Outdoor Monocular SLAM with Global Scale-Consistent 3D Gaussian Pointmaps

Supplementary Material

6. Overview

This supplementary material provides implementation details for keyframe management, patch-based scale alignment, pointmap replacement, and Gaussian map optimization modules. We also include additional experiments on KITTI, with runtime, memory, and patch size analysis. Furthermore, we present extra qualitative results on three datasets, including tracking trajectory and novel view synthesis. Finally, we discuss limitations and future work.

7. Implementation Details

7.1. Keyframe Management

As described in Section 3.3.3, we joinly refine camera poses and the Gaussian map within a local keyframe window $\mathcal{W}$. A well-designed keyframe selection strategy must ensure sufficient viewpoint coverage while avoiding redundancy. Given the computational cost of jointly optimizing the Gaussian scene and camera pose across all keyframes, we maintain a local keyframe window $\mathcal{W}$ to select nonredundant keyframes that observe overlapping areas of the scene. This approach provides better multi-view constraints for subsequent Gaussian map optimization. With this in mind, we adopt the keyframe management approach from [22], where keyframes are selected based on covisibility, and the local window is managed by assessing the overlap with the latest keyframe.

Specifically, we define the covisibility and overlap between keyframes i and j using Intersection over Union (IOU) and Overlap Coefficient (OC) [22]:

$$IOU_{\text{cov}}(i,j) = \frac{\left|\mathcal{G}_i^v \cap \mathcal{G}_j^v\right|}{\left|\mathcal{G}_i^v \cup \mathcal{G}_j^v\right|}, \tag{18}$$

$$OC_{\text{cov}}(i,j) = \frac{\left|\mathcal{G}_i^v \cap \mathcal{G}_j^v\right|}{\min\left(\left|\mathcal{G}_i^v\right|, \left|\mathcal{G}_j^v\right|\right)}, \tag{19}$$

where $\mathcal{G}_i^v$ is the set of visible Gaussians in keyframe i.

Given the latest keyframe j, keyframe i is added to the keyframe window $\mathcal{W}$ if: $IOU_{\text{cov}}(i,j) < k_I$ or the relative pose translation distance $d_{ij} > k_d \hat{D}_i$, where $\hat{D}_i$ represents the median pointmap depth of frame i. Given the newly added keyframe i', we remove keyframe l from the window if: $OC\text{cov}(i',l) < ko$. If the number of keyframes in the window $\mathcal{W}$ exceeds the maximum size, we remove the keyframe with the lowest OC value relative to i'.

For all experiments on three datasets, we set the keyframe management parameters as $k_I = 0.9, k_d = 0.08, k_o = 0.3$, with the keyframe window size set to $|\mathcal{W}| = 8$.

7.2. Patch-based Scale Alignment

As described in Section Section 3.3.1, we align the scale of the pretrained pointmap X^p to Gaussian scene, using the 3DGS pointmap X^r as reference. We propose a rigorous and detailed patch-based method to select highly reliable "correct points" and use them to calculate the scale factor. The detailed procedure is described in the following Algorithm 1:

