train` scene and plush toys in 'bonsai' scene.



(a) Group 1 (a) Group 2

Figure 8. PSNR vs. #Primitives plot for the top-1 setting of score functions in Tab. 5. The PSNR values are averaged across 8 scenes from Synthetic NeRF [30]. For clarity, the dots are plotted separately in group 1 (a) and group 2 (b).



(a) Group 1 (a) Group 2

Figure 9. PSNR vs. #Primitives plot for the top-1 setting of score functions in Tab. 5. The PSNR values are averaged across 9 scenes from MipNeRF360 [3]. For clarity, the dots are plotted separately in group 1 (a) and group 2 (b).

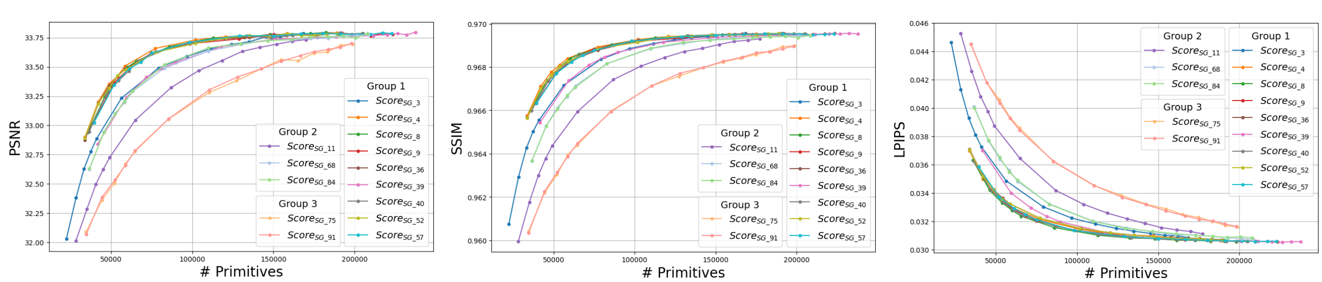


Figure 10. Sweep on the top-k parameter of the selected functions in Tab. 1. The values are averaged over 8 scenes from Synthetic NeRF [30].

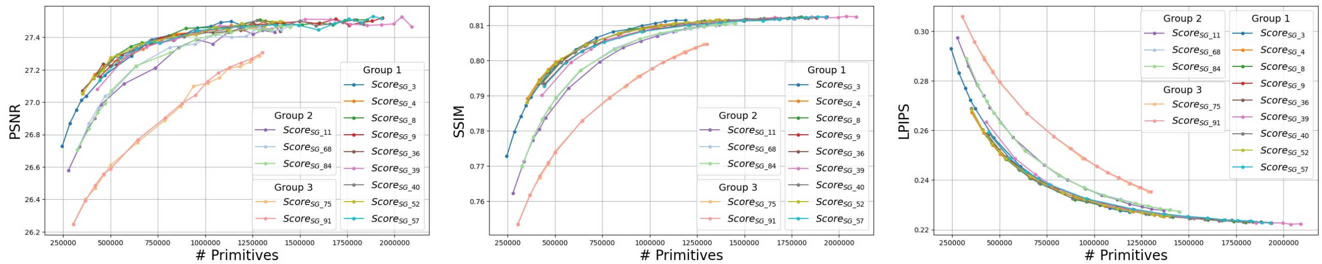


Figure 11. Sweep on the top-k parameter of the selected functions in Tab. 1. The values are averaged over 9 scenes from MipNeRF360 [3].

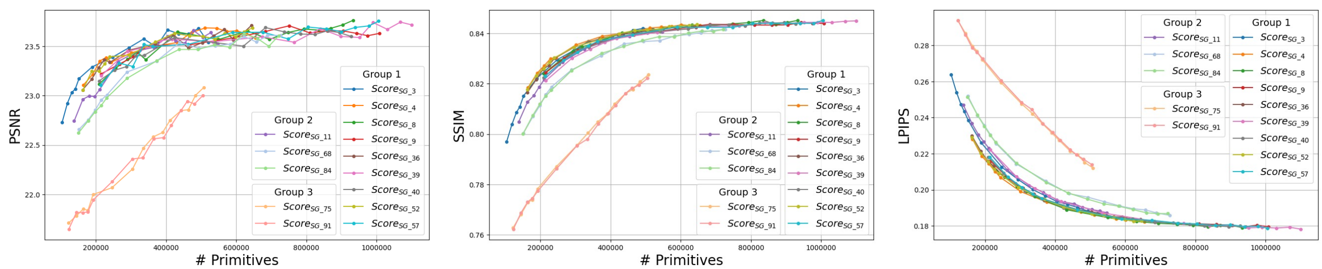


Figure 12. Sweep on the top-k parameter of the selected functions in Tab. 1. The values are averaged over 2 scenes from TanksAndTemples [21].

