Liberated-GS: 3D Gaussian Splatting Independent from SfM Point Clouds

Supplementary Material

In this supplementary material, we will introduce more implementation details (Sec. 6), algorithm derivation and details (Sec. 7), experiment details (Sec. 8), additional evaluation results (Sec. 9) and additional ablation studies (Sec. 10).

6. Implementation Details

6.1. Details of Unbiased Depth Rendering

As described in Sec. 3.2 and illustrated in Figs. 4 and 5, using alpha-blending to render depth in under-optimized 3DGS model results in significant noise in the depth map. This occurs because the centers of 3D Gaussians deviate far from the ray. Inspired by NeRF [28], we assume the 3D point contributing to the rendered depth should lie on the current ray. To simplify computation, we approximate this point as the location where the opacity contribution α reaches its maximum as the ray passes through the Gaussian $\mathcal{G}$. Since $\alpha = \sigma \cdot \mathcal{G}$, it follows that $\arg\max(\alpha) = \arg\max(\mathcal{G})$.

Given a ray $\boldsymbol{r}_w = \boldsymbol{o}_w + t\boldsymbol{d}_w$ in the world space and a Gaussian ellipsoid $\mathcal{G}(\boldsymbol{x}|\boldsymbol{u}, \boldsymbol{S}, \boldsymbol{q})$, where $\boldsymbol{u}$ denotes that center, $\boldsymbol{S}$ the scaling, and $\boldsymbol{q} = (q_r, q_i, q_j, q_k)$ the quaternion rotation with its corresponding rotation matrix $\boldsymbol{R}$ given by

$$\boldsymbol{R}(\boldsymbol{q}) = 2\begin{pmatrix} \frac{1}{2} - (q_j^2 + q_k^2) & (q_i q_j - q_r q_k) & (q_i q_k + q_r q_j) \\ (q_i q_j + q_r q_k) & \frac{1}{2} - (q_i^2 + q_k^2) & (q_j q_k - q_r q_i) \\ (q_i q_k - q_r q_j) & (q_j q_k + q_r q_i) & \frac{1}{2} - (q_i^2 + q_j^2) \end{pmatrix}. \tag{10}$$

We first transform the ray $\boldsymbol{r}_w$ into the Gaussian ellipsoid coordinate system as:

$$\boldsymbol{r}_g = \boldsymbol{o}_g + t\boldsymbol{d}_g \tag{11}$$

$$\boldsymbol{o}_g = \boldsymbol{S}^{-1}\boldsymbol{R}(\boldsymbol{q})(\boldsymbol{o}_w - \boldsymbol{u}) = (o_x, o_y, o_z) \tag{12}$$

$$\boldsymbol{d}_g = \boldsymbol{S}^{-1}\boldsymbol{R}(\boldsymbol{q})\boldsymbol{d}_w = (d_x, d_y, d_z), \tag{13}$$

with Gaussian $\mathcal{G}'$ is expressed as $\mathcal{G}'(\boldsymbol{x}) = \exp(-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{x})$, therefore $\arg\max(\mathcal{G}') = \arg\min(\boldsymbol{x}^T\boldsymbol{x})$. Substituting the parametric representation of ray $\boldsymbol{r}_g$, we have:

$$\begin{aligned} t^* &= \arg\min_t [(o_x + td_x)^2 + (o_y + td_y)^2 + (o_z + td_z)^2] \\ &= \arg\min_t [(d_x^2 + d_y^2 + d_z^2)t^2 + \\ &\qquad 2(o_x d_x + o_y d_y + o_z d_z)t + (o_x^2 + o_y^2 + o_z^2)] \\ &= -\frac{o_x d_x + o_y d_y + o_z d_z}{d_x^2 + d_y^2 + d_z^2} = -\frac{\boldsymbol{o}_g^T\boldsymbol{d}_g}{\boldsymbol{d}_g^T\boldsymbol{d}_g} \end{aligned} \tag{14}$$