Algorithm 1 Patch-based Pointmap Scale Alignment

```
1: procedure ALIGN($X^r, X^p, P, \delta_\mu, \delta_\sigma, \epsilon_r$, max_iter)
2:     $\sigma' \leftarrow 1$
3:     $X_1^p \leftarrow X^p$
4:     for iter $= 1$ to max_iter do
5:         Segment $X^r$ and $X_{iter}^p$ into $P \times P$ patches
6:         for each patch in $X^r, X_{iter}^p$ do
7:             $\mu_r, \sigma_r \leftarrow$ mean$(X^r)$, std$(X^r)$
8:             $\mu_p, \sigma_p \leftarrow$ mean$(X_{iter}^p)$, std$(X_{iter}^p)$
9:             if $|\mu_r - \mu_p| < \delta_\mu \cdot \mu_p \wedge |\sigma_r - \sigma_p| < \delta_\sigma \cdot \sigma_p$ then
10:                Add patch to candidates
11:            end if
12:        end for
13:        for each patch in candidates do
14:            $X_N^r, X_N^p \leftarrow \frac{X^r - \mu_r}{\sigma_r}, \frac{X^p - \mu_p}{\sigma_p}$
15:            for each $x$ in patch do
16:                if $|X_N^r(x) - X_N^p(x)| < \epsilon_r$ then
17:                    Add $x$ to $CP$
18:                end if
19:            end for
20:            if $CP$ is not empty then
21:                $\sigma' \leftarrow \frac{\mu(X^r[CP])}{\mu(X^p[CP])}$
22:            end if
23:        end for
24:        $X_{\text{iter}+1}^p \leftarrow \sigma' \cdot X^p$
25:    end for
26:    return $\hat{X}^p = \sigma' \cdot X^p$
27: end procedure
```

If the number of "correct points" is insufficient, i.e., $|CP| < \tau N_p$, where N_p represents the number of points in the pointmap, we apply a scale remedy strategy. Specifically, we use the fast NN algorithm [19] to establish matching points MP between the current frame X_n^p and the adja-

Parameter	P	δ_μ	δ_σ	ϵ_r	max_iter	τ
Value	10	0.3	0.3	0.1	3	0.01

Table 6. **Hyperparameters for Patch-Based Scale Alignment on three datasets.**

cent keyframe aligned pointmap $\hat{X}^p_{ak}$. Since $\hat{X}^p_{ak}$ is already aligned with the scene scale, it serves as a reference to calculate the scale factor for X^p_n:

$$\sigma' \leftarrow \frac{\mu(\hat{X}^p_{ak}[MP])}{\mu(X^p_n[MP])}. \tag{20}$$

We believe that our carefully designed Algorithm 1 provides the most reliable scale factor. Therefore, we first compute $X^p_{\text{max\_iter}+1} \leftarrow \sigma' \cdot X^p$ and then perform an additional iteration of the Patch-based Scale Alignment process to obtain a newly estimated scale factor σ''. If the number of "correct points" is sufficient, we adopt this iteration's result as the final output for scale alignment. However, if the number remains insufficient, we use σ'' as a remedial scale factor. Although this is not the ideal scenario, it still provides an adequate scale correction for the pre-trained pointmap X^p, effectively mitigating severe scale drift. Moreover, experiments show that the number of keyframes requiring a remedial scale factor does not exceed three per scene on average.

The parameter selection for the algorithm is shown in Table 6. Across three outdoor datasets, our method operates with the same parameters without requiring additional tuning for different scenes, demonstrating its robustness and generalizability.

7.3. Pointmap Supervision

To avoid inserting Gaussians at incorrect positions at keyframes, we replace "incorrect points" in the rendered pointmap X^r with the aligned pretrained pointmap $\hat{X}^p$, as shown in Section 3.3.2. For all three datasets, we set $\epsilon_m = 0.15$ to replace points with significant discrepancies.

Notably, when camera viewpoint changes are relatively mild and the Gaussian scene within view remains largely complete (e.g., in straight-line trajectories), the proportion of replaced points is around 10%, ensuring consistent scale for newly inserted Gaussians. In contrast, when viewpoint changes are large and the Gaussian scene has deficiencies (e.g., during sharp turns), the replacement ratio increases to 30%-50%. In such cases, the priority is to prevent inserting outlier Gaussians, while the aligned pre-trained pointmap is sufficient to maintain scale consistency. This demonstrates the dynamic adaptability of our method to complex environments. Additionally, we incorporate the Gaussian pruning approach proposed by [22] to remove outlier Gaussians during map optimization.