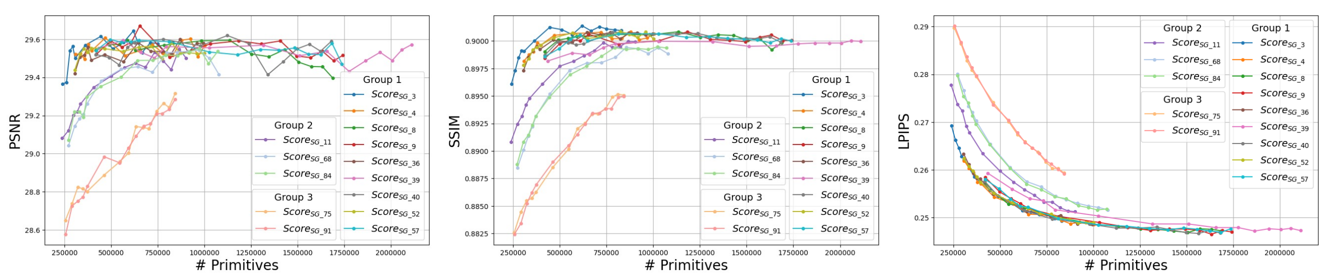


Figure 13. Sweep on the top-k parameter of the selected functions in Tab. 1. The values are averaged over 2 scenes from Deep Blending [18].

