CG-SLAM: Efficient Dense RGB-D SLAM in a Consistent Uncertainty-aware 3D Gaussian Field - Supplementary Material -

Jiarui Hu[1*], Xianhao Chen[2*], Boyin Feng[1], Guanglin Li[1], Liangjing Yang[2], Hujun Bao[1], Guofeng Zhang[1], and Zhaopeng Cui[1†]

[1] State Key Lab of CAD&CG, Zhejiang University
[2] ZJU-UIUC Institute, International Campus, Zhejiang University

In this supplementary material, we provide more details of our CG-SLAM system, including **1)** a comprehensive theoretical analysis of camera pose derivatives in 3D Gaussian Splatting [4] (Sec. A); **2)** further implementation details (Sec. B); **3)** quantitative and qualitative rendering results (Sec. C); **4)** further evaluation of efficiency performance, more ablation results, and comparison to some traditional baselines (Sec. D). Moreover, we also provide a supplementary video to visualize our system framework and experimental results.

A Mathematical Analysis Of Camera Pose Gradients

CUDA rasterization pipeline brings significant efficiency gains, but at the same time, it disabled the Pytorch automatic differentiation mechanism. In this section, we will report the comprehensive mathematical analysis of camera pose gradients in the 3D Gaussian splatting [4] framework to extend this popular algorithm to a broader range of applications.

Preliminary Framework. The 3D Gaussian splatting [4] algorithm applies standard α-blending to render RGB values on each pixel, as depicted in Eq. (1).

$$RGB = \sum_{i=1}^{N} c_i \cdot \alpha_i \cdot T_i \ . \tag{1}$$

In CUDA implementation, the gradient of an image-level loss function w.r.t the pose $\frac{d(L)}{d(pose)}$ is essentially divided into each pixel. If not specified otherwise, by default we define the camera pose as the transpose of the world-to-camera transformation as follows:

$$pose = \begin{bmatrix} W & t \\ \mathbf{0} & \mathbf{1} \end{bmatrix}^T = \begin{bmatrix} v_0 & v_1 & v_2 & 0 \\ v_4 & v_5 & v_6 & 0 \\ v_8 & v_9 & v_{10} & 0 \\ v_{12} & v_{13} & v_{14} & 1 \end{bmatrix} \ . \tag{2}$$

For a given pixel, using the chain rule, the gradient of the rendered RGB values w.r.t pose can be intuitively decomposed into two main parts: $\frac{d(c)}{d(pose)}$, $\frac{d(\alpha)}{d(pose)}$,

*Jiarui Hu and Xianhao Chen contributed equally to this work.
†Corresponding authors.

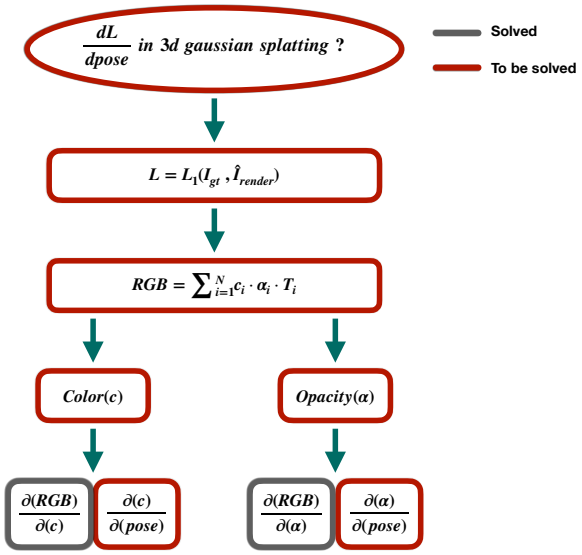

Fig. A: Derivation Framework for Computing the Derivative of L w.r.t pose. Note that the **gray boxes** denote easy-to-derive parts, considered as **"solved"** for readers, while the **red boxes** denote complicated and tedious parts, considered as **"To be solved"**, with their final results being provided directly for readability and reproducibility.

where c is the color of a Gaussian primitive G and α is the image-plane opacity. We show a preliminary mathematical framework in Fig. A. Subsequently, we will further elaborate on the above two branches.

Branch 1: the gradient of c w.r.t pose. The view-dependent color of a Gaussian primitive is obtained from spherical harmonics coefficients, which is further related to its normalized orientation $dir = (x_{norm}, y_{norm}, z_{norm})$ in the world coordinate, as in Eq. (3) and Eq. (4).

$$c = \sum_{i=1}^{M_1} s_i \cdot SH_i \cdot f_i(dir) \ , \tag{3}$$

$$f_i(dir) = f_i(\theta, \varphi) \ , \tag{4}$$

where M_1 is the degree of SH coefficients, s_i is the optimizable parameter and $f_i(dir)$ denotes a known function, the transformation from $dir = (x_{norm}, y_{norm}, z_{norm})$ to angles θ and φ is illustrated in Fig. B. Hence, in this branch, we only need to calculate the gradient of this orientation w.r.t pose, that is $\frac{d(dir)}{d(pose)}$.

$$dir = \frac{dir_{origin}}{|dir_{origin}|} = \frac{(G_x - C_x, G_y - C_y, G_z - C_z)}{|(G_x - C_x, G_y - C_y, G_z - C_z)|} \ , \tag{5}$$

$$\frac{d(dir)}{d(pose)} = \frac{d(x_{norm}, y_{norm}, z_{norm})}{d(C_x, C_y, C_z)} \ , \tag{6}$$

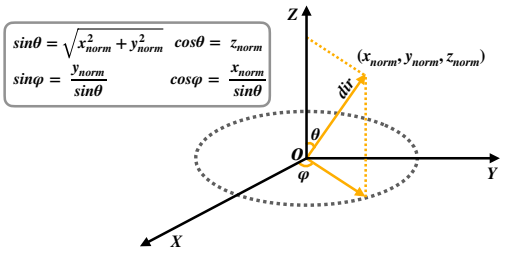

Fig. B: The conversion from dir to the angles (θ, φ), where θ and φ respectively denotes the zenith angle and the azimuth angle.

where $G_{x,y,z}$ and $C_{x,y,z}$ (Eq. (7)) represent the position of the Gaussian primitive and camera in the world coordinate. This branch is clearly visualized in Fig. C. Finally, we can easily derive corresponding results as shown in Fig. D.