To indicates whether ray $\boldsymbol{r}_g$ intersects with Gaussian $\mathcal{G}'$, we have the following function:

$$\begin{aligned} &\mathcal{G}'(\boldsymbol{r}_g(t)) = \exp(-\frac{1}{2}\boldsymbol{r}_g(t)^T\boldsymbol{r}_g(t)) = 1 \\ \Rightarrow\ &\boldsymbol{r}_g(t)^T\boldsymbol{r}_g(t) = 0 \\ \Rightarrow\ &(d_x^2 + d_y^2 + d_z^2)t^2 + \\ &\quad 2(o_x d_x + o_y d_y + o_z d_z)t + (o_x^2 + o_y^2 + o_z^2) - 1 = 0 \end{aligned} \tag{15}$$

with discriminant $\Delta = B^2 - AC >= 0$, where

$$\begin{aligned} A &= d_x^2 + d_y^2 + d_z^2 = \boldsymbol{d}_g^T\boldsymbol{d}_g \\ B &= o_x d_x + o_y d_y + o_z d_z = \boldsymbol{o}_g^T\boldsymbol{d}_g \\ C &= o_x^2 + o_y^2 + o_z^2 - 1 = \boldsymbol{o}_g^T\boldsymbol{o}_g - 1 \end{aligned} \tag{16}$$

We summarize the details in Algorithm 1.

Algorithm 1 Unbiased Gaussian Depth Algorithm

Input:
i: Gaussian sorted id
μ, S, q: Gaussian mean, scaling, and quaternion of rotation matrix
o, R, K: camera position, rotation matrix, and intrinsic matrix
x: pixel coordinate
Output: t_i: Gaussian depth
$d \leftarrow \text{GetRayDir}(x, K, R, o)$ ▷ Ray direction in world space
$R_g \leftarrow \text{QuaternionToRotation}(q)$ ▷ Gaussian rotation matrix
$o \leftarrow S^{-1}R_g(o - \mu),\ d \leftarrow S^{-1}R_g d$
▷ Transform ray to Gaussian space
$A \leftarrow d^T d,\ B \leftarrow o^T d,\ C \leftarrow o^T o - 1$
$\Delta \leftarrow B^2 - AC$
if $\Delta \geq 0$ **then**
  $t_i \leftarrow -B/A$ ▷ Intersect
else
  $t_i \leftarrow 0$ ▷ Not Intersect
end

6.2. Details of Initialize 3D Gaussian from Depth

In our proposed initialization process, each newly added pixel $\boldsymbol{u}$ generates a new 3D Gaussian in world space based on its corresponding ensembled depth $d(\boldsymbol{u})$. Similar to how the original 3D Gaussian Splatting (3DGS) initializes Gaussians from point clouds, we define these Gaussians as spheres with a radius r and center $\boldsymbol{o} + d(\boldsymbol{u}) \cdot dir(d(\boldsymbol{u}))$. The color of each Gaussian is set using which of the corresponding pixels. However, unlike 3DGS, which determines the radius r as the minimum distance between the current point with its nearest neighbors, *i.e.*,

$$r = \min(||\boldsymbol{p}_i - \boldsymbol{p}_j||_2),\ \boldsymbol{p}_j \in \mathcal{P} \text{ and } j \neq i, \tag{17}$$

this approach is incompatible with our progressive strategy. Since our 3D Gaussians are derived by per-pixel projection, we consider the most extreme case involves the distance to the adjacent ray, given by

$$r(\boldsymbol{u}) = d(\boldsymbol{u}) \cdot \sin(\theta), \tag{18}$$

where θ represents the angle between rays emitted by two neighboring pixels. Furthermore, as our goal is to refine depth, all Gaussians are initialized as fully opaque, ensuring that newly added Gaussians do not occlude those corresponding to existing pixels after optimization.

As mentioned in Sec. 3.3, for local frames, we only back-project the points from under-reconstruction areas. Given a pixel $\boldsymbol{u}$ with its rendered depth $d_{\text{render}}(\boldsymbol{u})$, if $d_{\text{render}}(\boldsymbol{u}) < \tau$, it indicates that the emitted ray does not hit any 3D Gaussian. In this case, pixel $\boldsymbol{u}$ is marked as under-reconstructed and included in the subsequent back-projection process. The predefined threshold τ is set to 10^{-3} in our experiments.

