ingegneria sismica
International Journal of Earthquake Engineering
Anno XLIII - Num. 2 - 2026

Gaussian Splatting-based Dense SLAM: A Unified Framework for Photo-realistic Reconstruction and Accurate Tracking

Weizhao Tian[1], *Lingyan Hu*[1,*] *and Dongmei Xu*[1]

[1] Electronic and Electrical Engineering of Shanghai University of Engineering Science Shanghai 201620, Shanghai, China

SUMMARY: *Gaussian Simultaneous Localization and Mapping (SLAM) technology has problems with reconstruction accuracy and visual quality in complex scenes and high-frequency details. Therefore, in this paper, a dense visual SLAM framework based on Gaussian Splatting is proposed to represent the scene as 3D Gaussian primitives. A new adaptive sampling method can be used to select multiple representative samples from the data for reconstruction. It is an exponential decay weight method that gives higher weight to recent observations to improve reconstruction accuracy and tracking performance. Add a perception loss function to the system that emphasizes high-level semantic features in optimisation. Strengthen the reconstruction of fine scene details and improve the general visual quality this way. According to the above experiments, the proposed method can achieve good results in scene reconstruction and precise camera trajectory estimation. Based on the above comparison, PSNR is approximately 32.38% higher on average, SSIM has risen by 17.48% on average, LPIPS has increased by 47.57% on average, and Absolute Trajectory Error (ATE) (Root Mean Square Error, RMSE) has improved by about 29.20% on average.*

KEYWORDS: *SLAM; Gaussian Splatting; adaptive sampling; exponential decay weighting; perceptual loss*

1 Introduction

Over the past two decades, Visual Simultaneous Localization and Mapping (SLAM) technology has been widely applied in robotics and autonomous driving systems for real-time 3D reconstruction of the environment and camera trajectory estimation [1]. Deep learning and hardware development have enhanced the accuracy of SLAM systems to a certain extent. Sparse-to-dense feature matching networks have increased the accuracy of feature matching significantly [2], and multi-scale Transformer structures have been applied to high-quality 3D reconstruction [3]. Further improve the retention of details in the field of video processing through inter-frame feedback fusion-based networks [4]. Neural Gaze-Area Super-Resolution [5], Sparse Aggregated 3D Convolutional Structures [6], and Efficient Stereo Rendering Methods [7] have also shown good results in improving the accuracy of image reconstruction and enhancing rendering efficiency. Recently, region-sampling neural radiation field (NeRF)-SLAM frameworks based on Kolmogorov-Arnold networks have been introduced that can reduce the initialization delay significantly and improve reconstruction efficiency and accuracy [8].

*linjunjie@hust.edu.cn
https://doi.org/10.65102/is2026882
Published: 30 April, 2026

Traditional SLAM methods are based on sparse keypoints or dense depth maps, and thus have deficiencies in capturing high-frequency details and processing large-scale scenes. To solve the problem of high computation cost for capturing fine details and handling large scenes with traditional methods, new scene representations such as neural rendering and NeRF have recently been introduced into SLAM systems. Recently, 3D Gaussian Splatting-based explicit representations have become a popular subject of research in the development of real-time dense SLAM systems due to their efficient rendering performance, as shown by methods such as SplaTAM and Gaussian-SLAM. However, the existing Gaussian Splatting-based SLAM methods are still limited in their ability to capture fine structural details due to insufficient modelling and processing strategies, and thus may lose important scene information. Optimisation of Gaussian parameters is also susceptible to interference from noise and can be trapped in a local optimum, thereby reducing the accuracy of the final reconstructed geometry and appearance. The initialisation and density adjustment of Gaussian points are unable to respond to changes in scene feature density. Therefore, there are relatively few specific details in the area of interest that can be recovered; thus, reconstruction accuracy will be restricted.

In response to the above problems, a new dense visual SLAM system based on Gaussian splatting is proposed in this paper. The first is as follows:

- An adaptive sampling strategy: To improve the accuracy of reconstruction, dynamically adjust the sampling positions of an adaptive sampling strategy.
- An exponential decay weighting: To enhance the accuracy of reconstruction and tracking, give higher weight to more recent observations using an exponential decay method.
- A Perceptual Loss Function: To improve the reconstruction of fine scene details and enhance the visual quality of the results, a perceptual loss function can be used to focus on high-level semantic features during optimisation.

2 Related Work

2.1 Visual SLAM

Early visual SLAM technology uses only a few sparse feature points for pose tracking and is therefore limited in building complete maps [9, 10]. However, the main deficiency of the above methods is that they are not suitable for effectively building models of the complex environment. To address the deficiency in representing detailed scene structures, KinectFusion [11] has been proposed as a Red, Green, Blue-Depth (RGB-D) SLAM technology that uses voxel fusion for dense reconstruction and significantly improves the geometric quality of indoor scenes. However, the above methods are still prone to missing high-frequency details because they are based on depth sensors and cannot acquire fine information such as textures and edges. They are also less economically viable at the same time. Fixed-resolution voxels result in oversampling of smooth regions and thus lack fine-grained details. They are also not suitable for change in the field.

2.2 Deep Learning Reconstruction Methods

To solve the problem of detailed reconstruction and the difficulty of real-time performance, neural rendering technology has recently been developed. The above are detail-enhancement methods. NeRF [12, 13] employs implicit volume representations to generate new views at the image level and has greatly improved the quality of reconstructed high-frequency scene details. NICE-SLAM also adds NeRF to the SLAM framework and achieves implicit dense mapping. However, the above methods have a high computational cost because of the challenges in ray sampling and multilayer perceptron (MLP) queries for NeRF to meet the real-time requirements

of SLAM, and the optimisation process also has slow convergence. To address the conflict between real-time performance and high-fidelity reconstruction, Transformer with Multi-Scale Dense Network (TMSDNet), based on Transformer multiscale networks and neural super-resolution techniques, has been proposed to enhance efficiency but has not fundamentally solved this problem.