$$[C_x, C_y, C_z]^T = -W^{-1}[v_{12}, v_{13}, v_{14}]^T \tag{7}$$

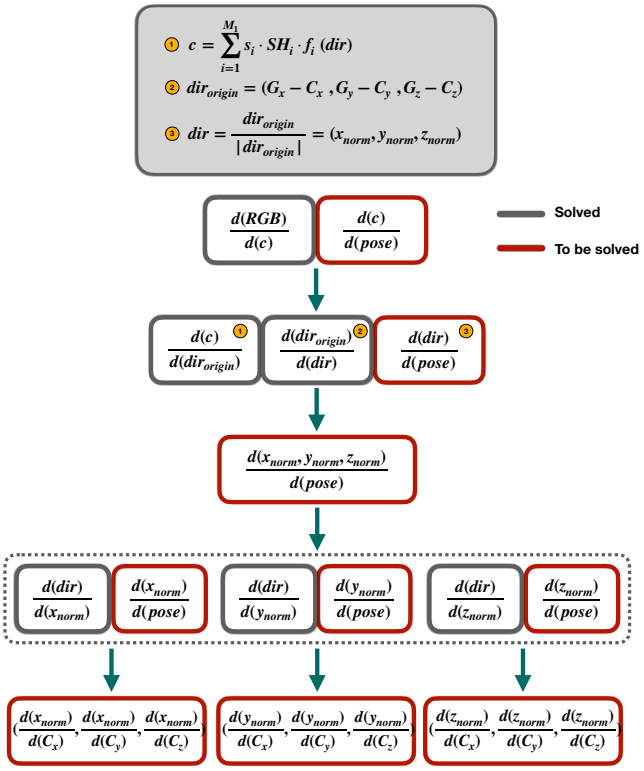

Fig. C: The framework for the derivation of the partial derivative of **Color**.

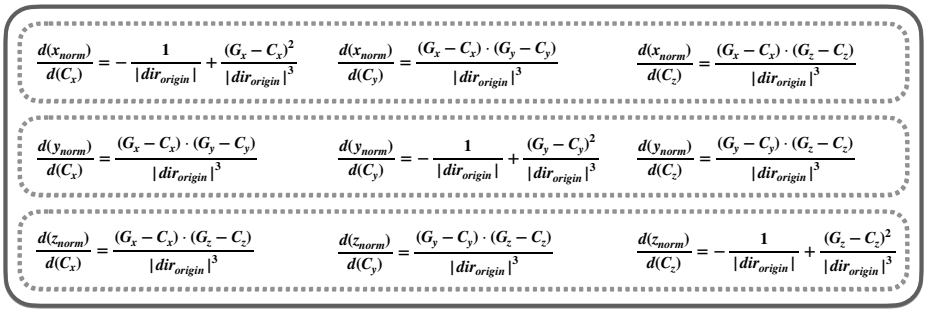

Fig. D: The derivative results of the **Color** branch.

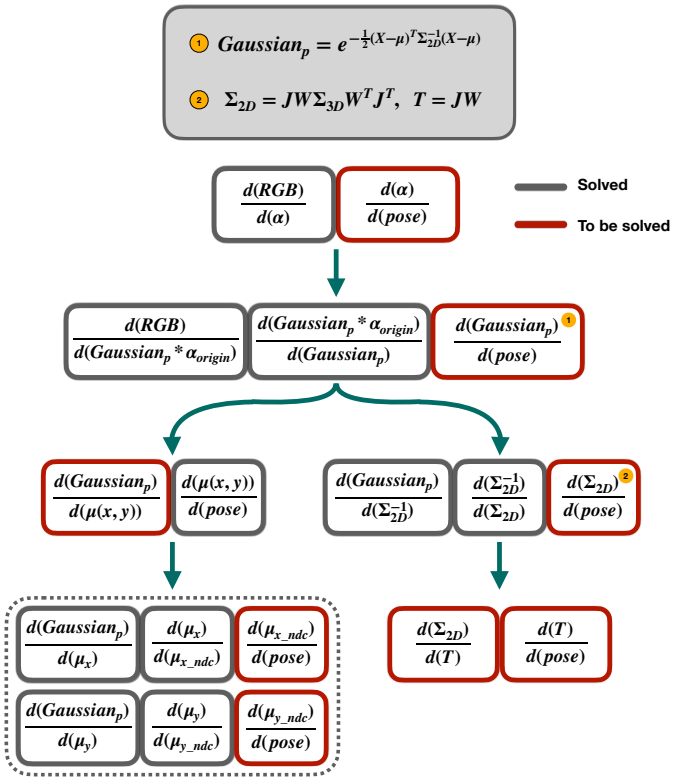

Fig. E: The framework for the derivation of the partial derivative of **Opacity**.

Branch 2: the gradient of α w.r.t pose. Camera pose plays an important role in the EWA algorithm [11]. Specifically, it determines the shape of the projected 2D Gaussian $Gaussian_p$=$\mathcal{N}(\mu, \mathbf{\Sigma_{2D}})$ and further affects the image-plane opacity α according to Eq. (5) in the main paper. We also clearly visualize this branch in Fig. E.

First, the expectation μ=(μ_x, μ_y) of a 2D Gaussian distribution is acquired by projecting the central point of its corresponding 3D Gaussian ellipsoid as in Eq. (8).

$$[g_x, g_y, g_z, g_w]^T = M \cdot \begin{bmatrix} W & t \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \cdot [G_x, G_y, G_z, 1]^T \ , \tag{8}$$

$$M = \begin{bmatrix} \frac{2n}{r-l} & 0 & -\frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & -\frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{n+f}{n-f} & -\frac{2nf}{n-f} \\ 0 & 0 & 1 & 0 \end{bmatrix} \ , \tag{9}$$

$$\mu_{ndc} = \left[\mu_{x\_ndc}, \mu_{y\_ndc}\right] = \left[\frac{g_x}{g_w}, \frac{g_y}{g_w}\right] \in [-1, 1] \ , \tag{10}$$

$$\frac{d(\mu_x)}{d(\mu_{x\_ndc})} = \frac{w}{2}, \quad \frac{d(\mu_y)}{d(\mu_{y\_ndc})} = \frac{h}{2} \ , \tag{11}$$

where M denotes the perspective matrix, set $l = -r$, $t = -b$, μ_{ndc} refers to the central point of 2D Gaussian distribution in the NDC coordinate, w and h are the width and height of an image. Then we can derive the gradient of μ w.r.t camera pose $\frac{d(\mu)}{d(pose)}$ from the forward projection process as follows:

$$\frac{d(\mu)}{d(pose)} = \frac{d(\mu_x)}{d(\mu_{x\_ndc})} \cdot \frac{d(\mu_{x\_ndc})}{d(pose)} + \frac{d(\mu_y)}{d(\mu_{y\_ndc})} \cdot \frac{d(\mu_{y\_ndc})}{d(pose)}. \tag{12}$$

We have omitted some simple intermediate steps and concluded the resulting formulas in Fig. F.