6.3. Details of Global Gaussian Adjustment in Progressive Initialization

As stated in Sec. 3.2, the scale-consistency assumption for monocular depth estimation can easily fail in scenarios with multiple objects, significant scale variations, or complex geometries. Therefore, we learn a separate scale and shift parameter for each detected object from [47].

During the initialization phase, we define the center $\boldsymbol{u}$ of all Gaussians as a function of the ensembled depth $d(\boldsymbol{u})$, which is further refined through gradient backpropagation to improve the depth prior. Meanwhile, the scale consistency within each object is maintained to preserve the integrity of the aligned depth while adjusting the 3D Gaussian's position. Unlike the rendered depth which is optimized directly, we retain the rescaling process of the aligned depth. By optimizing the scale and offset parameters for each detected object, the aligned depth is indirectly adjusted, effectively suppressing the occurrence of floaters.

7. Progressive Segmented Initialization Algorithm

The algorithm constructing initial point clouds from estimated monocular depths is summarized in Algorithm 2. We omit the usage of camera poses, intrinsic matrices, and semantic masks for simplification.

Algorithm 2 Initial Point Cloud Construction

Input:
$\mathcal{I} = \{I_t | t = 1...N\}$: Image sequence
$\mathcal{D}_e = \{D_{e_t} | t = 1...N\}$: Estimated depths
Output: $\mathcal{S}$: Initial point cloud
$\mathcal{G}_c \leftarrow$ CoarseGaussin$(\mathcal{I})$ ▷ Coarse Gaussian model with random initialization
$\mathcal{D}_r \leftarrow$ UnbiasedRendering$(\mathcal{G}_c, \mathcal{I})$ ▷ Rendered depth
$\mathcal{D}'_e \leftarrow$ AlignDepth$(\mathcal{D}_e, \mathcal{D}_r)$ ▷ Aligned depth
$\mathcal{D} \leftarrow$ ErosionAndEnsemble$(\mathcal{D}'_e, \mathcal{D}_r)$ ▷ Depth erosion and ensemble
$\mathcal{I}_k = \{I_{k_t} | t = 1...K\} \leftarrow$ GetKeyframe$(\mathcal{I})$ ▷ KeyFrame
$\mathcal{G}_g \leftarrow$ InitGaussian(I_1, D_1) ▷ Init global Gaussian with all pixels
for $i \leftarrow 2$ **to** K **do**
  $\mathcal{G}_g \leftarrow$ BackProject(I_{k_i}, D_{k_i}) ▷ Update global Gaussian with all pixels
  $\mathcal{I}' \leftarrow \{I_1, ..., I_{k_i}\}$, $\mathcal{D}' \leftarrow \{D_1, ..., D_{k_i}\}$
  $\mathcal{G}'_g, \mathcal{D}' \leftarrow$ RefineDepth$(\mathcal{G}_g, \mathcal{I}', \mathcal{D}')$ ▷ Update global Gaussian
  $\mathcal{I}_l \leftarrow \{I_{k_{i-1}+1}, ..., I_{k_i-1}\}$ ▷ LocalFrame
  $\mathcal{G}_l \leftarrow$ BackProject$(I_{k_{i-1}}, I_{k_i}, D'_{k_{i-1}}, D'_{k_i})$ ▷ Init local Gaussian with two nearby keyframes
  for $I_j \in \mathcal{I}_l$ **do**
    $D'_{r_j} \leftarrow$ UnbiasedRendering$(\mathcal{G}_l, I_j)$
    $M_j \leftarrow D'_{r_j} < \tau$ ▷ Get unseen area
    $\mathcal{G}_l \leftarrow$ BackProject(I_j, D_j, M_j) ▷ Update local Gaussian from unseen area
  end
  $\mathcal{G}_g \leftarrow \mathcal{G}_l$ ▷ Update global Gaussian with local Gaussian
end
$\mathcal{S} \leftarrow$ ImportanceResampling$(\mathcal{G}_g)$

8. Experiment Details

In the main paper experiments, we compare three methods that do not rely on SfM point clouds for initialization. The results of RAIN-GS [18] are directly adopted from the original paper, while the other two are trained under the new experimental setup.