2.3 3D Gaussian Splatting SLAM

In light of the deficiencies of implicit methods in real-time applications, explicit 3D Gaussian Splatting (3DGS) has gradually been introduced into the field of research. Point-SLAM combines 3D Gaussian Splatting with SLAM to achieve high-speed differentiable rendering, for example. SplaTAM can help increase the real-time perception of the environment for people through RGB textures and 3DGS. These ways have reduced the real-time bottleneck of implicit SLAM, but at the same time, new problems have emerged in the optimisation of high-frequency details. The Gaussian parameter is very sensitive to noise, so the optimisation will be affected by this noise; as a result, the texture restoration effect cannot be as good as that achieved by NeRF. The point distribution strategy is still not ideal; it uses a fixed initialization and thus leads to under-sampling in feature-dense areas and oversampling in smooth areas. The design of the loss function is limited to photometric errors, lacks high-level semantic or geometric constraints, and thus is unable to correct trajectory drift effectively. Recently, some scholars have addressed the problems mentioned above. Mao and others proposed a hierarchical SLAM framework using 3D LiDAR sensors to enhance the accuracy and stability of reconstruction by integrating voxel mapping, LiDAR odometry, global pose graph optimisation and dense surface reconstruction [14]. Xiang et al. have proposed a loop-closure-enhanced SLAM system for high-similarity indoor scenes to improve the accuracy of initialization and inter-frame correlation through deep-learning-based detection mechanisms [15]. Zou and his group have shown that, in order to improve the robustness and accuracy of SLAM systems in dynamic environments under difficult conditions, it is necessary to combine 3D LiDAR point clouds with visual-inertial data [16].

In short, the current SLAM technology is still limited by various deficiencies and cannot meet the demand for real-time operation. The 3D Gaussian Splatting method is good at reconstructing high-quality and accurate-to-tracking 3D models of ordinary scenes, but it has been difficult to achieve fine details in complex environments. The optimisation process does not have good stability because the Gaussian parameters are prone to local optima. The current SplaTAM and Gaussian-SLAM methods have shown that they cannot effectively separate the impact of midpoint density on scene complexity, and thus are limited in reconstruction quality and tracking accuracy.

In response to the problems of the existing 3DGS SLAM method, such as a lack of detail reconstruction, this paper puts forward an adaptive Gaussian SLAM system. The three main technologies of this system address the above problems effectively: (1) an adaptive sampling strategy that changes sampling locations dynamically to improve the accuracy of reconstruction; (2) an exponential decay weighting scheme that weighs recent observations more heavily for enhanced reconstruction quality and tracking precision; (3) a perceptual loss function that focuses on high-level semantic features during optimisation to improve the reconstruction of fine scene details and boost the visual quality of the results.

The following is the organisation of the rest of the paper. The way I will do so is shown in Section III. Basic Principles of 3D Gaussian Splatting, Camera Pose Optimisation, Map Construction and Optimisation (including adaptive sampling strategy, exponential decay weighting strategy, and perceptual loss function). In Section IV, this paper presents the analysis

results of the experiments and adds tests for rendering performance, tracking accuracy, etc., along with ablation studies. Finally, the conclusion is in Section V.

3 Method

The first method introduced in this paper is based on a 3D Gaussian distribution for map construction. RGB-D neural network SLAM can also achieve better rendering effects and scalability in real-world environments, and thus is likely to be applied in practice. As shown in Figure 1, the first two categories of the optimised proposals to the benchmark method are pose tracking optimisation and map construction optimisation. First, the input keyframes are used to estimate the position of the camera, generate Gaussian point clouds, and determine whether a new sub-map needs to be created based on the threshold for translation or rotation. After observation sampling in the adaptive sampling module, the quality of the point cloud is continuously improved by parameter optimisation of the Gaussian point cloud, which integrates colour, depth, perception, regularization loss and exponential decay weighting. In addition, keyframes are rendered with differential rasterization to calculate depth residuals and densification masks, and thus drive the update of the dense point cloud. Finally, by activating keyframes in the sub-map, a closed-loop real-time high-precision mapping and position tracking have been achieved.