$$\frac{d\mu_{x\_ndc}}{dv_0} = \frac{1}{g_w} \cdot \left(\frac{2n}{r-l}\right) \cdot G_x \qquad \frac{d\mu_{x\_ndc}}{dv_4} = \frac{1}{g_w} \cdot \left(\frac{2n}{r-l}\right) \cdot G_y \qquad \frac{d\mu_{x\_ndc}}{dv_8} = \frac{1}{g_w} \cdot \left(\frac{2n}{r-l}\right) \cdot G_z \qquad \frac{d\mu_{x\_ndc}}{dv_{12}} = \frac{1}{g_w} \cdot \left(\frac{2n}{r-l}\right) \cdot 1$$

$$\frac{d\mu_{x\_ndc}}{dv_1} = 0 \qquad \frac{d\mu_{x\_ndc}}{dv_5} = 0 \qquad \frac{d\mu_{x\_ndc}}{dv_9} = 0 \qquad \frac{d\mu_{x\_ndc}}{dv_{13}} = 0$$

$$\frac{d\mu_{x\_ndc}}{dv_2} = g_x \cdot \left(-\frac{1}{g_w^2}\right) \cdot G_x \qquad \frac{d\mu_{x\_ndc}}{dv_6} = g_x \cdot \left(-\frac{1}{g_w^2}\right) \cdot G_y \qquad \frac{d\mu_{x\_ndc}}{dv_{10}} = g_x \cdot \left(-\frac{1}{g_w^2}\right) \cdot G_z \qquad \frac{d\mu_{x\_ndc}}{dv_{14}} = g_x \cdot \left(-\frac{1}{g_w^2}\right) \cdot 1$$

$$\frac{d\mu_{y\_ndc}}{dv_0} = 0 \qquad \frac{d\mu_{y\_ndc}}{dv_4} = 0 \qquad \frac{d\mu_{y\_ndc}}{dv_8} = 0 \qquad \frac{d\mu_{y\_ndc}}{dv_{12}} = 0$$

$$\frac{d\mu_{y\_ndc}}{dv_1} = \frac{1}{g_w} \cdot \frac{2n}{t-b} \cdot G_x \qquad \frac{d\mu_{y\_ndc}}{dv_5} = \frac{1}{g_w} \cdot \frac{2n}{t-b} \cdot G_y \qquad \frac{d\mu_{y\_ndc}}{dv_9} = \frac{1}{g_w} \cdot \frac{2n}{t-b} \cdot G_z \qquad \frac{d\mu_{y\_ndc}}{dv_{13}} = \frac{1}{g_w} \cdot \frac{2n}{t-b} \cdot 1$$

$$\frac{d\mu_{y\_ndc}}{dv_2} = g_y \cdot \left(-\frac{1}{g_w^2}\right) \cdot G_x \qquad \frac{d\mu_{y\_ndc}}{dv_6} = g_y \cdot \left(-\frac{1}{g_w^2}\right) \cdot G_y \qquad \frac{d\mu_{y\_ndc}}{dv_{10}} = g_y \cdot \left(-\frac{1}{g_w^2}\right) \cdot G_z \qquad \frac{d\mu_{y\_ndc}}{dv_{14}} = g_y \cdot \left(-\frac{1}{g_w^2}\right) \cdot 1$$

Fig. F: The derivative results of the **opacity** branch: $\frac{d(\mu_{ndc})}{d(pose)}$

Second, Eq. (13) describes in detail how the EWA splatting algorithm [11] converts a 3D symmetric covariance Σ_{3D} to a 2D symmetric covariance Σ_{2D}.

$$\Sigma_{2D} = JW\Sigma_{3D}W^T J^T \ . \tag{13}$$

This EWA process involves world-to-camera rotation $\mathbf{W}$ and an affine approximation $\mathbf{J}$ of the projective transformation. To make subsequent derivation concise and readable, we temporarily define $\mathbf{T} = \mathbf{JW}$. So far, the gradient of 2D covariance w.r.t camera pose $\frac{d(\Sigma_{2D})}{d(pose)}$ can be reformulated as the following equations. Here, Eq. (15) provides a element-wise expansion of Eq. (13).

$$\frac{d(\Sigma_{2D})}{d(pose)} = \frac{d(\Sigma_{2D})}{d(T)} \cdot \frac{d(T)}{d(pose)} \ , \tag{14}$$

$$\Sigma_{2D} = \begin{bmatrix} a & b \\ b & c \end{bmatrix} = \left[\begin{array}{ccc} T_{00} & T_{01} & T_{02} \\ T_{10} & T_{11} & T_{12} \\ \hline T_{20} & T_{21} & T_{22} \end{array}\right] \begin{bmatrix} \epsilon_0 & \epsilon_1 & \epsilon_2 \\ \epsilon_1 & \epsilon_3 & \epsilon_4 \\ \epsilon_2 & \epsilon_4 & \epsilon_5 \end{bmatrix} \left[\begin{array}{cc|c} T_{00} & T_{10} & T_{20} \\ T_{01} & T_{11} & T_{21} \\ T_{02} & T_{12} & T_{22} \end{array}\right] \ , \tag{15}$$

$$T = JW = \begin{bmatrix} \frac{f_x}{u_2} & 0 & -\frac{f_x u_0}{u_2^2} \\ 0 & \frac{f_y}{u_2} & -\frac{f_y u_1}{u_2^2} \end{bmatrix} \cdot \begin{bmatrix} v_0 & v_4 & v_8 \\ v_1 & v_5 & v_9 \\ v_2 & v_6 & v_{10} \end{bmatrix} \ , \tag{16}$$

$$\begin{bmatrix} u_0, u_1, u_2, 1 \end{bmatrix}^T = \begin{bmatrix} W & t \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \cdot \begin{bmatrix} G_x, G_y, G_z, 1 \end{bmatrix}^T \ . \tag{17}$$

Refering to the EWA algorithm [11], we ignore the third row of $\mathbf{T}$ matrix (gray elements). Additionally, we multiply the first and second rows of matrix $\mathbf{J}$ by camera intrinsics f_x and f_y, respectively, to transform Σ_{2D} into the pixel coordinate system. The gradient of Σ_{2D} w.r.t $\mathbf{T}$ is shown in Fig. G. Furthermore, in this branch, only $\frac{d(T)}{d(pose)}$ remains to be resolved, and we combine Eq. (15), Eq. (16) and Eq. (17) to provide the answer in the Fig. H.