Method	ATE	PSNR	GPU	Time
DROID-SLAM	1.30	-	11 G	$\sim$ 1.5 min
MASt3R-SLAM	1.35	-	7 G	$\sim$2 min
MonoGS	5.68	13.3	9 G	$\sim$5 min
OpenGS-SLAM	1.41	17.9	9 G	$\sim$10 min
CF-3DGS	5.99	15.9	12 G	$\sim$ 120 min
Splat-SLAM	1.25	19.5	10.5 G	$\sim$36 min
Ours	**0.55**	**20.6**	**9.5 G**	**$\sim$5 min**

Table 7. Added Comparison on KITTI.

Patch Size	5	8	10	12	16	20	25	30
ATE	1.35	0.62	0.55	0.57	0.49	0.69	0.73	0.97
PSNR	18.9	20.1	20.6	20.5	20.8	20.0	20.1	19.5

Table 8. Impact of patch size on KITTI.

7.4. Gaussian Map Optimization

In Section 3.3.3, we optimize the Gaussian map within the keyframe window $\mathcal{W}$. For three datasets, we set $\lambda_{iso} = 10, \alpha = 0.98$. In relatively confined scenes where pointmap values exhibit limited variation, we recommend using a smaller α, such as 0.96.

8. Additional Experiments

We conducted additional experiments on the KITTI-07 sequence, including further comparisons with CF-3DGS [7] , MASt3R-SLAM [19], DROID-SLAM [27], and Splat-SLAM [33]. We also analyzed runtime and memory consumption, and performed an ablation study on the patch size used in Algorithm 1.

8.1. Added Comparison

Tab. 8 shows that CF-3DGS performs significantly worse on the KITTI dataset, while our method achieves notably higher tracking accuracy compared to both DROID-SLAM and MASt3R-SLAM. These improvements stem from our targeted design tailored to the characteristics of outdoor environments and a specific remedy for the scale issues in the MASt3R framework.

8.2. Running Time and Memory

Tab. 8 shows that, compared to other 3DGS-based SLAM methods, our approach achieves higher accuracy while maintaining acceptable runtime and memory consumption. Although Splat-SLAM also achieves competitive novel view synthesis (NVS) accuracy, its extensive global optimization procedures incur significant additional runtime. Note: To ensure high-quality rendering and fair comparison, all the above 3DGS-based methods perform approximately 10 minutes of color refinement after SLAM execution. The reported runtime includes only the full SLAM pipeline, excluding post-processing.

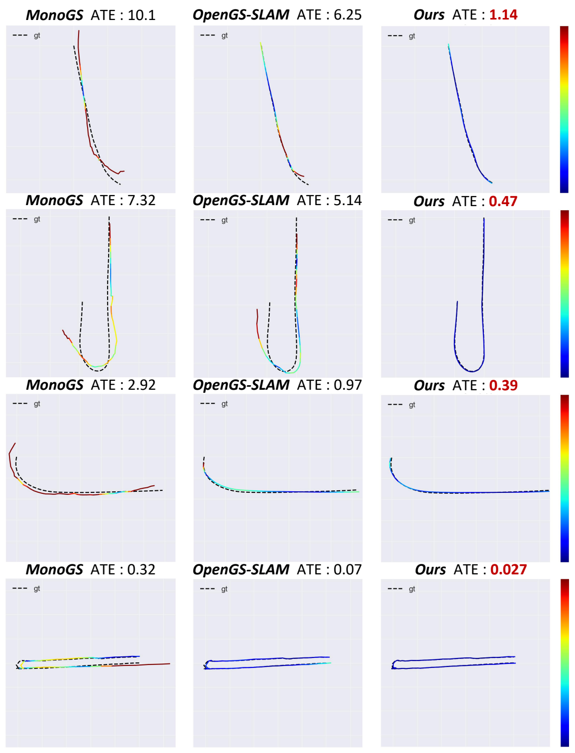

Figure 7. **Comparison of Tracking Trajectories with MonoGS and OpenGS-SLAM.**

8.3. Ablation Study on Patch Size

Tab. 8 shows that a patch size of 10–16 is optimal: larger patches introduce too many outliers, while smaller ones yield noisy statistics.