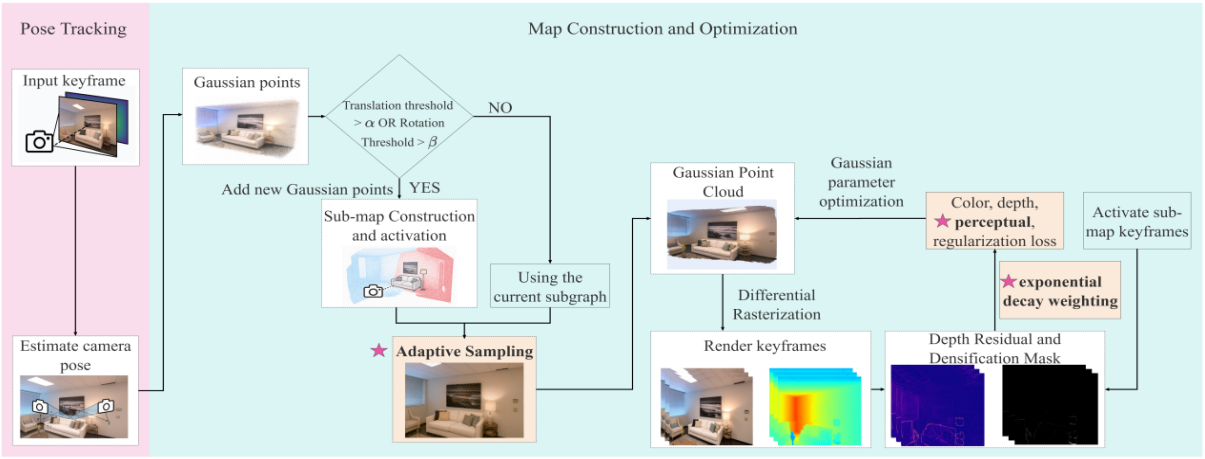


Figure 1: Structure Diagram of Gaussian Splatting-based Dense SLAM.

3.1 Basic Principles of 3D Gaussian Splatting

$\boldsymbol{u} \in \mathbb{R}^3$ The 3D Gaussian function is used to represent the basic unit of the scene in this paper, and the environment is modeled as a point cloud of many anisotropic 3D Gaussian points. A 3D Gaussian point is expressed as an ellipsoid and has a centre position vector. The position and scale of the object, an opacity value to adjust its transparency, and an RGB colour vector of the object are provided in this covariance matrix. This representation naturally includes continuous volumetric regions in space and can perform continuous image generation during rendering.

Given the camera pose and intrinsic parameters, the pinhole model can be used to project 3D Gaussian points onto the image plane and obtain the corresponding 2D Gaussian distributions $\boldsymbol{P} \in \mathbb{R}^{4\times4}$. denotes the rigid transformation from the world coordinate system to the camera coordinate system. $\boldsymbol{T}_{wc} \in \boldsymbol{SE(3)}$ is the camera projection function that maps 3D points in the camera coordinate system to the image plane in homogeneous coordinates. $\pi(\cdot)$ applies

perspective division to convert homogeneous image coordinates into normalized 2D-pixel coordinates for further processing, such as rendering and loss computation. Then, the projected position of the 3D Gaussian centre on the image plane can be written as:

$$\boldsymbol{u}_{\mathrm{I}} = \pi\left(\boldsymbol{P}\left(\boldsymbol{T}_{\mathrm{wc}} \cdot [u,1]^{\mathrm{T}}\right)\right) \tag{1}$$

$[u,1]^{\mathrm{T}}$ u $\boldsymbol{\Sigma}$ $\boldsymbol{\Sigma}_{\mathrm{I}}$ $\boldsymbol{R}_{\mathrm{wc}} \in \mathbb{R}^{3\times3}$ $\boldsymbol{T}_{\mathrm{wc}}$ $\boldsymbol{J}$ Where denotes adding a homogeneous coordinate of 1 to . A Jacobian matrix can be used on the covariance matrix of a 3D Gaussian distribution to obtain the corresponding projection covariance matrix in a 2D image space [19]. Let $\mathbf{R}_p$ and $\mathbf{p}$ be the rotational part and projection Jacobian matrix, respectively, of $\mathbf{p}$ in the camera coordinate system. Next:

$$\boldsymbol{\Sigma}_{\mathrm{I}} = \boldsymbol{J}\boldsymbol{R}_{\mathrm{wc}}\boldsymbol{\Sigma}\boldsymbol{R}_{\mathrm{wc}}^{\mathrm{T}}\boldsymbol{J}^{\mathrm{T}} \tag{2}$$

$\boldsymbol{\Sigma}_{\mathrm{I}}$ Where it shows the scale and shape of the projected Gaussian in the image. Each projected Gaussian point covers a certain range of pixels in rendering and contributes to a particular colour and opacity.

j i $\alpha_j(i)$ $T_j(i)$ $T_j(i) = \prod_{u<j}(1-\alpha_j(i))$ j The contribution of a single Gaussian point to a pixel is determined by its opacity and the accumulated transmittance, considering the cumulative transparency of all preceding Gaussians. The final reconstructed colour of a pixel is computed as a weighted sum of the contributions from all Gaussians: $\boldsymbol{I}(i)$ i

$$\boldsymbol{I}(i) = \sum_{j=1}^{m} \alpha_j(i) T_j(i) \boldsymbol{C}_j \tag{3}$$

m i $\boldsymbol{C}_j$ i $\boldsymbol{D}(i)$ Where does the number of Gaussians influencing a pixel come from, and what is the colour of the n-th Gaussian? The depth map of the rendering result can be further obtained by weighting and merging the depth information at the centre of the Gaussian.

$$\boldsymbol{D}(i) = \alpha_j(i) T_j(i) \boldsymbol{u}_{z,j} \tag{4}$$

$\boldsymbol{u}_{z,j}$ j Where represents the depth value of the centre of the -th Gaussian in the camera coordinate system. The Rendering Process mentioned above has microscale features. Together, the positions, covariances, colours and opacities of all the Gaussian points determine the colours of the displayed pixels. Therefore, changes in these features will continue to affect the image reconstruction, and a gradient of the reconstruction error can be obtained through backpropagation. Provide a theoretical basis for the following gradient-based optimisation of maps and camera poses.

3.2 Camera Pose Optimization

The entire system needs to precisely determine the position and orientation of the camera for good construction of the 3D model in the scene.