$$\frac{d(a)}{d(T_{00})} = 2T_{00}\epsilon_0 + 2T_{01}\epsilon_1 + 2T_{02}\epsilon_2 \qquad \frac{d(a)}{d(T_{01})} = 2T_{00}\epsilon_1 + 2T_{01}\epsilon_3 + 2T_{02}\epsilon_4 \qquad \frac{d(a)}{d(T_{02})} = 2T_{00}\epsilon_2 + 2T_{01}\epsilon_4 + 2T_{02}\epsilon_5$$

$$\frac{d(a)}{d(T_{10})} = 0 \qquad \frac{d(a)}{d(T_{11})} = 0 \qquad \frac{d(a)}{d(T_{12})} = 0$$

$$\frac{d(b)}{d(T_{00})} = T_{10}\epsilon_0 + T_{11}\epsilon_1 + T_{12}\epsilon_2 \qquad \frac{d(b)}{d(T_{01})} = T_{10}\epsilon_1 + T_{11}\epsilon_3 + T_{12}\epsilon_4 \qquad \frac{d(b)}{d(T_{02})} = T_{10}\epsilon_2 + T_{11}\epsilon_4 + T_{12}\epsilon_5$$

$$\frac{d(b)}{d(T_{10})} = T_{00}\epsilon_0 + T_{01}\epsilon_1 + T_{02}\epsilon_2 \qquad \frac{d(b)}{d(T_{11})} = T_{00}\epsilon_1 + T_{01}\epsilon_3 + T_{02}\epsilon_4 \qquad \frac{d(b)}{d(T_{12})} = T_{00}\epsilon_2 + T_{01}\epsilon_4 + T_{02}\epsilon_5$$

$$\frac{d(c)}{d(T_{00})} = 0 \qquad \frac{d(c)}{d(T_{01})} = 0 \qquad \frac{d(c)}{d(T_{02})} = 0$$

$$\frac{d(c)}{d(T_{10})} = 2T_{10}\epsilon_0 + 2T_{11}\epsilon_1 + 2T_{12}\epsilon_2 \qquad \frac{d(c)}{d(T_{11})} = 2T_{10}\epsilon_1 + 2T_{11}\epsilon_3 + 2T_{12}\epsilon_4 \qquad \frac{d(c)}{d(T_{12})} = 2T_{10}\epsilon_2 + 2T_{11}\epsilon_4 + 2T_{12}\epsilon_5$$

Fig. G: The derivative results of the **opacity** branch: $\frac{d(\Sigma_{2D})}{d(T)}$.

$$\frac{d(T_{00})}{d(v_0)} = \frac{f_x}{u_2}(1 - \frac{v_2 G_x}{u_2}) \quad \frac{d(T_{00})}{d(v_4)} = \frac{f_x}{u_2}(-\frac{v_2 G_y}{u_2}) \quad \frac{d(T_{00})}{d(v_8)} = \frac{f_x}{u_2}(-\frac{v_2 G_z}{u_2}) \quad \frac{d(T_{00})}{d(v_{12})} = \frac{f_x}{u_2}(-\frac{v_2}{u_2})$$

$$\frac{d(T_{00})}{d(v_1)} = 0 \quad \frac{d(T_{00})}{d(v_5)} = 0 \quad \frac{d(T_{00})}{d(v_9)} = 0 \quad \frac{d(T_{00})}{d(v_{13})} = 0$$

$$\frac{d(T_{00})}{d(v_2)} = \frac{f_x}{u_2}(\frac{2v_2u_0G_x}{u_2^2} - \frac{u_0}{u_2} - \frac{v_0G_x}{u_2}) \quad \frac{d(T_{00})}{d(v_6)} = \frac{f_x}{u_2}(\frac{2v_2u_0G_x}{u_2^2} - \frac{v_0G_y}{u_2}) \quad \frac{d(T_{00})}{d(v_{10})} = \frac{f_x}{u_2}(\frac{2v_2u_0G_z}{u_2^2} - \frac{v_0G_z}{u_2}) \quad \frac{d(T_{00})}{d(v_{14})} = \frac{f_x}{u_2}(\frac{2v_2u_0}{u_2^2} - \frac{v_0}{u_2})$$

$$\frac{d(T_{01})}{d(v_0)} = \frac{f_x}{u_2}(-\frac{v_6 G_x}{u_2}) \quad \frac{d(T_{01})}{d(v_4)} = \frac{f_x}{u_2}(1 - \frac{v_6 G_y}{u_2}) \quad \frac{d(T_{01})}{d(v_8)} = \frac{f_x}{u_2}(-\frac{v_6 G_z}{u_2}) \quad \frac{d(T_{01})}{d(v_{12})} = \frac{f_x}{u_2}(-\frac{v_6}{u_2})$$

$$\frac{d(T_{01})}{d(v_1)} = 0 \quad \frac{d(T_{01})}{d(v_5)} = 0 \quad \frac{d(T_{01})}{d(v_9)} = 0 \quad \frac{d(T_{01})}{d(v_{13})} = 0$$

$$\frac{d(T_{01})}{d(v_2)} = \frac{f_x}{u_2}(\frac{2v_6u_0G_x}{u_2^2} - \frac{v_4G_x}{u_2}) \quad \frac{d(T_{01})}{d(v_6)} = \frac{f_x}{u_2}(\frac{2v_6u_0G_y}{u_2^2} - \frac{u_0}{u_2} - \frac{v_4G_y}{u_2}) \quad \frac{d(T_{01})}{d(v_{10})} = \frac{f_x}{u_2}(\frac{2v_6u_0G_z}{u_2^2} - \frac{v_4G_z}{u_2}) \quad \frac{d(T_{01})}{d(v_{14})} = \frac{f_x}{u_2}(\frac{2v_6u_0}{u_2^2} - \frac{v_4}{u_2})$$

$$\frac{d(T_{01})}{d(v_0)} = \frac{f_x}{u_2}(-\frac{v_6 G_x}{u_2}) \quad \frac{d(T_{01})}{d(v_4)} = \frac{f_x}{u_2}(1 - \frac{v_6 G_y}{u_2}) \quad \frac{d(T_{01})}{d(v_8)} = \frac{f_x}{u_2}(-\frac{v_6 G_z}{u_2}) \quad \frac{d(T_{01})}{d(v_{12})} = \frac{f_x}{u_2}(-\frac{v_6}{u_2})$$

$$\frac{d(T_{01})}{d(v_1)} = 0 \quad \frac{d(T_{01})}{d(v_5)} = 0 \quad \frac{d(T_{01})}{d(v_9)} = 0 \quad \frac{d(T_{01})}{d(v_{13})} = 0$$

$$\frac{d(T_{01})}{d(v_2)} = \frac{f_x}{u_2}(\frac{2v_6u_0G_x}{u_2^2} - \frac{v_4G_x}{u_2}) \quad \frac{d(T_{01})}{d(v_6)} = \frac{f_x}{u_2}(\frac{2v_6u_0G_y}{u_2^2} - \frac{u_0}{u_2} - \frac{v_4G_y}{u_2}) \quad \frac{d(T_{01})}{d(v_{10})} = \frac{f_x}{u_2}(\frac{2v_6u_0G_z}{u_2^2} - \frac{v_4G_z}{u_2}) \quad \frac{d(T_{01})}{d(v_{14})} = \frac{f_x}{u_2}(\frac{2v_6u_0}{u_2^2} - \frac{v_4}{u_2})$$