3DGS [20] with random points. Instead of loading 3D points and corresponding colors from the SfM point cloud, we randomly sample $N = 100\_000$ points. Such random sampling is performed within the bounding box determined by the training images. Random colors are then assigned to these points. All other experiment settings remain unchanged.

Colmap-Free 3DGS (CFGS) [14] with ground truth poses. CFGS proposes a method for reconstructing 3DGS from a sequence of images and corresponding monocular depth maps, without requiring poses or initial point clouds. To align with our experimental setup, we set the initial poses of CFGS as ground truth poses derived from COLMAP reconstruction and disable all pose-related optimization terms and gradient back-propagation. For scale ambiguity between estimated monocular depths and ground-truth poses, we compare the depths with SfM points to solve it, simi-

Table 6. Per-scene quantitative results from the Mip-NeRF360 and Tanks&Temples dataset.

PSNR↑

	Mip-NeRF360									Tanks&Temples	
	Garden	Stump	Treehill	Bicycle	Flowers	Counter	Kitchen	Bonsai	Room	Train	Truck
3DGS	27.410	26.550	22.490	25.246	21.520	28.700	30.317	31.980	30.632	21.097	25.187
3DGS (Random)	23.217	20.745	18.986	21.034	17.815	23.608	26.078	18.538	29.685	20.730	20.236
Colmap-Free 3DGS*	26.158	18.200	17.620	16.458	14.286	24.563	22.910	25.406	27.544	20.551	20.858
RAIN-GS	26.884	26.680	22.528	25.042	21.762	28.529	31.270	21.547	30.809	21.436	24.816
Ours	27.289	26.894	22.117	25.476	21.508	29.122	31.524	32.646	31.720	22.043	25.128

SSIM↑

	Mip-NeRF360									Tanks&Temples	
	Garden	Stump	Treehill	Bicycle	Flowers	Counter	Kitchen	Bonsai	Room	Train	Truck
3DGS	0.868	0.775	0.638	0.771	0.605	0.905	0.922	0.938	0.914	0.802	0.879
3DGS (Random)	0.783	0.618	0.550	0.575	0.469	0.833	0.893	0.719	0.894	0.771	0.758
Colmap-Free 3DGS*	0.848	0.305	0.318	0.286	0.381	0.797	0.824	0.839	0.840	0.694	0.728
RAIN-GS	0.854	0.768	0.621	0.747	0.616	0.895	0.920	0.934	0.906	0.786	0.865
Ours	0.866	0.786	0.626	0.785	0.632	0.908	0.929	0.945	0.918	0.824	0.872

LPIPS↓

	Mip-NeRF360									Tanks&Temples	
	Garden	Stump	Treehill	Bicycle	Flowers	Counter	Kitchen	Bonsai	Room	Train	Truck
3DGS	0.103	0.210	0.317	0.205	0.336	0.204	0.129	0.205	0.220	0.218	0.148
3DGS (Random)	0.175	0.345	0.413	0.378	0.403	0.276	0.161	0.401	0.265	0.256	0.249
Colmap-Free 3DGS*	0.106	0.498	0.518	0.534	0.429	0.288	0.195	0.257	0.279	0.339	0.321
RAIN-GS	0.114	0.215	0.342	0.238	0.324	0.223	0.137	0.218	0.247	0.244	0.169
Ours	0.100	0.186	0.273	0.174	0.254	0.183	0.118	0.184	0.210	0.162	0.118

Table 7. Per-scene quantitative results from the OMMO dataset.

PSNR↑

	03	05	06	10	13	14	15
3DGS	25.81	26.19	26.29	28.35	28.27	30.18	29.42
3DGS (Random)	22.69	24.79	25.99	26.31	28.03	28.38	24.81
RAIN-GS	25.02	25.69	26.74	27.36	26.89	28.93	27.89
Ours	27.26	25.60	27.44	28.20	28.43	30.33	28.84

SSIM↑

	03	05	06	10	13	14	15
3DGS	0.866	0.834	0.900	0.860	0.869	0.938	0.927
3DGS (Random)	0.804	0.805	0.891	0.794	0.852	0.909	0.834
RAIN-GS	0.839	0.821	0.914	0.825	0.825	0.918	0.903
Ours	0.894	0.823	0.921	0.860	0.881	0.940	0.921