The first way to show the problem systematically is as maximum a posteriori estimation:

$$\boldsymbol{T}_t^* = \arg\max_{\boldsymbol{T}_t} p(\boldsymbol{T}_t \mid \boldsymbol{I}_t, \boldsymbol{D}_t, \mathcal{G}, \boldsymbol{T}_{t-1}) \tag{5}$$

$\boldsymbol{G} \in \mathbb{R}^{N\times d}$ $\boldsymbol{I}_t, \boldsymbol{D}_t$ $\boldsymbol{T}_{t-1} \in \boldsymbol{SE}(3)$ Where is a Gaussian scene representation? RGB and depth maps are currently in the current frame, and the last pose is unknown. To improve the convergence speed, the initialisation will be performed using a dense RGB-D odometry or a constant-velocity assumption. When the texture is rich in information, its initial position is not . However, if the texture is too light, its first appearance will be . Next, the pose can be expressed in a minimally parameterized form of the Lie group: $\boldsymbol{T}_t^{\text{init}} = \boldsymbol{T}_{t-1} \cdot \Delta \boldsymbol{T}_{\text{odom}}$ $\boldsymbol{T}_t^{\text{init}} = \boldsymbol{T}_{t-1} \cdot \Delta \boldsymbol{T}_{\text{const}}$

$$\boldsymbol{T}(\boldsymbol{\xi}) = \begin{pmatrix} \boldsymbol{R}(\boldsymbol{q}) & \boldsymbol{t} \\ \boldsymbol{0}^{\mathrm{T}} & 1 \end{pmatrix} \tag{6}$$

$\boldsymbol{\xi} = [\boldsymbol{q}, \boldsymbol{t}]^{\mathrm{T}}$ $\boldsymbol{\xi}$ $\boldsymbol{q} \in \mathbb{R}^4$ $\boldsymbol{SO}(3)$ $\boldsymbol{t} \in \mathbb{R}^3$ $\boldsymbol{R}(\cdot)$ **R(q) T(ξ) SE(3)** $\boldsymbol{\xi}$ Where . is the minimal parameterized vector of the optimised variables of the camera pose. is a quaternionic rotational component, guaranteed by the exponential mapping constraints. is the translation vector. is a mapping that maps unit quaternions to orthogonal rotation matrices in 3D space. converts quaternions to rotation matrices. is a pose matrix constructed by combining . Then, by using the Gaussian algorithm to generate a combined view and transparency information, a Gaussian scene can be micro-rendered:

$$\boldsymbol{I}_t, \boldsymbol{D}_t, \alpha_t = \mathfrak{R}(\mathcal{G}, \boldsymbol{T}(\boldsymbol{\xi}); \boldsymbol{K}) \tag{7}$$

$\boldsymbol{I}_t$ $\boldsymbol{D}_t$ α_t $\mathfrak{R}$ $\boldsymbol{K} \in \mathbb{R}^{3\times 3}$ Where is the rendered RGB image? Where is the depth map rendered? What is the rendering transparency of each pixel? Is there a differentiable rendering operator? Is there a camera's internal reference matrix? Set up the weighted robust residuals for each pixel:

$$\rho(\boldsymbol{u}) = \omega(\boldsymbol{u}) \left[\lambda \left| \boldsymbol{I}_t(\boldsymbol{u}) - \boldsymbol{I}_t(\boldsymbol{u}) \right| + (1-\lambda) \left| \boldsymbol{D}_t(\boldsymbol{u}) - \boldsymbol{D}_t(\boldsymbol{u}) \right| \right] \cdot M_{\text{depth}}(\boldsymbol{u}) \cdot M_{\text{outlier}}(\boldsymbol{u}) \tag{8}$$

$\omega_\alpha(\boldsymbol{u}) = [\alpha_t(\boldsymbol{u})]^3$ $\lambda \in [0,1]$ $M_{\text{depth}}, M_{\text{outlier}}$ Where is the transparency weight? 'is a photometric and geometric equilibrium factor. 'are depth-effective values and outlier masks. 'is the total weighted residual at the pixel level. The overall loss function is then: $\rho(\boldsymbol{u})$

$$\mathcal{L}(\boldsymbol{\xi}) = \sum_{u \in \Omega} \rho(\boldsymbol{u}) \tag{9}$$

Ω $\mathcal{L}(\boldsymbol{\xi})$ Where is the set of pixels in the image? is the total residual loss across all pixels, and is the final optimisation target.

To find the optimal pose parameters of the camera, we want to minimise the loss: $\boldsymbol{\xi}$

$$\boldsymbol{\xi}^* = \arg\min_{k \in (1,\ldots,N)} \mathcal{L}(\boldsymbol{\xi}^{(k)}) \tag{10}$$

k $\boldsymbol{\xi}^*$ Where is the number of iterations, and what is the parameter corresponding to the minimum loss in an iteration?