$$\frac{d(T_{10})}{d(v_0)} = 0 \quad \frac{d(T_{10})}{d(v_4)} = 0 \quad \frac{d(T_{10})}{d(v_8)} = 0 \quad \frac{d(T_{10})}{d(v_{12})} = 0$$

$$\frac{d(T_{10})}{d(v_1)} = \frac{f_y}{u_2}(1 - \frac{v_2 G_x}{u_2}) \quad \frac{d(T_{10})}{d(v_5)} = \frac{f_y}{u_2}(-\frac{v_2 G_y}{u_2}) \quad \frac{d(T_{10})}{d(v_9)} = \frac{f_y}{u_2}(-\frac{v_2 G_z}{u_2}) \quad \frac{d(T_{10})}{d(v_{13})} = \frac{f_y}{u_2}(-\frac{v_2}{u_2})$$

$$\frac{d(T_{10})}{d(v_2)} = \frac{f_y}{u_2}(\frac{2v_2u_0G_x}{u_2^2} - \frac{u_1}{u_2} - \frac{v_1G_x}{u_2}) \quad \frac{d(T_{10})}{d(v_6)} = \frac{f_y}{u_2}(\frac{2v_2u_0G_y}{u_2^2} - \frac{v_1G_y}{u_2}) \quad \frac{d(T_{10})}{d(v_{10})} = \frac{f_y}{u_2}(\frac{2v_2u_0G_z}{u_2^2} - \frac{v_1G_z}{u_2}) \quad \frac{d(T_{10})}{d(v_{14})} = \frac{f_y}{u_2}(\frac{2v_2u_0}{u_2^2} - \frac{v_1}{u_2})$$

$$\frac{d(T_{10})}{d(v_0)} = 0 \quad \frac{d(T_{10})}{d(v_4)} = 0 \quad \frac{d(T_{10})}{d(v_8)} = 0 \quad \frac{d(T_{10})}{d(v_{12})} = 0$$

$$\frac{d(T_{10})}{d(v_1)} = \frac{f_y}{u_2}(1 - \frac{v_2 G_x}{u_2}) \quad \frac{d(T_{10})}{d(v_5)} = \frac{f_y}{u_2}(-\frac{v_2 G_y}{u_2}) \quad \frac{d(T_{10})}{d(v_9)} = \frac{f_y}{u_2}(-\frac{v_2 G_z}{u_2}) \quad \frac{d(T_{10})}{d(v_{13})} = \frac{f_y}{u_2}(-\frac{v_2}{u_2})$$

$$\frac{d(T_{10})}{d(v_2)} = \frac{f_y}{u_2}(\frac{2v_2u_0G_x}{u_2^2} - \frac{u_1}{u_2} - \frac{v_1G_x}{u_2}) \quad \frac{d(T_{10})}{d(v_6)} = \frac{f_y}{u_2}(\frac{2v_2u_0G_y}{u_2^2} - \frac{v_1G_y}{u_2}) \quad \frac{d(T_{10})}{d(v_{10})} = \frac{f_y}{u_2}(\frac{2v_2u_0G_z}{u_2^2} - \frac{v_1G_z}{u_2}) \quad \frac{d(T_{10})}{d(v_{14})} = \frac{f_y}{u_2}(\frac{2v_2u_0}{u_2^2} - \frac{v_1}{u_2})$$

$$\frac{d(T_{12})}{d(v_0)} = 0 \quad \frac{d(T_{12})}{d(v_4)} = 0 \quad \frac{d(T_{12})}{d(v_8)} = 0 \quad \frac{d(T_{12})}{d(v_{12})} = 0$$

$$\frac{d(T_{12})}{d(v_1)} = \frac{f_y}{u_2}(-\frac{v_{10}G_x}{u_2}) \quad \frac{d(T_{12})}{d(v_5)} = \frac{f_y}{u_2}(-\frac{v_{10}G_y}{u_2}) \quad \frac{d(T_{12})}{d(v_9)} = \frac{f_y}{u_2} \cdot (1 - \frac{v_{10} \cdot G_z}{u_2}) \quad \frac{d(T_{12})}{d(v_{13})} = \frac{f_y}{u_2} \cdot (-\frac{v_{10} \cdot 1}{u_2})$$

$$\frac{d(T_{12})}{d(v_2)} = \frac{f_y}{u_2}(\frac{2v_{10}u_0G_x}{u_2^2} - \frac{v_9G_x}{u_2}) \quad \frac{d(T_{12})}{d(v_6)} = \frac{f_y}{u_2}(\frac{2v_{10}u_0G_y}{u_2^2} - \frac{v_9G_y}{u_2}) \quad \frac{d(T_{12})}{d(v_{10})} = \frac{f_y}{u_2}(\frac{2v_{10}u_0G_z}{u_2^2} - \frac{u_1}{u_2} - \frac{v_9G_z}{u_2}) \quad \frac{d(T_{12})}{d(v_{14})} = \frac{f_y}{u_2}(\frac{2v_{10}u_0}{u_2^2} - \frac{v_9}{u_2})$$

Fig. H: The derivative results of the **opacity** branch: $\frac{d(T)}{d(pose)}$.

B Implementation Details

More Hyperparameters. We implemented the pipeline using PyTorch 1.11.0 and Python 3.8.18. We employed the Adam optimizer with the hyperparameters $beta = (0.9, 0.999), weight\_decay = 0$, and epi =(1e-08, 1e-15) for tracking and mapping. We set the learning rate of $\{\mathfrak{so}(3)|T\}$ to $\{0.001, 0.001\}$ in sliding bundle adjustment. The Gaussian optimizer focuses on optimizing Gaussian properties including XYZ coordinates (xyz), SH coefficients, opacity, scaling, and rotation. In all our experiments, we employed the following learning rates for Gaussian optimization: $\{xyz : 0.00025;\ SH\ coefficients : 0.0025;\ opacity : 0.05;\ scaling : 0.0001;\ rotation : 0.001\}$.

Gaussian Properties Initialization. For newly added Gaussian primitives, their 3D positions and SH coefficients are calculated from depth and color observations, and other properties are initialized similarly with the original 3D Gaussian [4]. In addition, we will clone and split Gaussian primitives in the first half of mapping optimization.

Table A: Rendering Performance on the Replica Dataset [7]. Our system achieves the best performance surpassing existing methods on all three metrics: PSNR, SSIM, and LPIPS. "*****" indicates that, in the half-resolution setting, structural metrics such as SSIM and LPIPS are not comparable to those of full-resolution images.