LPIPS↓

	03	05	06	10	13	14	15
3DGS	0.215	0.243	0.188	0.190	0.213	0.107	0.136
3DGS (Random)	0.273	0.267	0.200	0.272	0.247	0.143	0.200
RAIN-GS	0.247	0.262	0.187	0.249	0.283	0.145	0.153
Ours	0.162	0.211	0.150	0.182	0.182	0.102	0.107

lar to many previous works [9, 32]. For each image, we project SfM points onto the camera view to obtain a set of sparse depths. We solve for the scale and shift parameters for each image using the closed-form linear regression solution Eq. (7) to align the monocular depth with sparse depth. Then, we apply the per-image aligned monocular depths to CFGS and keep all other settings unchanged.

9. Additional Evaluation Results

As described in the main paper, we conduct novel view synthesis experiments on the Mip-NeRF360, Tanks&Temples, and OMMO datasets, comparing our method with 3DGS, random initialized 3DGS, RAIN-GS, and Colmap-Free 3DGS with ground-truth poses. In this section, we provide more detailed quantitative and qualitative results. Tab. 6 and Tab. 7 expand on Tab. 1 by presenting detailed results across individual scenes. As shown in Tab. 6 and Tab. 7, our method demonstrates clear superiority over the other four approaches, particularly in terms of LPIPS, a metric more reflective of human visual perception than PSNR and SSIM. This indicates that our approach effectively leverages the monocular depth priors while mitigating the potential errors they may introduce. Additionally, we observed that our method yields greater improvements in indoor scenes compared to outdoor scenes. Specifically, for indoor scenes, our method surpasses the original 3DGS which relies on SfM point cloud initialization across all metrics. This is because the monocular depth priors are more accurate in indoor environments, providing a stronger foundation for reconstruction.

In Fig. 8 and Fig. 7, we showcase additional comparisons with the original 3DGS and RAIN-GS, which are the second and third best methods, respectively, as shown in Tab. 1. The highlighted areas in the images further emphasize our improvements. Please zoom in for more details.

Figure 7. Qualitative comparison for novel view synthesis on Mip-NeRF360 and Tanks&Temples dataset with RAIN-GS and 3DGS initialized with SfM points (Part I). Each viewpoint contains a rendered RGB image and a depth image. For Ground Truth, we display the monocular depth map estimated from the off-the-shelf network [19]. Our method shows better performance in both reconstruction details and scene structures without relying on additional point clouds.

10. Additional Ablation Studies

Tab. 8 expands on Tab. 2 by presenting the performance of different initialization methods on two other Gaussian models across individual scenes. The results indicate that the sensitivity of 3DGS to initialization is a common issue and that our method is also applicable to more advanced approaches.

In Fig. 9, we provide a more intuitive comparison corresponding to Tab. 3 conducted on the *stump* scene of MipNeRF-360, illustrating the effectiveness of our progressive method. The *base model* represents progressive initialization based solely on scale-consistent estimated depth from keyframes. Subsequent components, including importance resampling, depth alignment, and local Gaussian, are added step by step. The quality of the rendered images is evaluated using the same three metrics as in the main paper, *i.e.*, PSNR, SSIM, and LPIPS, displayed in the top-left corner of each image. As each component is incorporated, the rendering quality improves progressively, highlighting the effectiveness of each component in our method.

In addition to the three ablation experiments presented in the main paper, the following subsections further discuss our proposed progressive method and unbiased depth rendering. We also provide an experiment to explain why depth loss is not utilized in our approach.

10.1. Ablation on Unbiased Depth Rendering

To validate the effectiveness of our proposed unbiased depth rendering, we replace it with the original alpha-blending approach and compare the results, as shown in Tab. 10 and the second column of Fig. 10. It can be observed that alpha-blending consistently underperforms across all datasets on the three evaluation metrics compared to our unbiased depth rendering. As illustrated, the initial point cloud generated with alpha-blending contains numerous points with incorrect scales (*e.g.*, 3D points below the ground) and lacks accurate geometric priors (*e.g.*, distant tree trunks). These issues result in the final rendered depth maps with significant holes. In contrast, our method effectively addresses these problems, producing more accurate and complete depth maps.