9. Additional Qualitative Results

Figure 7 presents additional trajectory comparisons, further highlighting the robustness of our method in location under challenging outdoor environments.

Figures 8 to 10 shows additional novel view synthesis results in the Waymo, DL3DV, and KITTI datasets. Clearly, our method produces higher-fidelity images and more accurate depth maps.

10. Limitations and Future Works

1. Our method cannot handle dynamic objects in outdoor scenes. Monocular RGB-only SLAM for outdoor environments with dynamic objects remains a highly interesting and challenging problem.
2. Our method does not incorporate loop closure or global BA. While their inclusion would benefit long-sequence SLAM, it also introduces challenges related to training time and memory consumption.

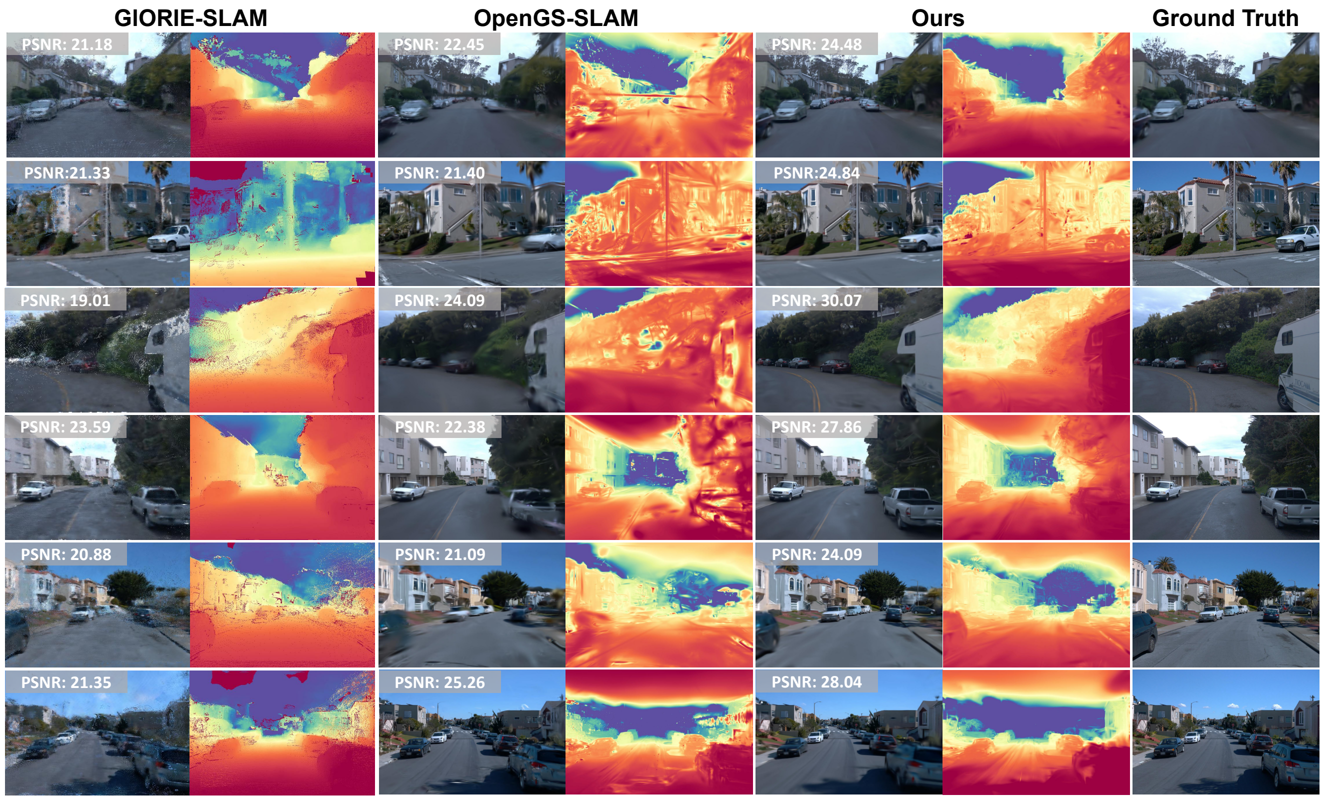

Figure 8. **Novel View Synthesis Results on Waymo, including Rendered RGB and Depth Maps.**