Method	Metric	rm-0	rm-1	rm-2	off-0	off-1	off-2	off-3	off-4	Avg.
Vox-Fusion	PSNR [dB]↑	22.39	22.36	23.92	27.79	29.83	20.33	23.47	25.21	24.41
	SSIM↑	0.683	0.751	0.798	0.857	0.876	0.794	0.803	0.847	0.801
	LPIPS↓	0.303	0.269	0.234	0.241	0.184	0.243	0.213	0.199	0.236
NICE-SLAM	PSNR [dB]↑	22.12	22.47	24.52	29.07	30.34	19.66	22.23	24.94	24.42
	SSIM↑	0.689	0.757	0.814	0.874	0.886	0.797	0.801	0.856	0.809
	LPIPS↓	0.330	0.271	0.208	0.229	0.181	0.235	0.209	0.198	0.233
Co-SLAM	PSNR [dB]↑	27.27	28.45	29.06	34.14	34.87	28.43	28.76	30.91	30.24
	SSIM↑	0.910	0.909	0.932	0.961	0.969	0.938	0.941	0.955	0.939
	LPIPS↓	0.324	0.294	0.266	0.209	0.196	0.258	0.229	0.236	0.252
Point-SLAM	PSNR [dB]↑	32.40	34.08	35.50	38.26	39.16	**33.99**	33.48	33.49	35.17
	SSIM↑	0.974	0.977	0.982	0.983	0.986	0.960	0.960	0.979	0.975
	LPIPS↓	0.113	0.116	0.111	0.100	0.118	0.156	0.132	0.142	0.124
GS-SLAM	PSNR [dB]↑	31.56	32.86	32.59	38.70	41.17	32.36	32.03	32.92	34.27
	SSIM↑	0.968	0.973	0.971	0.986	**0.993**	0.978	0.970	0.968	0.975
	LPIPS↓	0.094	0.075	0.093	0.050	**0.033**	0.094	0.110	0.112	0.082
SplaTAM	PSNR [dB]↑	32.86	33.89	35.25	38.26	39.17	31.97	29.70	31.81	34.11
	SSIM↑	**0.980**	0.970	0.980	0.980	0.980	0.970	0.950	0.950	0.970
	LPIPS↓	**0.070**	0.100	0.080	0.090	0.090	0.100	0.120	0.150	0.100
Ours	PSNR [dB]↑	**33.27**	**37.78**	**38.04**	**41.03**	**41.38**	33.84	**34.60**	**37.44**	**37.17**
	SSIM↑	0.977	**0.989**	**0.990**	**0.992**	**0.993**	**0.983**	**0.987**	**0.988**	**0.987**
	LPIPS↓	0.077	**0.043**	**0.055**	**0.036**	0.038	**0.086**	**0.074**	**0.073**	**0.060**
Ours-light*	PSNR [dB]↑	32.43	35.45	36.10	39.53	40.61	33.77	34.03	37.12	36.09
	SSIM↑	0.978	0.987	0.990	0.993	0.994	0.989	0.991	0.991	0.989
	LPIPS↓	0.074	0.047	0.039	0.026	0.027	0.054	0.039	0.050	0.045

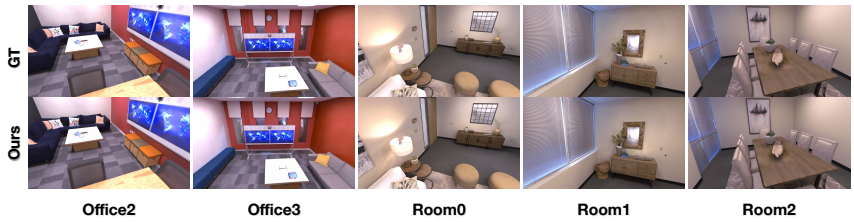

Fig. I: Rendering Performance on Replica [7] Dataset. Our system can achieve photo-realistic rendering performance.

C Rendering Results

We evaluate the average peak signal-to-noise ratio (PSNR), structural similarity (SSIM), and learned perceptual image patch similarity (LPIPS) as the rendering metrics. We compare the rendering quality of our method with the existing NeRF-SLAM and concurrent Gaussian-based works in Tab. A, where results show that our method can yield more photorealistic rendering images. We require fewer optimization iterations, than the most photorealistic NeRF-based Point-SLAM [6] and concurrent methods [3, 10], to achieve better rendering performance. Additionally, relying on the 3D Gaussian rasterization, our method can render at an extremely high speed of 770 FPS in the test.

Our proposed CG-SLAM is able to perform efficient and realistic rendering, and in this section, we additionally show more qualitative rendering images. At the same time, as a novel application, CG-SLAM allows setting a third-person view (Novel View) to observe camera motions in real time.

Additional Qualitative Renderings Results. In Fig. I, we show our rendering results on Replica [7] Dataset. We can observe that our rendered images closely resemble the ground truth ones, demonstrating that CG-SLAM is capable of achieving extremely photorealistic rendering performance.

Online Third-person View Rendering. Due to our highly consistent 3D Gaussian field and GPU-accelerated rasterizer, our system is able to support online third-person view rendering (real-time novel-view synthesis) as shown in Fig. J. Please refer to our supplementary video for the video result.

Fig. J: Third-person View Rendering. Benefiting from our efficient CG-SLAM system, We can observe camera motions in real time from a third-person viewpoint.

Table B: Runtime Analysis. In this table, we further compare real-time performance between ours and Co-SLAM [9] in TUM-RGBD [8] and ScanNet [1]. CG-SLAM still exhibits excellent efficiency in the real world. Tracking FPS is calculated solely based on per-frame tracking time.

Dataset	Method	Tracking [ms×it]↓	Mapping [ms×it]↓	Mapping Interval	Tracking FPS↑
TUM	**Ours-light**	**2.47 × 25**	13.5 × 50	15	**16.2**
	Co-SLAM	6.50 × 10	17.6 × 20	5	15.4
ScanNet	**Ours-light**	**2.59 × 25**	12.1 × 50	15	**15.4**
	Co-SLAM	6.92 × 10	19.0 × 10	5	14.4

D Further Evaluation

In this section, we further analyze the efficiency performance of our proposed system on real-world datasets in Sec. D.1. We have also provided more ablation results (Sec. D.2), tracking variance evaluation (Sec. D.3), and qualitative reconstruction details (Sec. D.4) to illustrate the effectiveness and advancement of our designs. In addition, we have added some traditional baselines in Sec. D.5 for a more comprehensive evaluation.

D.1 More Efficiency Analysis

Noisy depth images in real-world scenes pose an efficiency challenge to Gaussian-based SLAM systems because low-quality Gaussian properties initialization requires a higher optimization cost. Therefore, compared with the most efficient NeRF-based Co-SLAM [9], we further evaluate the efficiency performance of CG-SLAM in TUM-RGBD [8] and ScanNet [1]. Results in Tab. B demonstrate that our system can still maintain better efficiency in the real world.

D.2 More Ablation Results

Uncertainty Module. We perform an ablation study on the uncertainty module as shown in Tab. C, where the results further illustrate the improvement from our uncertainty module. It is worth noting that the uncertainty is also calculated in parallel, bringing negligible additional computational cost.