10.2. Ablation on Progressive Initialization

One of our primary motivations for adopting the progressive strategy is to reduce redundancy in 3D points. To validate its effectiveness, we conduct comparative experiments on Mip-NeRF360 with two other point cloud downsampling approaches. (1) The first method is derived from the downsampling strategy mentioned in mini-splatting [11]. It selects a fixed total number of points for each scene, empirically set to around 3.5 million. Then, a random subset of pixels from each image is chosen to generate the initial

Table 8. Per-scene results applying our initialization method to Mini-Splatting and 3DGS-MCMC on the OMMO dataset.

PSNR↑

	03	05	06	10	13	14	15
Mini-Splatting (SfM)	25.18	25.66	25.62	27.87	26.51	29.02	28.20
Mini-Splatting (Random)	24.50	24.20	25.35	26.15	26.26	27.91	24.48
Mini-Splatting (Ours)	26.45	25.55	26.53	27.59	26.41	29.75	28.76
3DGS-MCMC (SfM)	27.55	26.65	27.17	29.70	27.83	30.30	29.15
3DGS-MCMC (Random)	27.37	26.55	27.28	29.23	25.18	29.90	28.73
3DGS-MCMC (Ours)	27.60	26.54	27.41	29.85	28.93	30.23	28.95

SSIM↑

	03	05	06	10	13	14	15
Mini-Splatting (SfM)	0.844	0.823	0.887	0.834	0.813	0.924	0.912
Mini-Splatting (Random)	0.826	0.794	0.880	0.774	0.807	0.904	0.829
Mini-Splatting (Ours)	0.880	0.820	0.921	0.850	0.807	0.938	0.922
3DGS-MCMC (SfM)	0.897	0.843	0.936	0.878	0.864	0.942	0.929
3DGS-MCMC (Random)	0.894	0.841	0.936	0.864	0.818	0.938	0.916
3DGS-MCMC (Ours)	0.899	0.843	0.938	0.897	0.886	0.939	0.926

LPIPS↓

	03	05	06	10	13	14	15
Mini-Splatting (SfM)	0.243	0.235	0.202	0.221	0.280	0.129	0.128
Mini-Splatting (Random)	0.260	0.273	0.214	0.302	0.290	0.148	0.211
Mini-Splatting (Ours)	0.192	0.190	0.147	0.196	0.290	0.101	0.105
3DGS-MCMC (SfM)	0.177	0.242	0.126	0.166	0.224	0.103	0.101
3DGS-MCMC (Random)	0.180	0.234	0.131	0.188	0.295	0.108	0.122
3DGS-MCMC (Ours)	0.161	0.224	0.122	0.164	0.185	0.105	0.108

points with corresponding depth. (2) The second method is the commonly used voxel downsampling. With this approach, we initialize the voxel size to 0.05 and set the maximum number of points to 3.5 million, matching the configuration of mini-splatting. If the number of downsampled points exceeds this limit, the voxel size is doubled, and the downsampling process is repeated.

The relevant results are presented in Tab. 9 and the third and fourth columns of Fig. 10. It can be observed that simple random downsampling, whether image-wise or voxel-wise, fails to eliminate the numerous misaligned 3D points caused by depth errors. This results in the inability to produce photo-realistic rendered images. Across all test scenes, the three evaluation metrics consistently demonstrate inferior performance compared to our progressive strategy.

10.3. Ablation on Depth Loss

Depth-related loss has been widely used in previous works. To evaluate its effectiveness, we incorporated it into the refinement stage. Based on our proposed ensembled depth D and edge-aware valid mask M, the depth loss is defined as:

$$\mathcal{L}_{\hat{D}} = \frac{1}{|M|} \sum \log(1 + M \odot ||D - \hat{D}||_1), \tag{19}$$

where $|M|$ indicates the total number of pixels with valid depth, and $\hat{D}$ denotes the unbiased rendered depth during the refinement stage. Therefore, the overall objective function is then defined as:

$$\mathcal{L} = \mathcal{L}_c + \lambda_d \mathcal{L}_{\hat{D}}, \tag{20}$$

where $\mathcal{L}_c$ is the original photometric loss proposed in 3DGS and λ_d is set to 0.2 in our experiments.