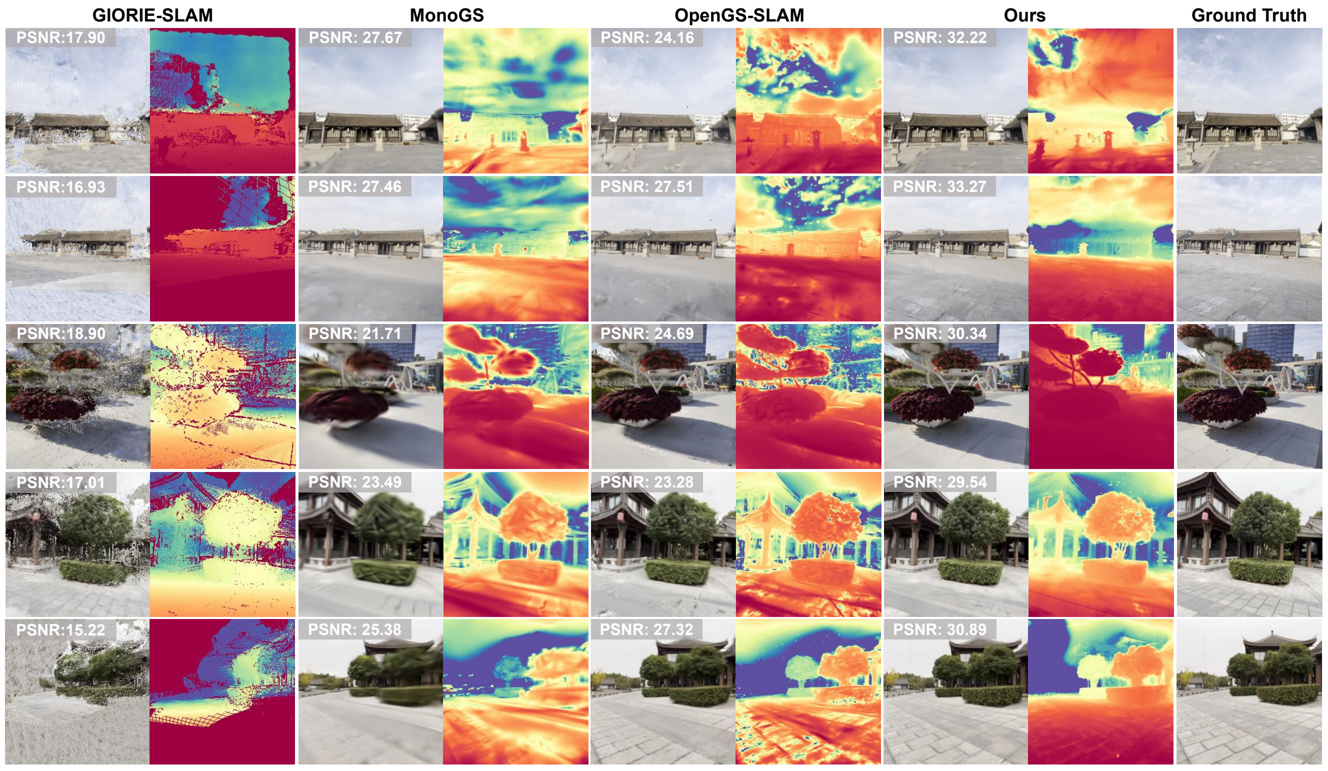

Figure 9. **Novel View Synthesis Results on DL3DV, including rendered RGB and depth maps.**