We have also visualized uncertainty heat maps as shown in Fig. K. Higher uncertainty frequently appears at edge regions. This occurs as Gaussians at edge regions may contribute to multiple pixels with significant depth differences, resulting in noticeable ambiguity.

Bundle Adjustment. We perform an ablation study on the bundle adjustment to demonstrate its effectiveness, as shown in Tab. D. Results show that this strategy can utilize co-visible relationships to improve tracking accuracy

Lie Algebra. We perform an ablation study on mathematical pose formats (Lie Algebra/Quaternion) in Tab. E to illustrate our empirical finding mentioned in the main paper.

Table C: Uncertainty Model Ablation Results (ATE RMSE [cm] ↓). The uncertainty model can effectively improve tracking performance.

Setting	rm-0	rm-1	rm-2	off-0	off-1	off-2	off-3	off-4	Avg.
w/ UN	**0.29**	**0.27**	**0.25**	**0.33**	**0.14**	**0.28**	**0.31**	**0.29**	**0.27**
w/o UN	0.32	0.28	0.27	**0.33**	0.18	0.40	0.46	**0.29**	0.32

Table D: Bundle Adjustment Ablation Results (ATE RMSE [cm] ↓). It can be seen that the tracking accuracy gradually improves with increasing BA iterations.

Setting	rm-0	rm-1	rm-2	off-0	off-1	off-2	off-3	off-4	Avg.
BA-30iters	**0.29**	**0.27**	**0.25**	**0.33**	**0.14**	**0.28**	**0.31**	**0.29**	**0.27**
BA-20iters	0.38	0.32	0.28	0.34	0.16	0.36	0.38	0.30	0.32
BA-10iters	0.47	0.32	0.53	0.44	0.15	0.36	0.45	0.33	0.38

Table E: Pose Fromats Ablation Results (ATE RMSE [cm] ↓). We empirically discovered that the Lie Algebra is more advantageous for tracking in a Gaussian field. "-" indicates a failure situation.

Setting	rm-0	rm-1	rm-2	off-0	off-1	off-2	off-3	off-4	Avg.
Lie Algebra	**0.29**	**0.27**	**0.25**	0.33	**0.14**	**0.28**	**0.31**	**0.29**	**0.27**
Quad	0.52	1.43	0.31	**0.28**	0.22	0.53	1.53	-	-

Table F: Isotropy Ablation in Rendering (PSNR [dB] ↓). These results indicate that anisotropy regularization only has a slight impact on rendering quality. "-": we don't list results for Office-4 due to the failure of 'w/o $\mathcal{L}_{iso}$' setting.

Setting	rm-0	rm-1	rm-2	off-0	off-1	off-2	off-3	off-4	Avg.
w/ $\mathcal{L}_{iso}$	33.27	37.78	38.04	41.03	41.38	33.84	**34.60**	**-**	37.13
w/o $\mathcal{L}_{iso}$	**34.92**	**37.89**	**38.66**	**42.31**	**42.01**	**37.01**	30.63	**-**	**37.63**

Isotropy in Rendering. In Tab. F, quantitative results reveal the impact of the anisotropy regularization term on rendering. It is evident that anisotropy regularization is a necessary design, considering the slight decrease in rendering quality and significant improvement in tracking.

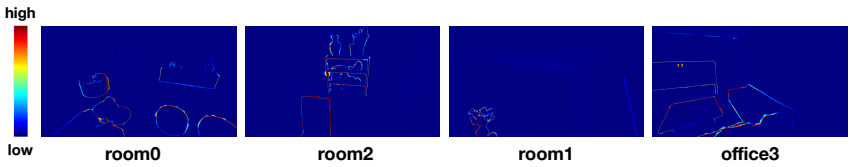

Fig. K: Uncertainty Heat Map. It is clear that high uncertainty always occurs at ambiguous edge regions.

Table G: Alignment & Variance Loss Ablation Results. We analyze the effectiveness of alignment and variance losses using tracking and reconstruction metrics. We show average results from 8 Replica [7] scenes in this table.

Metric	w/o $\mathcal{L}_{align}$	w/o $\mathcal{L}_{Var}$	w/o $\mathcal{L}_{align}$&$\mathcal{L}_{var}$	Ours-full
RMSE.[cm]↓	0.28	0.30	0.33	**0.27**
Chamfer dis.[cm]↓	4.74	4.57	4.79	**3.85**

Table H: Variance Evaluation. Experimental results show that compared to pixel-level optimization (Ray-tracing), image-level optimization (Rasterization) can further reduce tracking variance.

Method	Metric	rm-0	rm-1	rm-2	off-0	off-1	off-2	off-3	off-4	Avg.
Co-SLAM	Mean [cm]↓	0.46	0.60	0.61	0.45	0.41	1.68	0.93	0.59	0.72
	Median [cm]↓	0.42	0.52	0.57	0.39	0.38	1.56	0.86	0.50	0.65
	Std [cm]↓	0.25	0.38	0.35	0.32	0.21	0.80	0.46	0.36	0.39
Ours-light	Mean [cm]↓	**0.32**	**0.31**	**0.23**	**0.23**	**0.28**	**0.40**	**0.49**	**0.49**	**0.34**
	Median [cm]↓	**0.31**	**0.28**	**0.20**	**0.22**	**0.27**	**0.37**	**0.48**	**0.45**	**0.32**
	Std [cm]↓	**0.13**	**0.14**	**0.17**	**0.10**	**0.10**	**0.17**	**0.19**	**0.22**	**0.15**

Alignment&Variance Losses. To explain the contribution of our alignment and variance losses to a 3D Gaussian field, we add the *Chamfer Distance* as a new metric to quantify the distance between Gaussian primitives and surfaces. As shown in Tab. G, these two loss functions effectively reduce the *Chamfer Distance* and further have a positive impact on the tracking metric.

D.3 Variance Evaluation

In Tab. H, we evaluate the standard deviation, median, and mean metrics from Co-SLAM [9] and ours. It is obvious that image-level optimization can produce lower variance compared to the pixel-level one.

D.4 Reconstruction Details.

As shown in Fig. L, compared to the most efficient NeRF-based Co-SLAM [9], we zoomed in on qualitative reconstruction details, demonstrating that our method can maintain fine-grained mapping performance while achieving higher efficiency.