Adam is an adaptive optimisation method that uses first-order and second-order moment estimates of the gradient to dynamically adjust the learning rate for each parameter and thus improves the speed of convergence and numerical stability in complex, non-convex optimisation problems. To optimise the camera's pose parameters of the current frame, the Adam optimizer is employed: $\boldsymbol{\xi}$

$$\boldsymbol{\xi}^{(k+1)} = \boldsymbol{\xi}^{(k)} - \boldsymbol{\eta} \odot \frac{\boldsymbol{m}^{(k)}}{\sqrt{\boldsymbol{v}^{(k)}} + \varepsilon} \tag{11}$$

$\boldsymbol{m}^{(k)}$ $\boldsymbol{m}^{(k)} = \beta_1 \boldsymbol{m}^{(k-1)} + (1-\beta_1)\nabla_{\xi}\mathcal{L}$ $\boldsymbol{v}^{(k)}$ $\boldsymbol{v}^{(k)} = \beta_2 \boldsymbol{v}^{(k-1)} + (1-\beta_2)(\nabla_{\xi}\mathcal{L})^2$ $\boldsymbol{\eta} \in \mathbb{R}^7$ Where is a first-order gradient, is the second-order gradient, the component learning rate vector, is the momentum factor, is a numerical stabilizer, and is element-wise multiplication. β_1, β_2 ε $\odot$

$\boldsymbol{\xi}^*$ Although it is the optimised relative motion of the current frame with respect to the previous frame, it is in the camera coordinate system. To obtain the overall pose in the world coordinate system, this relative transformation is combined with the global pose of the previous frame to get: $\boldsymbol{T}_t^*$ $\boldsymbol{T}_{t-1}$

$$\boldsymbol{T}_t^* = \boldsymbol{T}_{t-1} \cdot \boldsymbol{T}(\boldsymbol{\xi}^*)^{-1} \tag{12}$$

The above process will use the global reference coordinate system as the reference. Based on the Gaussian Scene Representation constructed by the module, optimize the Pose of each image frame. First, the position and orientation of the camera in the current frame are obtained by a dense RGB-D odometry or a constant-velocity motion model. Next, rotation and translation are expressed as quaternions and 3D vectors, respectively, and their minimal parameterised expressions on the Lie group are constructed to serve as the basis for the following optimisation and solution. A differentiable rendering module generates the predicted image, predicted depth map and opacity map of the current frame. According to the above, error functions are constructed as the optimisation objective functions. To increase the stability of the optimisation process, a suitable pixel mask of the depth map is used to assign different weights to each pixel according to its rendered opacity and exclude outliers with large residuals. Adam is a first-order optimisation algorithm that uses gradients to update the pose parameters iteratively during optimisation. After each optimisation iteration, the best pose estimate for the current frame is obtained, and the convergence criterion is checked; if not met, repeat the optimisation; otherwise, terminate according to the set-up stop condition, as shown in Algorithm 1.

Algorithm 1: Camera Pose Optimization

Input:

$\boldsymbol{I}_t, \boldsymbol{D}_t$ ▷ RGB image and depth map of the current frame

$\mathcal{G}$ ▷ Gaussian scene representation

$\boldsymbol{T}_{t-1}$ ▷ Global pose of the previous frame

$\boldsymbol{K}$ ▷ Camera internal reference matrix

Output:

$\boldsymbol{T}_t^*$ ▷ Global pose of the current frame

if IsTextured() **then**

$\boldsymbol{T}_t^{\text{init}} \leftarrow \boldsymbol{T}_{t-1} \cdot \Delta\boldsymbol{T}_{\text{odom}}$ ▷ Rich texture

else

$\boldsymbol{T}_t^{\text{init}} \leftarrow \boldsymbol{T}_{t-1} \cdot \Delta\boldsymbol{T}_{\text{const}}$ ▷ Use constant-speed prior

end if

$\boldsymbol{T}(\boldsymbol{\xi}) \leftarrow \left[\boldsymbol{R}(\boldsymbol{q}), \boldsymbol{t}; \boldsymbol{0}^{\mathrm{T}}, 1\right]$ ▷ Minimal parameterization

repeat

$\boldsymbol{I}_t, \boldsymbol{D}_t, \alpha_t \leftarrow \Re(\mathcal{G}, \boldsymbol{T}(\boldsymbol{\xi}); \boldsymbol{K})$ ▷ Micro-rendering

$\rho(\boldsymbol{u}) \leftarrow \omega(\boldsymbol{u})\left[\lambda\left|\boldsymbol{I}_t(\boldsymbol{u}) - \boldsymbol{I}_t(\boldsymbol{u})\right| + (1-\lambda)\left|\boldsymbol{D}_t(\boldsymbol{u}) - \boldsymbol{D}_t(\boldsymbol{u})\right|\right] \cdot M_{\text{depth}}(\boldsymbol{u}) \cdot M_{\text{outlier}}(\boldsymbol{u})$

$\boldsymbol{\xi} \leftarrow \boldsymbol{\xi} - \boldsymbol{\eta} \cdot$ (AdamGradientStep) ▷ Backprop + Adam step
until convergence
$\boldsymbol{T}_t^* \leftarrow \boldsymbol{T}_{t-1} \cdot \boldsymbol{T}(\boldsymbol{\xi}^*)^{-1}$ ▷ Global pose update

3.3 Map Construction and Optimization

3.3.1 Gaussian Model Initialization

At each of the selected sampling points, a 3D Gaussian function is established at the depth position of the fact in this paper. The first location of the Gaussian function is directly obtained by back-projection of the depth map in world coordinates.

$$\boldsymbol{u}_i = \boldsymbol{T}_{\text{wc}}^{-1} \cdot \left[\frac{u \cdot D(u,v)}{f_x}, \frac{v \cdot D(u,v)}{f_y}, D(u,v), 1 \right]^{\text{T}} \tag{13}$$

The equation under consideration projects the pixel coordinates and the depth value into the 3D world coordinate system, and thus obtains the 3D position of the sampling point in the world coordinate system. At the same time, the initial position of the Gaussian function is set to match the actual scene position observed in the image accurately, and f is the focal length of the camera. To set the scale of the new initialization Gaussian function adaptively based on how far it is from the nearest Gaussian center in the current set of Gaussians: f_x f_y s_i u_j $\mathcal{G}$

$$s_i = \log(\min_{g_i \in \mathcal{G}} \left\| \boldsymbol{p}_i - \boldsymbol{u}_j \right\|_2 + \varepsilon) - 1 \tag{14}$$

ε Where is a real number and how to avoid division by zero? Spatial deduplication check before addition.

$$\mathcal{G}_{\text{new}} = \left\{ g_k \left| \min_{g_j \in \mathcal{G}} \left\| \boldsymbol{u}_k - \boldsymbol{u}_j \right\|_2 > \rho \right. \right\} \tag{15}$$

ρ What is the minimum Distance threshold? New Gaussians that satisfy the conditions are directly added to the Gaussian set of the active subgraph, and their parameters will be used in the following optimisation process.