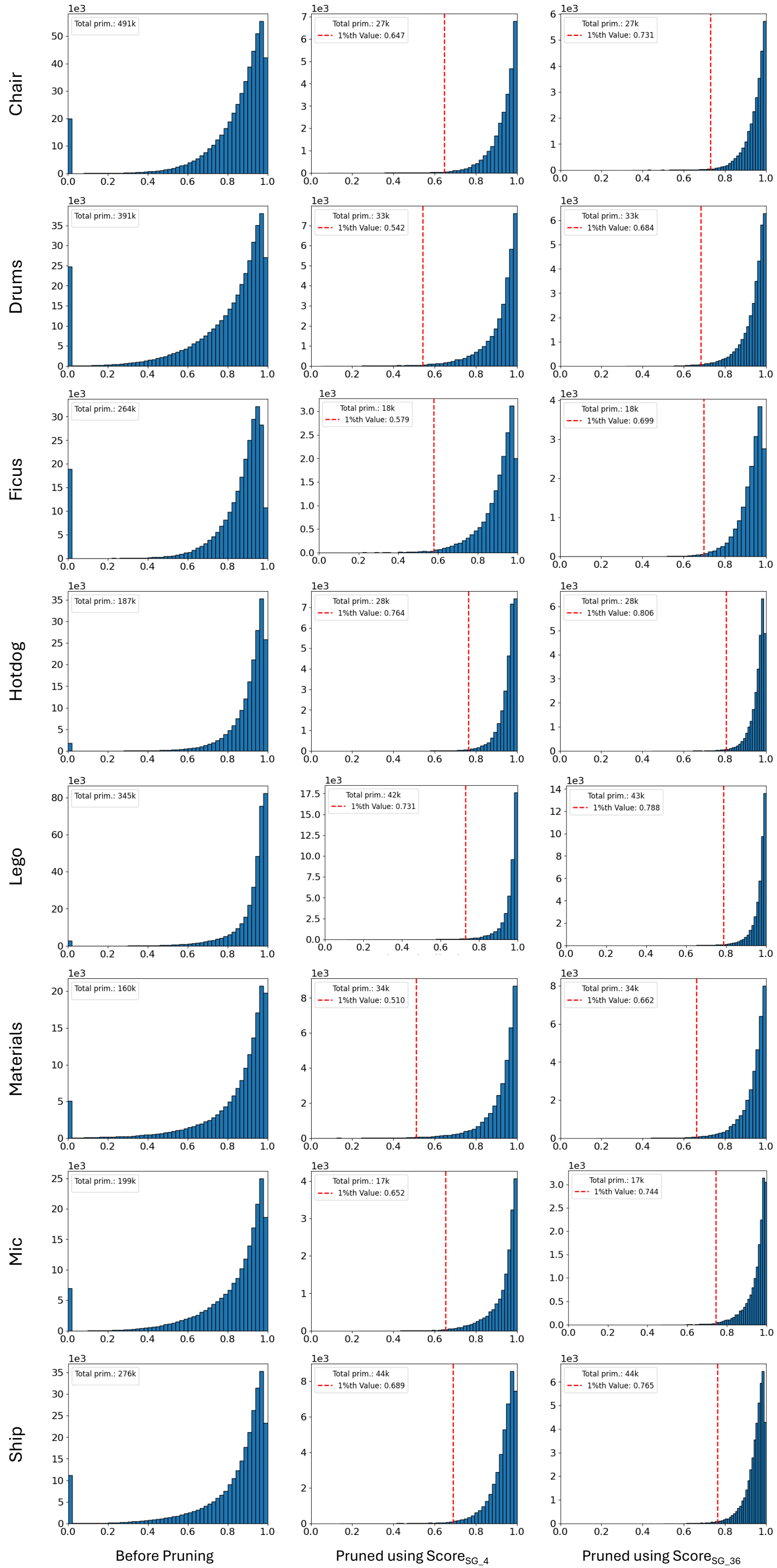


Figure 14. Histogram of primitives' color similarity across 8 scenes from Synthetic NeRF [30]. The X-axis represents color similarity, and the Y-axis shows the number of primitives associated into each bin. The dotted red line indicates the boundary of lower 1% color similarity value.