The experimental results, as presented in Tab. 11, indicate that the depth loss does not play a significant role. In fact, it even performs worse overall compared to using the photometric loss alone.

Figure 8. Qualitative comparison for novel view synthesis on Mip-NeRF360 and Tanks&Temples dataset with RAIN-GS and 3DGS initialized with SfM points (Part II). Each viewpoint contains a rendered RGB image and a depth image. For Ground Truth, we display the monocular depth map estimated from the off-the-shelf network [19]. Our method shows better performance in both reconstruction details and scene structures without relying on additional point clouds.

Table 9. Ablation study on Mip-NeRF360 comparing different point cloud downsample methods with the proposed progressive segmented method to reduce point redundancy. The best score is highlighted in bold.

PSNR↑

	Garden	Stump	Treehill	Bicycle	Flowers	Counter	Kitchen	Bonsai	Room	Avg
Random Sample	26.953	24.194	21.629	24.990	20.961	28.937	31.254	30.602	30.832	26.706
Voxel Downsample	26.841	24.091	21.327	24.939	20.657	28.861	30.716	30.368	31.131	26.548
Ours	**27.289**	**26.894**	**22.117**	**25.476**	**21.508**	**29.122**	**31.524**	**32.646**	**31.720**	**27.588**

SSIM↑

	Garden	Stump	Treehill	Bicycle	Flowers	Counter	Kitchen	Bonsai	Room	Avg
Random Sample	0.857	0.698	0.612	0.769	0.610	0.907	0.926	0.922	0.916	0.802
Voxel Downsample	0.854	0.701	0.614	0.764	0.596	0.904	0.922	0.915	0.914	0.798
Ours	**0.866**	**0.786**	**0.626**	**0.785**	**0.632**	**0.908**	**0.929**	**0.945**	**0.918**	**0.822**

LPIPS↓

	Garden	Stump	Treehill	Bicycle	Flowers	Counter	Kitchen	Bonsai	Room	Avg
Random Sample	0.103	0.245	0.280	0.181	0.268	**0.183**	0.121	0.202	**0.209**	0.199
Voxel Downsample	0.109	0.245	0.284	0.200	0.278	0.198	0.128	0.212	0.220	0.208
Ours	**0.100**	**0.186**	**0.273**	**0.174**	**0.254**	**0.183**	**0.118**	**0.184**	0.210	**0.187**

Table 10. Ablation study on Mip-NeRF360 comparing alpha-blending with the proposed unbiased depth rendering method to align the estimated depth. The best score is highlighted in bold.

PSNR↑

	Garden	Stump	Treehill	Bicycle	Flowers	Counter	Kitchen	Bonsai	Room	Avg
Alpha-Blending	27.149	26.392	22.110	25.211	21.375	28.987	31.286	32.461	31.482	27.384
Ours	**27.289**	**26.894**	**22.117**	**25.476**	**21.508**	**29.122**	**31.524**	**32.646**	**31.720**	**27.588**

SSIM↑

	Garden	Stump	Treehill	Bicycle	Flowers	Counter	Kitchen	Bonsai	Room	Avg
Alpha-Blending	0.863	0.772	0.619	0.768	0.627	0.905	0.925	0.943	0.914	0.815
Ours	**0.866**	**0.786**	**0.626**	**0.785**	**0.632**	**0.908**	**0.929**	**0.945**	**0.918**	**0.822**

LPIPS↓

	Garden	Stump	Treehill	Bicycle	Flowers	Counter	Kitchen	Bonsai	Room	Avg
Alpha-Blending	0.105	0.199	0.299	0.199	0.265	0.198	0.126	0.193	0.225	0.201
Ours	**0.100**	**0.186**	**0.273**	**0.174**	**0.254**	**0.183**	**0.118**	**0.184**	**0.210**	**0.187**

Table 11. Ablation study on Mip-NeRF360 for depth loss. The best score is highlighted in bold.