3.3.2 Sub-map Construction

Every time a new keyframe is added, a new 3D Gaussian function can be added to the current active sub-map to represent the newly observed scene area. Estimate the pose of the current keyframe and, using RGBD data from this frame, generate a dense 3D point cloud. When a new sub-map is started, areas with a large colour gradient in the keyframe point cloud are selected, and several points (set as) are uniformly sampled from these areas to serve as seed positions for the newly added Gaussian function. Subgraph creation is triggered by camera motion in the following keyframe processing: P_a P_b

$$\Delta \boldsymbol{T} = \boldsymbol{T}_t \cdot \boldsymbol{T}_{t-1}^{-1} \tag{16}$$

A new subgraph is created when either the translation threshold exceeds or the rotation threshold exceeds. Data structure Definition for Each Subgraph: α β

$$\boldsymbol{M}_k = (\mathcal{G}_k, \mathcal{F}_k, \mathcal{H}_k) \tag{17}$$

$\mathcal{F}_k$ $\mathcal{H}_k$ Where is a collection of subgraph keyframes? It is a spatial hash structure that speeds up Gaussian search.

All of the newly introduced Gaussian functions are anisotropic Gaussian functions, and their scale parameters are adaptively adjusted based on the distance to the nearest neighbour of a Gaussian point in the current active sub-map.

3.3.3 Dense Point Cloud Reconstruction and Multimodal Adaptive Sampling

A new keyframe arrives, and the system first creates a dense point cloud for the current frame based on the depth map, as shown in the following equation:

$$\mathcal{P} = \left\{ \boldsymbol{T}_{\text{wc}}^{-1} \cdot \left[\frac{u \cdot D(u,v)}{f_x}, \frac{v \cdot D(u,v)}{f_y}, D(u,v), 1 \right]^{\text{T}} \middle| (u,v) \in \Omega \right\} \tag{18}$$

$\mathcal{P}$ $(u,v) \in \Omega$ $D(u,v)$ $\boldsymbol{T}_{\text{wc}}$ Where is the set of all 3D points in the current frame? Is it the image pixel coordinates? Is it the depth value at the pixel point? Is it the pose transformation matrix from the camera to the world coordinate system? The purpose of the above equation is to produce a dense point cloud from a single image by converting the depth information of all pixels into a large number of 3D points in space.

A good way to select some of the representative samples is to build a probability distribution and then randomly select from that distribution. $P(u,v)$

Geometric Gradient Characterization:

$$G_d(u,v) = \left|\partial_x D(u,v)\right| + \left|\partial_y D(u,v)\right| \tag{19}$$

$G_d(u,v)$ Where represent is the gradient intensity of the current pixel in the depth map, which is used to detect geometric edges, and $\text{ are the derivatives of the depth map in the and directions . } \partial_x D$ $\partial_y D$ x y

Color Gradient Features

$$G_c(u,v) = \frac{1}{3} \sum_{c \in \{R,G,B\}} \left(\left|\partial_x I_c(u,v)\right| + \left|\partial_y I_c(u,v)\right| \right) \tag{20}$$

$I_c(u,v)$ $G_c(u,v)$ The intensity of the pixel in the colour channel is used to indicate colour gradient response, and thus how strongly and where a change in the image texture occurs.

Sobel Edge Features:

$$E(u,v) = \sum_c \sqrt{(I_c * S_x)^2 + (I_c * S_y)^2} \tag{21}$$

S_x S_y $* E(u,v)$ Where is a Sobel operator used to detect horizontal and vertical edges, respectively. is a convolution operation. is the Sobel Edge Response. After normalisation, the above features are combined to create a single information response map:

$$F(u,v) = \frac{G_d}{\max G_d} + \frac{G_c}{\max G_c} + \frac{E}{\max E} \tag{22}$$

$F(u,v)$ Where can we see the response features of the fusion? Next, construct a normalised sampling probability distribution:

$$P(u,v) = \frac{F(u,v)}{\sum_{(u',v')\in\Omega} F(u',v')} \tag{23}$$

$P(u,v)$ $F(u,v)$ $F(u,v)$ (u,v) $\sum_{(u',v')\in\Omega} F(u',v')$ The above equation is employed to build a normalised sampling probability distribution, and its purpose is to achieve adaptive sampling of image pixels based on the fusion response feature. Represents the fusion response value of a particular pixel, which indicates how much information or importance that pixel has in the current image; represents the sum of the fusion response values of all pixels in the image, which serves as a normalisation factor for the total response. The design will give a larger weight to the sampling probability of a pixel that has a high response value and a smaller weight to a pixel with a low response value. This way can also reduce the computational burden and improve both the speed of dense point cloud reconstruction and operation. A typical set of pixels is finally obtained from this distribution:

$$\mathcal{S} = \{(u_i, v_i) \sim P(u,v) | i = 1, ..., N_s\} \tag{24}$$

$\mathcal{S}$ N_s Where is the set of coordinates of the selected pixels, and how many samples are there? Thus, a Gaussian coverage of low-texture areas has been improved.