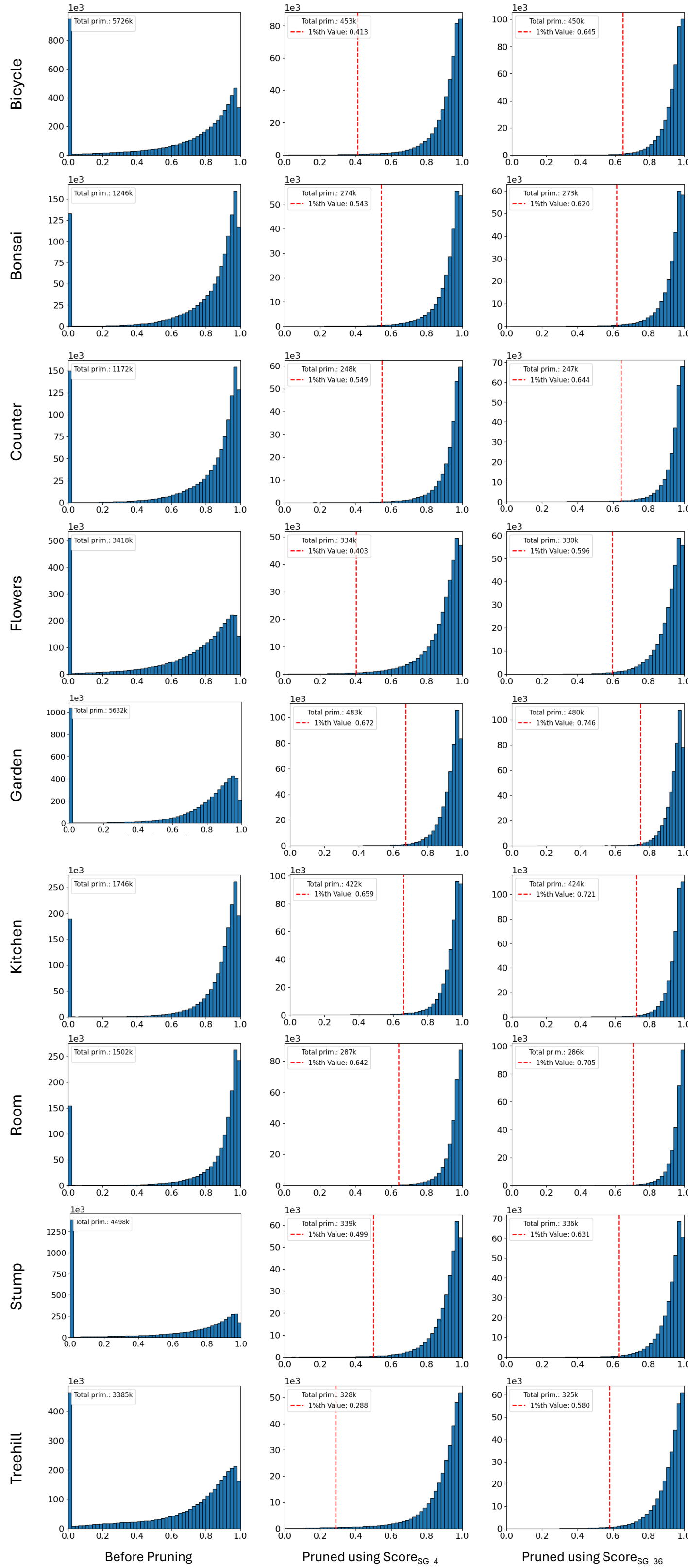


Figure 15. Histogram of primitive's color similarity over 9 scenes from MipNeRF360 [3]. The X-axis represents color similarity, and the Y-axis shows the number of primitives associated into each bin. The dotted red line indicates the boundary of lower 1% color similarity value.

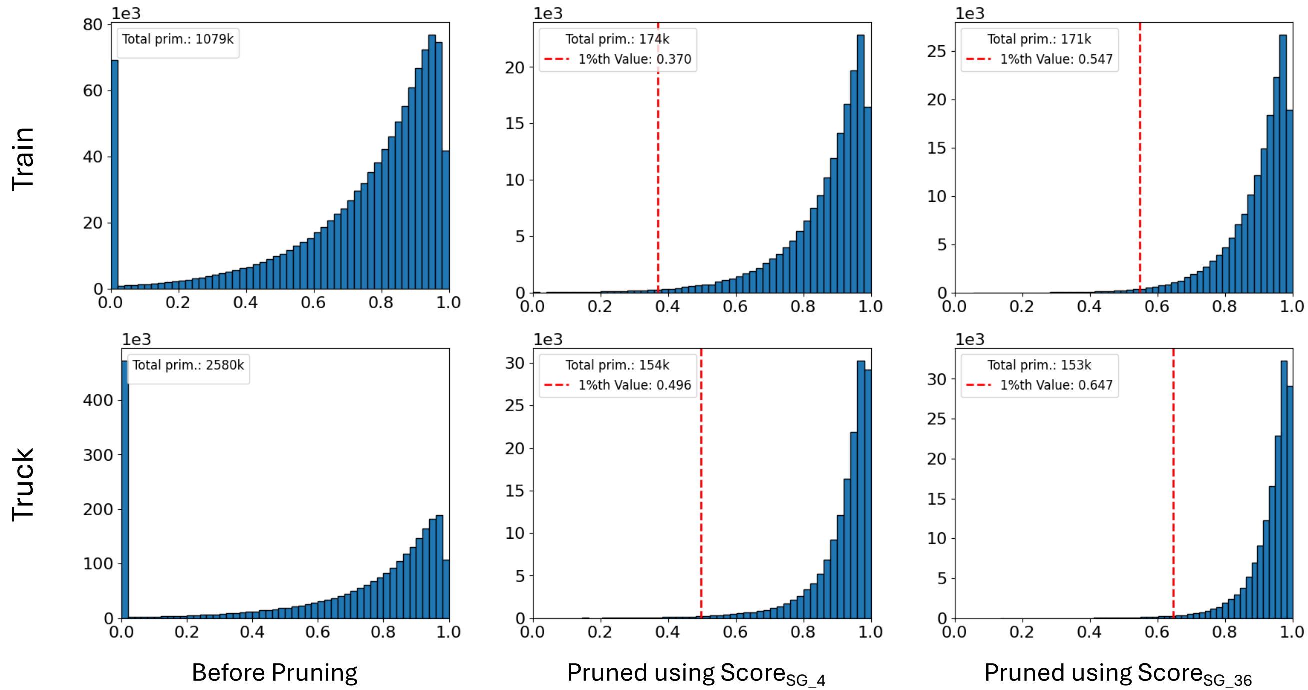


Figure 16. Histogram of primitive's color similarity over 2 scenes from TanksAndTemples [21]. The X-axis represents color similarity, and the Y-axis shows the number of primitives associated into each bin. The dotted red line indicates the boundary of lower 1% color similarity value.