PSNR↑

	Garden	Stump	Treehill	Bicycle	Flowers	Counter	Kitchen	Bonsai	Room	Avg
w. depth loss	**27.326**	26.863	**22.149**	**25.478**	21.457	29.075	**31.651**	32.569	31.512	27.564
Ours	27.289	**26.894**	22.117	25.476	**21.508**	**29.122**	31.524	**32.646**	**31.720**	**27.588**

SSIM↑

	Garden	Stump	Treehill	Bicycle	Flowers	Counter	Kitchen	Bonsai	Room	Avg
w. depth loss	0.865	0.784	**0.627**	0.784	0.630	0.907	0.927	**0.945**	0.917	0.821
Ours	**0.866**	**0.786**	0.626	**0.785**	**0.632**	**0.908**	**0.929**	**0.945**	**0.918**	**0.822**

LPIPS↓

	Garden	Stump	Treehill	Bicycle	Flowers	Counter	Kitchen	Bonsai	Room	Avg
w. depth loss	**0.100**	0.188	0.275	0.175	0.257	0.185	0.120	0.185	0.212	0.188
Ours	**0.100**	**0.186**	**0.273**	**0.174**	**0.254**	**0.183**	**0.118**	**0.184**	**0.210**	**0.187**

Figure 9. Visual comparison with different components of the proposed progressive strategy. Zoom in for more details.

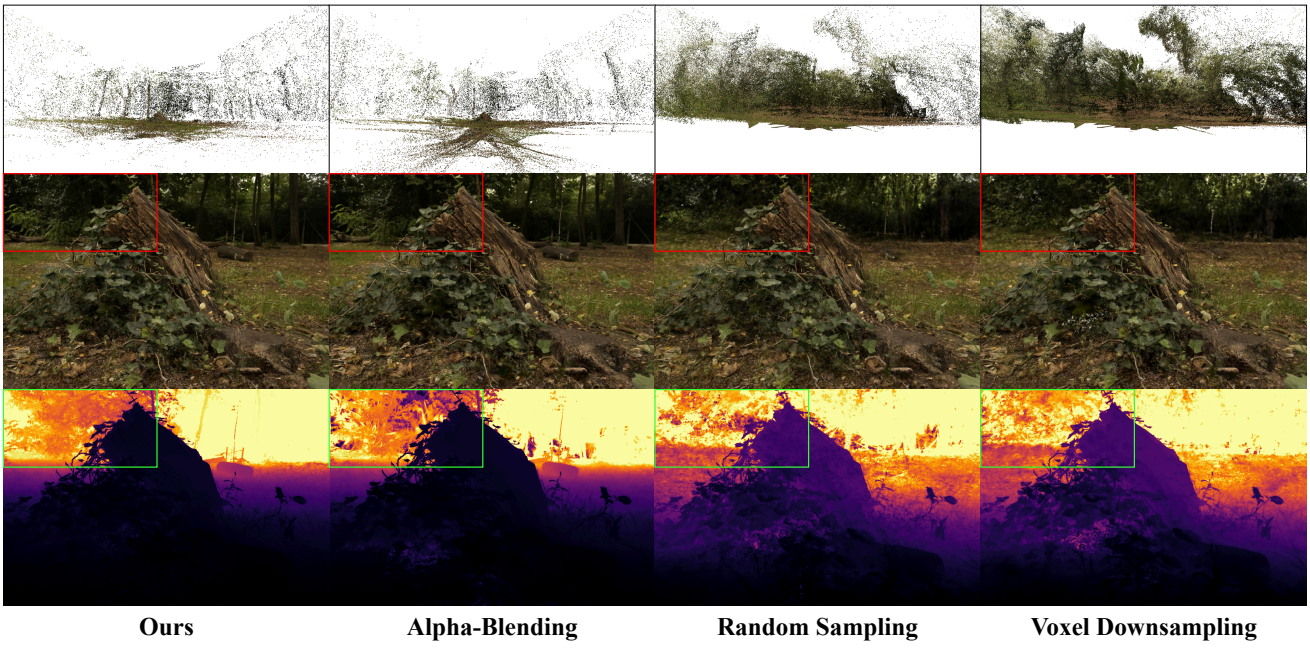

Figure 10. Visual comparison with different point cloud downsample strategy and depth rendering method for the initialization process. The 1^{st} row shows the point cloud generated with different setups for refinement. The 2^{nd} and 3^{rd} rows are rendered RGB images and rendered depths, respectively.