3.3.4 Sub-map Optimization

Exponential Decay Weighting:

To enhance the reconstruction quality and tracking accuracy, an exponential decay weighting strategy in the optimisation process is used. After adding new keyframes, several iterations of optimisation are carried out for each activity subgraph, and at this time, a multitask loss function is established as follows:

$$\mathcal{L} = w_c \mathcal{L}_{\text{color}} + w_d \mathcal{L}_{\text{depth}} + w_p \mathcal{L}_{\text{percep}} + w_r \mathcal{L}_{\text{reg}} \tag{25}$$

$w_c = w_r = 1$ w_d w_p Where \$\omega\$, λ_1 and λ_2 are the weight hyper-parameters of the respective loss terms. Rather than fixing the weights during the optimisation process, the exponential decay weighting strategy dynamically changes them to prioritise recent observations and gradually reduce the influence of older data. The initial weights and decay coefficients of all loss terms are set as follows: At each iteration of the update, reduce the weight by. After a few iterations, the weight can be expressed as ω_k, and it will decrease exponentially. This adaptive weighting gives more weight to the recent frames in the optimisation and thus helps to improve reconstruction quality and accuracy of tracking. w^0 $\gamma \in (0,1)$ $w^{(n+1)} = \gamma \cdot w^{(n)}$ t $w^{(t)} = w^{(0)} \cdot \gamma^t$

Colour loss.

$$\mathcal{L}_{\text{color}} = (1-\lambda)\cdot\left\|\hat{I}-I\right\|_1 + \lambda\cdot\frac{1-\text{SSIM}(\hat{I},I)}{2} \tag{26}$$

$\hat{I}$ I Where, are the rendered image and real image; is the weighting factor of and . λ SSIM $\|\cdot\|_1$

Depth consistency loss:

$$\mathcal{L}_{\text{depth}} = \left\|D-D\right\|_1 \tag{27}$$

Perceptual Loss:

L_1 L_2 $\hat{I}$ I $\phi_l(\cdot)$ To enhance the perceptual quality of the reconstructed images further, a perceptual loss based on the feature space of the VGG16 network is introduced during optimisation. Different from the traditional method of only using the difference in pixel space, perceptual loss can consider the discrepancy between the predicted and actual images at a higher level in the semantic feature space to better retain texture details and structural information of the images. A VGG16 model pre-trained on the ImageNet dataset is used, and the first 16 convolutional layers of this model are selected as the perceptual feature extractor. Normalise the predicted and actual images during the calculation process, and then pass them through a parameter-frozen VGG16 network to extract feature maps at various levels. Finally, considering the differences in feature layers, a perceptual loss is designed to assist the model in reconstructing the visual quality of the image more accurately during training.

$$\mathcal{L}_{\text{percep}} = \sum_{l\in\mathcal{L}}\left\|\phi_l(\hat{I})-\phi_l(I)\right\|_F^2 \tag{28}$$

$\phi_l(\cdot)$ l $\mathcal{L}$ 'denotes the feature map extracted by the -th convolutional layer, and S is the selected set of feature layers.' Perceptual loss can be used to better address the high-level semantic differences that are more noticeable to the human eye than pixel-level loss, and reconstruction quality in areas such as edges, textures, and high-reflectivity spots has been significantly improved. At this time, the perceptual loss is jointly optimised with the colour-reconstruction loss, depth loss and regularization loss under different weights to construct a total loss function. An exponential decay strategy is used to dynamically adjust the weights of all components, and earlier in training, more weight is given to perceptual constraints for structure and detail. Optimisation for pixel accuracy and other general consistency has gradually been carried out in the later stage.

Scale Normalisation:

$$\mathcal{L}_{\text{reg}} = \frac{1}{\mathcal{G}}\sum\left\|\boldsymbol{s}_i-\bar{\boldsymbol{s}}\right\|_1 \tag{29}$$

where $\bar{\boldsymbol{s}} = \frac{1}{\mathcal{G}}\sum\boldsymbol{s}_i$

The three stages of sub-map optimisation are progressive. The first 30% of the total iteration time is a geometric convergence period that mainly reduces the depth reconstruction loss; the other weightings are not given. Early transparency pruning will also be used at this time to speed up the construction of scene geometry. The proportion of the texture optimisation step is 40% of the total iteration. A perceptual loss with a weight of 0.2 and VGG-extracted features are used to enhance the realism and texture detail of the image. In the final 30 per cent stage of

the iteration process, a detection scale for outliers is employed, and this is used to perform the final pruning; at the same time, the scale variance is reduced to be less than 0.1. S.t. $\lambda_a \ \alpha < S_a$ $|s-u| > 3\sigma \ \alpha < S_b$

$$\begin{cases} \dfrac{\left|\mathcal{L}^{(t)} - \mathcal{L}^{(t-10)}\right|}{\mathcal{L}^{(t)}} < \tau_{\text{rel}} \\ \left\|\nabla \mathcal{L}\right\|_2 < \tau_{\text{grad}} \\ t \geq t_{\max} \end{cases} \tag{30}$$