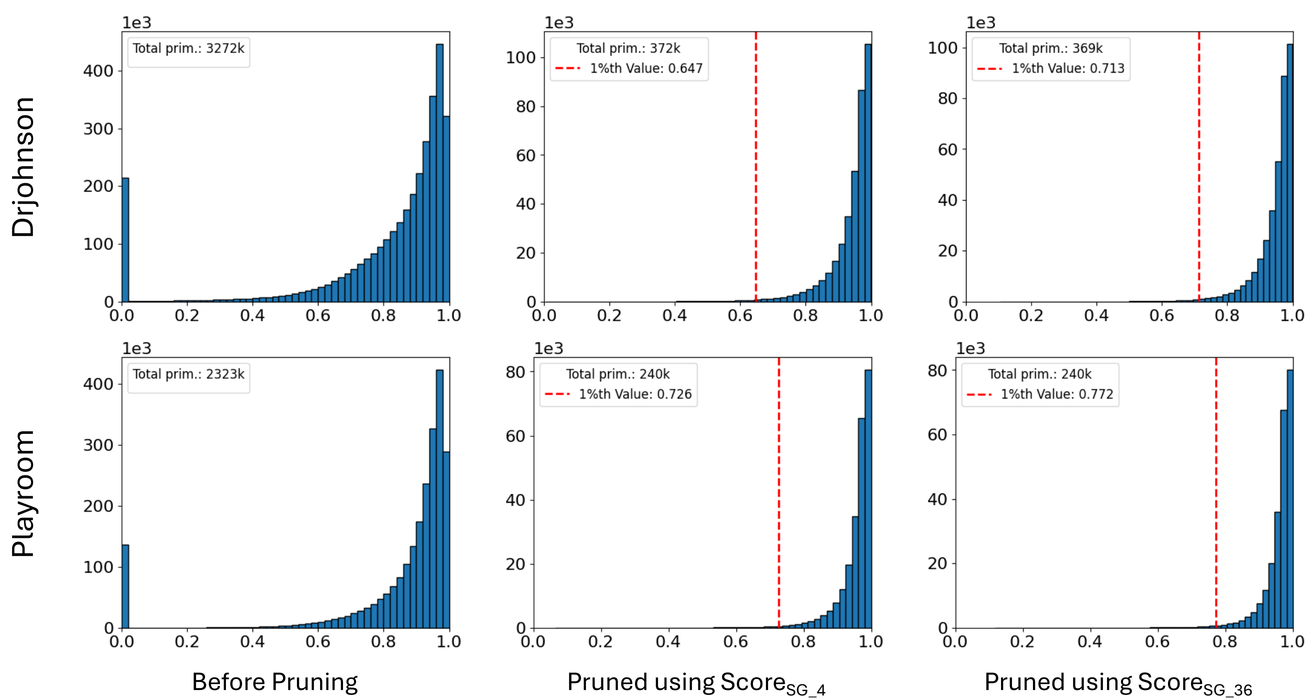


Figure 17. Histogram of primitive's color similarity over 2 scenes from Deep Blending [18]. The X-axis represents color similarity, and the Y-axis shows the number of primitives associated into each bin. The dotted red line indicates the boundary of lower 1% color similarity value.