D.5 Traditional Baselines

Experiments. We have taken the currently popular ORB-SLAM2 [5] and BundleFusion [2] as our traditional baselines, and similarly evaluated tracking accuracy across various datasets. Tab. I shows results in Replica [7], which demonstrate that CG-SLAM can achieve better performance than traditional baselines with decent depth maps. Traditional feature-based pipelines are sensitive to textureless scenes. For example, ORB-SLAM2 failed when facing a textureless wall in

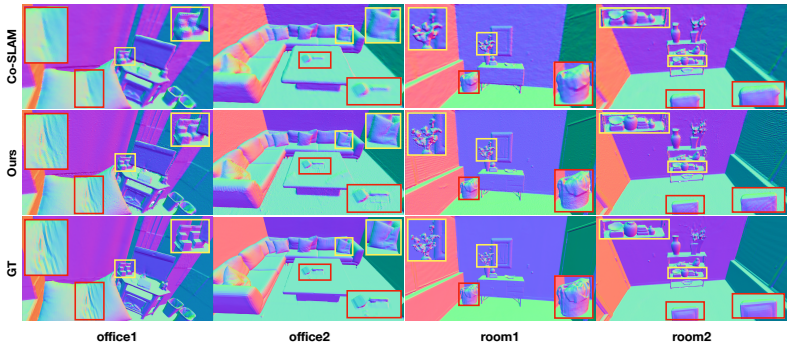

Fig. L: Reconstruction Details. Some reconstruction details are emphasized in this figure, such as folds on cushions and corners of chairs, to illustrate that our method can produce more precise and sharp results.

Room 2. However, relying on image-level backpropagation, our system can still solve accurate poses in the same scene. The Gaussian-based system exhibits robustness to extreme scenarios. In challenging real-world TUM-RGBD [8] and ScanNet [1] datasets, despite being interfered by noisy depth, CG-SLAM still produces state-of-the-art tracking results in Tabs. J and K across various scenes, except for being slightly worse than ORB-SLAM2 [5] in the TUM-RGBD [8]. We reproduce all results from traditional methods in RGB-D mode.

Discussion. Both NeRF-based and Gaussian-based SLAM works utilize the differentiable rendering pipeline for tracking and mapping, which is essentially different from traditional methods. Such rendering-based works enable photorealistic rendering and full-dense reconstruction, which show promising potential to solve some longstanding challenges. However, there is still a gap between these new-developed methods and traditional ones, especially in efficiency and memory consumption. This is exactly why traditional methods can gain popularity in real-world applications. To this end, we expect that our work can effectively bridge this gap and make meaningful contributions to this emerging rendering-based SLAM community.

Table I: Tracking Results on the Replica Dataset [7] (ATE RMSE [cm] ↓). Both our versions achieve state-of-the-art performance in this dataset. "*" indicates that ORB-SLAM2 [5] fails at the end of the Room2 sequence and we exclude those failure frames in the evaluation.

Method	rm-0	rm-1	rm-2	off-0	off-1	off-2	off-3	off-4	Avg.
ORB-SLAM2	0.55	0.33	0.47*	0.49	0.58	1.09	0.91	1.29	0.71
BundleFusion	0.64	0.59	1.34	0.60	0.97	1.03	0.66	0.90	0.84
Co-SLAM	0.77	1.04	1.09	0.58	0.53	2.05	1.49	0.84	0.99
Ours	**0.29**	**0.27**	**0.25**	0.33	**0.14**	**0.28**	**0.31**	**0.29**	**0.27**
Ours-light	0.44	0.36	0.33	**0.29**	0.27	0.43	0.52	0.58	0.40

Table J: Tracking Results on the TUM-RGBD Dataset [8] (ATE RMSE [cm] ↓). Our method demonstrates competitive performance on this dataset, compared to traditional baselines, although slightly inferior to ORB-SLAM2 [5].

Method	fr1/desk	fr1/desk2	fr1/room	fr2/xyz	fr3/office	Avg.
ORB-SLAM2	**1.60**	**2.20**	**4.70**	**0.40**	**1.00**	**1.98**
BundleFusion	3.45	3.47	14.39	1.66	3.43	5.28
Co-SLAM	2.70	4.57	30.16	1.90	2.60	8.38
Ours	2.43	4.54	9.39	1.20	2.45	4.00
Ours-light	3.14	4.73	10.67	1.28	2.60	4.48

Table K: Tracking Results on the ScanNet Dataset [1] (ATE RMSE [cm] ↓). CG-SLAM is able to perform more accurate and robust camera tracking in ScanNet [1] dataset. "-" indicates failure cases of BundleFusion [2].

Method	Sc.0000	Sc.0059	Sc.0106	Sc.0169	Sc.0181	Sc.0207	Avg.
ORB-SLAM2	8.43	**7.46**	8.94	8.74	23.97	7.42	10.83
BundleFusion	9.09	18.64	-	8.06	**11.33**	8.47	-
Co-SLAM	7.18	12.29	10.9	**6.62**	13.43	7.13	9.37
Ours	7.09	**7.46**	**8.88**	8.16	11.60	**5.34**	**8.08**
Ours-light	**5.62**	8.73	9.78	7.93	12.02	5.45	8.17

References

1. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5828–5839 (2017)
2. Dai, A., Nießner, M., Zollhöfer, M., Izadi, S., Theobalt, C.: Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. ACM Transactions on Graphics (ToG) **36**(4), 1 (2017)
3. Keetha, N., Karhade, J., Jatavallabhula, K.M., Yang, G., Scherer, S., Ramanan, D., Luiten, J.: Splatam: Splat, track & map 3d gaussians for dense rgb-d slam. arXiv preprint arXiv:2312.02126 (2023)
4. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (2023)
5. Mur-Artal, R., Tardós, J.D.: ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras. IEEE Transactions on Robotics **33**(5), 1255–1262 (2017). https://doi.org/10.1109/TRO.2017.2705103
6. Sandström, E., Li, Y., Van Gool, L., Oswald, M.R.: Point-slam: Dense neural point cloud-based slam. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 18433–18444 (2023)
7. Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J.J., Mur-Artal, R., Ren, C., Verma, S., Clarkson, A., Yan, M., Budge, B., Yan, Y., Pan, X., Yon, J., Zou, Y., Leon, K., Carter, N., Briales, J., Gillingham, T., Mueggler, E., Pesqueira, L., Savva, M., Batra, D., Strasdat, H.M., Nardi, R.D., Goesele, M., Lovegrove, S., Newcombe, R.: The replica dataset: A digital replica of indoor spaces (2019)

8. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of rgb-d slam systems. In: 2012 IEEE/RSJ international conference on intelligent robots and systems. pp. 573–580. IEEE (2012)
9. Wang, H., Wang, J., Agapito, L.: Co-slam: Joint coordinate and sparse parametric encodings for neural real-time slam. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13293–13302 (2023)
10. Yan, C., Qu, D., Xu, D., Zhao, B., Wang, Z., Wang, D., Li, X.: Gs-slam: Dense visual slam with 3d gaussian splatting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 19595–19604 (2024)
11. Zwicker, M., Pfister, H., Van Baar, J., Gross, M.: Ewa volume splatting. In: Proceedings Visualization, 2001. VIS'01. pp. 29–538. IEEE (2001)