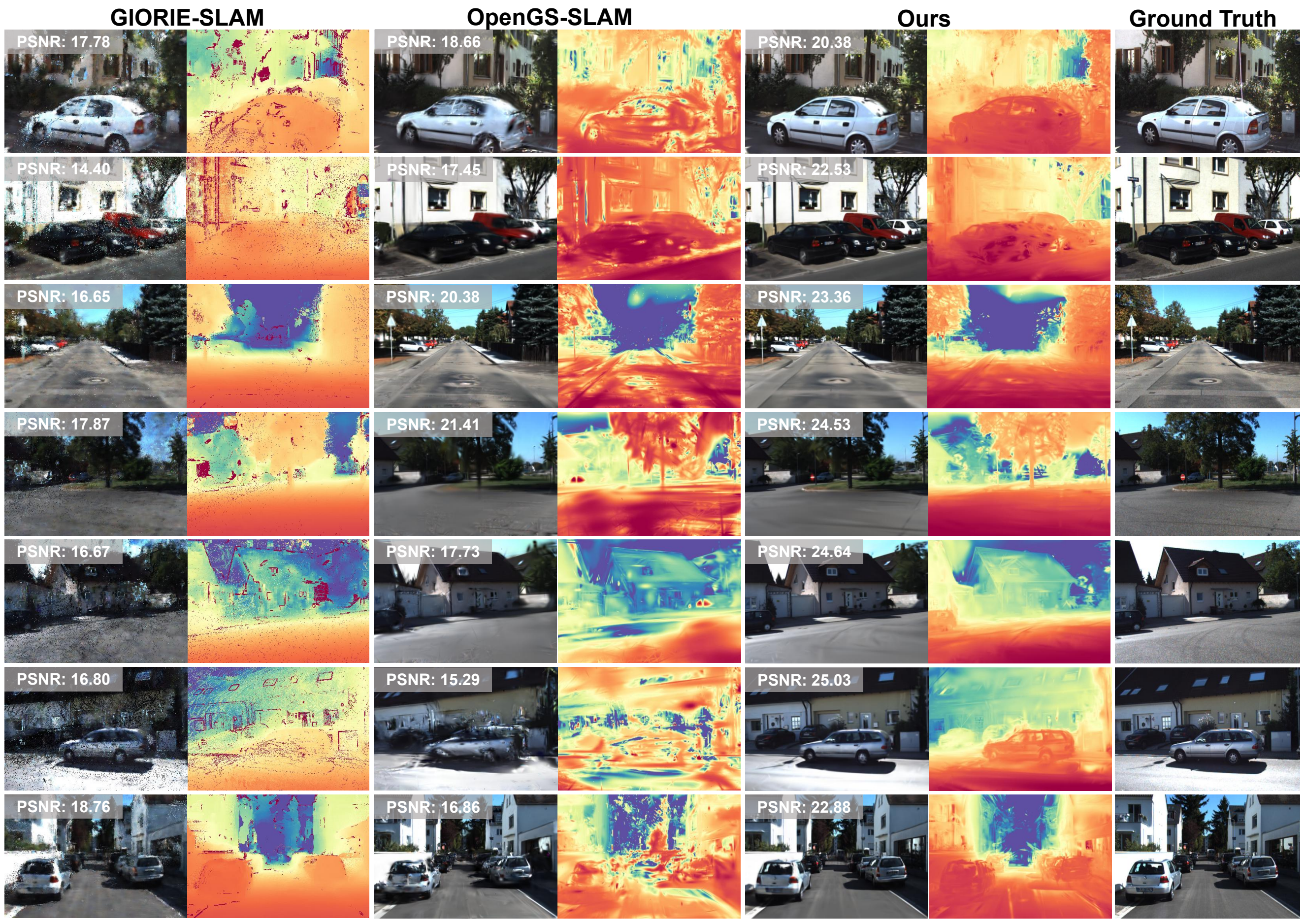

Figure 10. **Novel View Synthesis Results on KITTI, including Rendered RGB and Depth Maps.**