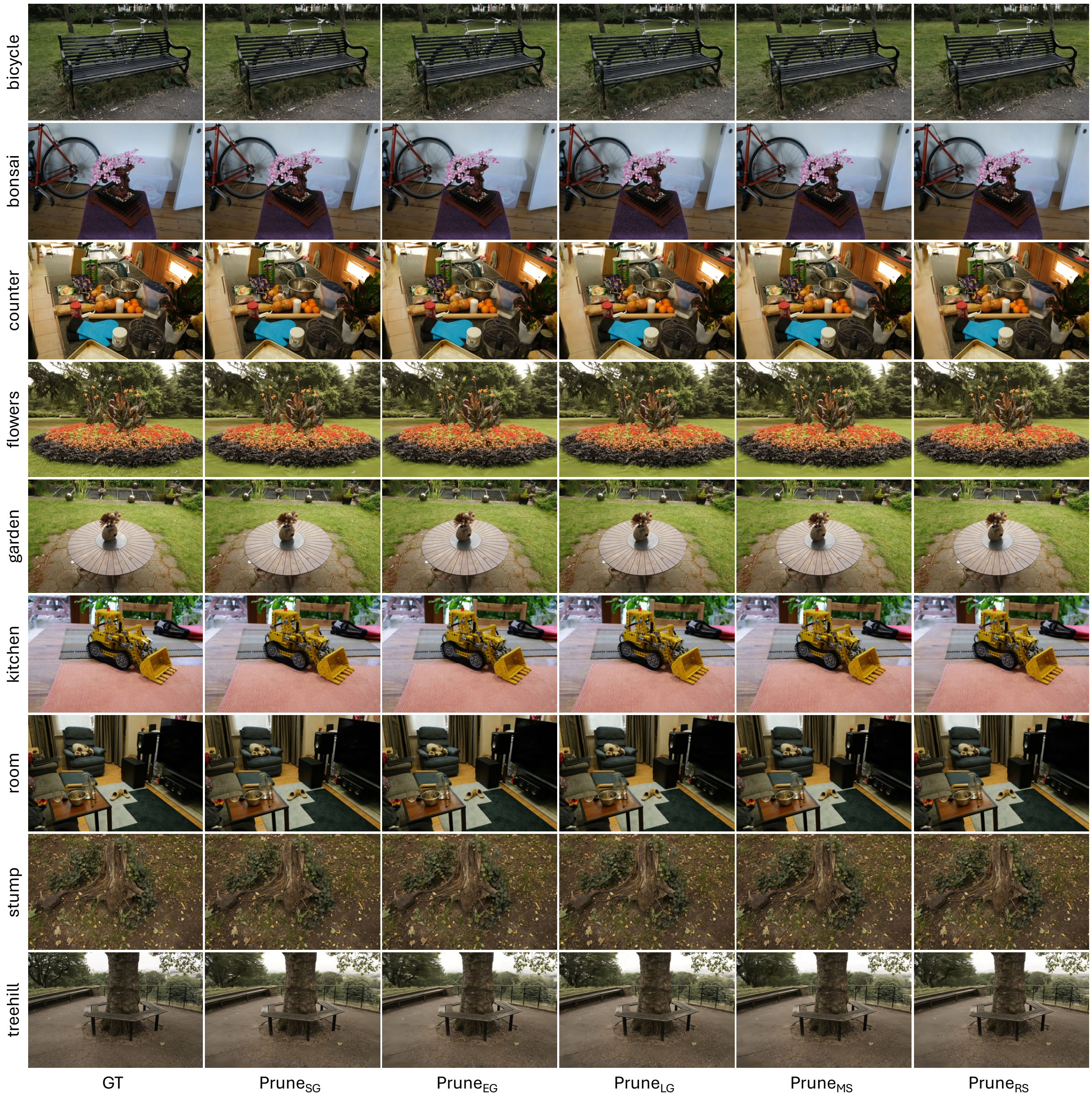


Figure 18. Rendering result of MipNeRF360 [3]

Figure 19. Rendering result of TanksAndTemples [21]

Figure 20. Rendering result of Deep Blending [18]

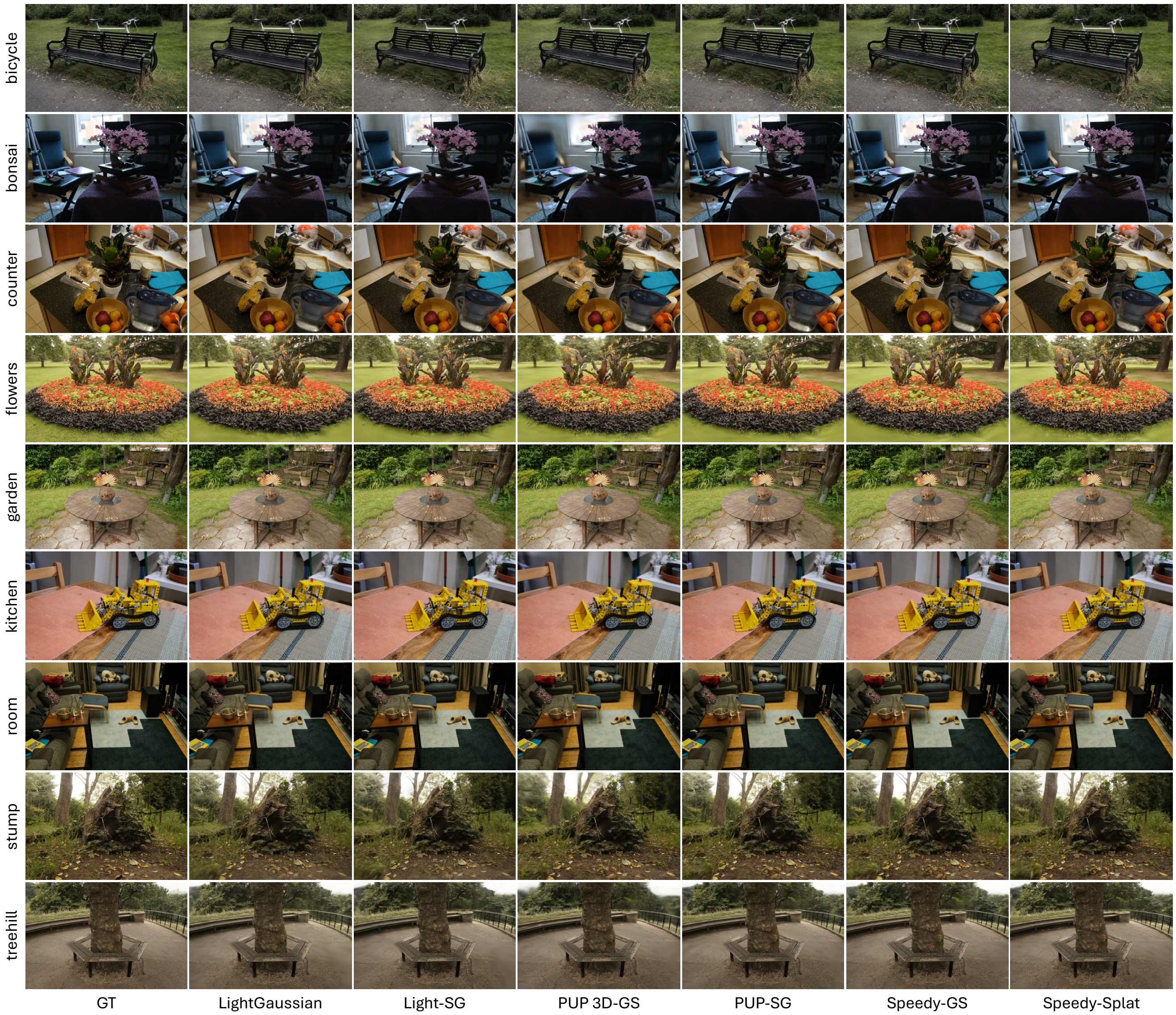


Figure 21. Rendering result of MipNeRF360 [3]

Figure 22. Rendering result of TanksAndTemples [21]

Figure 23. Rendering result of Deep Blending [18]

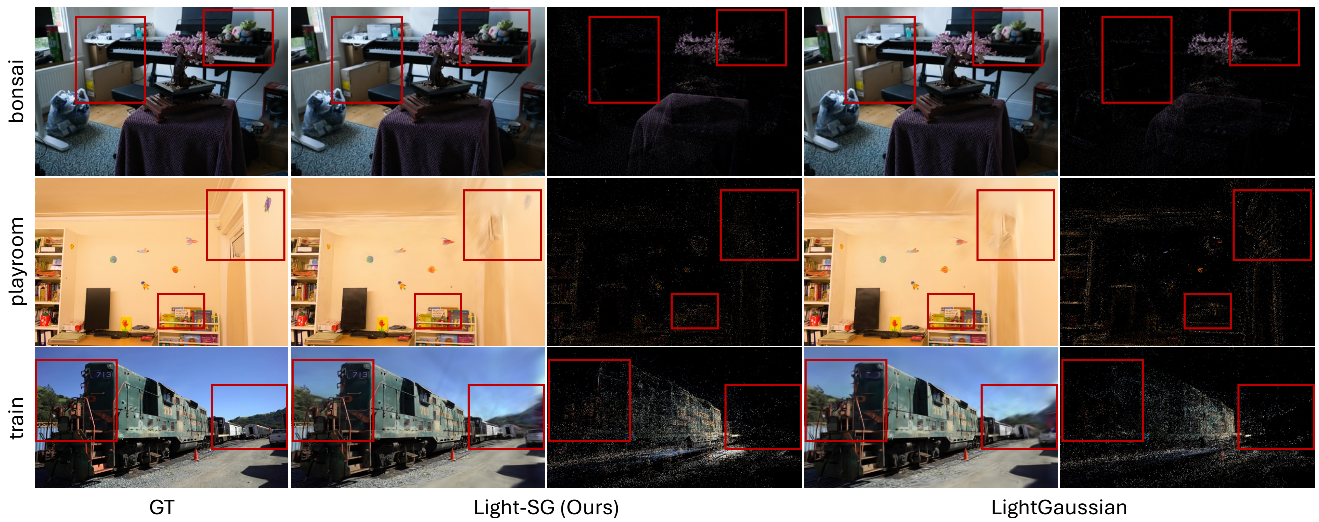


Figure 24. Point visualization of Light-SG and LightGaussian [9]

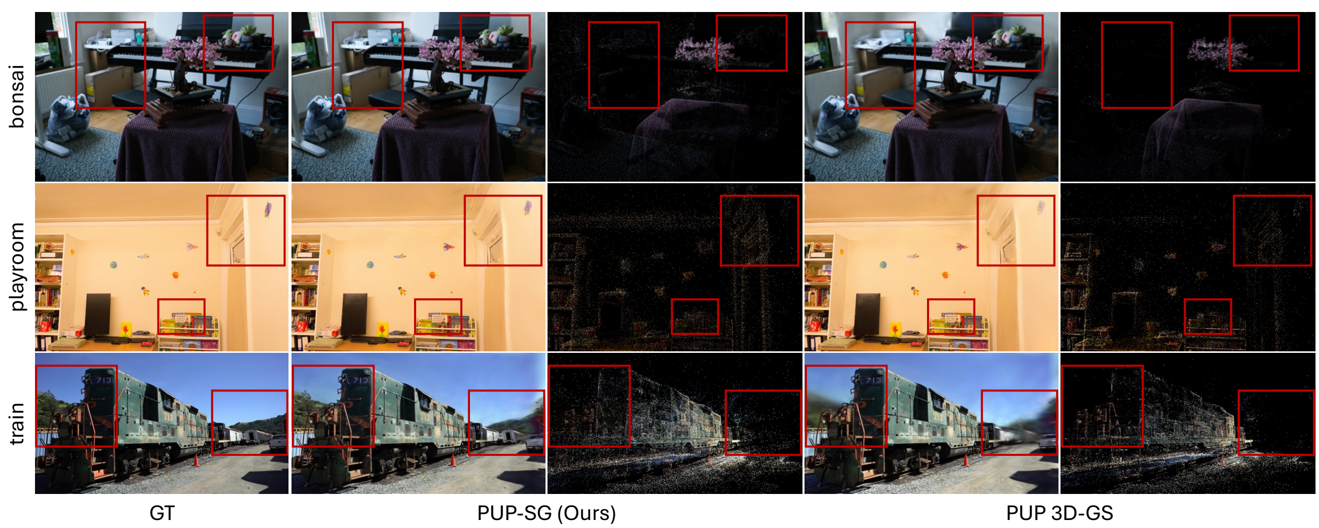


Figure 25. Point visualization of PUP-SG and PUP 3D-GS [17]

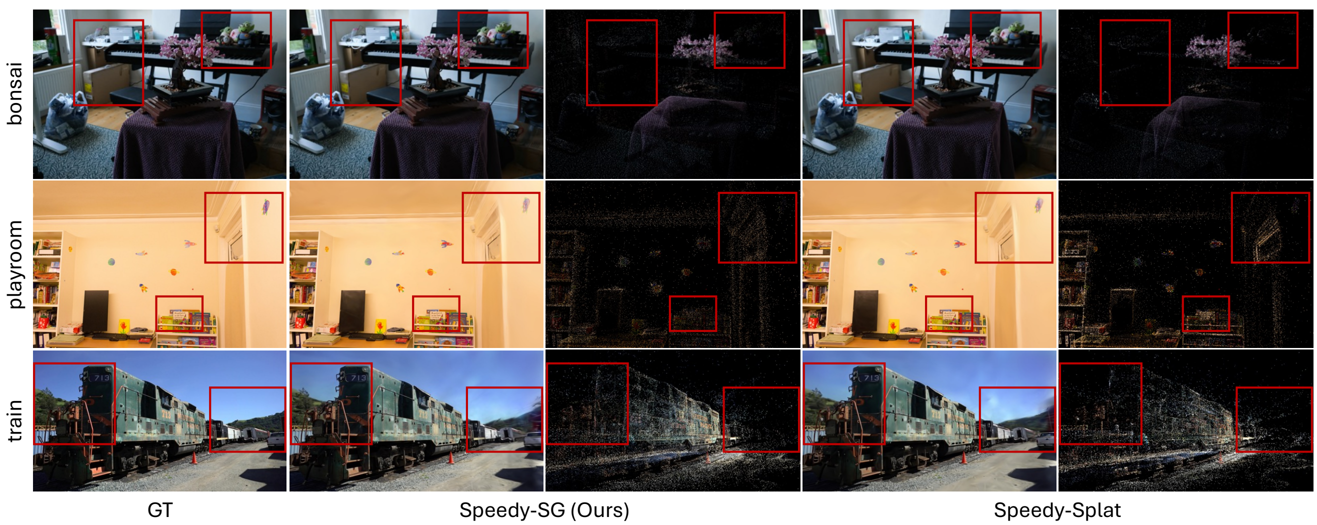


Figure 26. Point visualization of Speedy-SG and Speedy-Splat [16]