First, the proposed method converts the pixel coordinates and depth values of the depth map into 3D world coordinates via projection to initialise the Gaussian model. At these positions, 3D Gaussian functions are used for initialization, and their initial scales are adaptively set according to the distance to the nearest Gaussian center. Filter by a lower bound on the distance, and then add the result to the current sub-map. At the time of adding a new keyframe in the sub-map construction phase, a dense point cloud is generated from its RGB-D data. Points with high feature values are selected as Gaussian seed points. If the change in camera position exceeds a certain range of translation or rotation, then a new sub-map will be generated. Depth maps are employed in the dense point cloud reconstruction and multimodal adaptive sampling stage to create the current frame's dense point cloud. Geometric gradients, colour gradients and Sobel edge features are extracted and fused to form a sampling probability distribution, thus achieving priority sampling of high-information-content areas. Finally, in the sub-map optimisation stage, Gaussian parameters are jointly optimised with colour loss, depth consistency loss and VGG perceptual loss, and exponentially decaying weights are dynamically adjusted for these losses. At this time in the iteration process, a scale-regularisation and outlier-removal strategy will be employed for final adjustment to preserve fine details and ensure global consistency, as shown in Algorithm 2.

Algorithm 2: Map Construction and Optimization

Input:

$\boldsymbol{I}_t, \boldsymbol{D}_t, \boldsymbol{K}, \mathcal{G}, \boldsymbol{T}_t, \boldsymbol{T}_{t-1}$

Output:

$\{\mathcal{S}\}$

for each selected pixel (u, v) **do** ▷ Gaussian Initialization

$\boldsymbol{u}_i \leftarrow \boldsymbol{T}_{\text{wc}}^{-1}\left[u \cdot D(u,v) / f_x, v \cdot D(u,v) / f_y, D(u,v), 1\right]^{\text{T}}$

$s_i \leftarrow \log(\min_{g_i \in \mathcal{G}} \left\|\boldsymbol{p}_i - \boldsymbol{u}_j\right\|_2 + \varepsilon) - 1$

if $G_{new} = \left\{ g_k \left| \min_{g_j \in \mathcal{G}} \left\|\boldsymbol{u}_k - \boldsymbol{u}_j\right\|_2 > \rho \right. \right\}$ **then**

AddGaussian($\boldsymbol{u}_i, S_i$)

end if

end for

for each new keyframe **do** ▷ Sub-map Construction

$\mathcal{P} \leftarrow$ ComputeDensePointCloud($\boldsymbol{D}_t, \boldsymbol{T}$)

$(P_a, P_b) \leftarrow$ SampleHighGradientPoints($\mathcal{P}$)
for $u \in (P_a, P_b)$ **do**
  InitialzeAnisotropicGaussian($\boldsymbol{u}$)
end for
$\Delta \boldsymbol{T} \leftarrow \boldsymbol{T}_{\text{current}} \boldsymbol{T}_{\text{last}}^{-1}$
if Trans(ΔT) > α or Rot(ΔT) > β **then**
  Initialize new sub-map:
    $\boldsymbol{M}_k = (\mathcal{G}_k, \mathcal{F}_k, \mathcal{H}_k)$
end if
end for
for each frame **do** ▷ Adaptive Sampling
$G_d \leftarrow |\partial_x D(u,v)| + |\partial_y D(u,v)|$
$G_c \leftarrow \sum_{c \in \{R,G,B\}} (|\partial_x I_c(u,v)| + |\partial_y I_c(u,v)|)$
$E \leftarrow \sum_c \sqrt{(I_c * S_x)^2 + (I_c * S_y)^2}$
$R(u) \leftarrow \text{Normalize}(G_d + G_c + \text{E})$
$P(u) \leftarrow R(u) / \sum_u R(u)$
$S \leftarrow \{(u_i, v_i) \sim P(u,v) | i = 1, \ldots, N_s\}$
end for
for each active sub-map **do** ▷ Sub-map Optimization
for t = 1 to T **do**
$\mathcal{L} = w_c \mathcal{L}_{\text{color}} + w_d \mathcal{L}_{\text{depth}} + w_p \mathcal{L}_{\text{percep}} + w_r \mathcal{L}_{\text{reg}}$
$w^{(t)} \leftarrow w^{(0)} \cdot \gamma^t$
if t ∈ [0%, 30%] **then**
$\lambda \leftarrow \lambda_a$
PruneGaussians(α)
else if t ∈ [30%, 70%] **then**
$\lambda \leftarrow \lambda_p$
ExtractFeaturesWithVGG()
else
Prune($|s - u| > 3\sigma$)
Ensure $\text{Var}(\alpha) < 0.1$
end if
BackpropagateAndUpdate()
end for
end for

4 Experiments

To comprehensively evaluate the rendering quality and reconstruction accuracy of the proposed method, three experiments are designed and carried out in this paper. The first experiment will compare the rendering performance and tracking accuracy of the proposed method with those of NICE-SLAM, Point-SLAM, SplaTAM, and Gaussian-SLAM to this end. The second experiment is an ablation study to investigate the impact of omitting adaptive sampling, perceptual loss functions or exponential decay weighting strategies on rendering performance and tracking accuracy separately. The third experiment is to determine how much more memory this method and the basic method use.

4.1 Experimental Setup

A high-end desktop computer is used for the experiments, and it has an NVIDIA RTX 3090 24GB GPU, a Xeon Silver 4210R processor, and 64GB of RAM. Run the experimental code in Ubuntu 20.04.

4.1.1 Datasets

The three commonly used RGB-D datasets covering scenes of different scales and environments in this paper are Replica (https://cvg-data.inf.ethz.ch/nice-slam/data/Replica.zip), TUM RGB-D (https://vision.in.tum.de/rgbd/dataset/freiburg1/rgbd_dataset_freiburg1_desk.tgz, https://vision.in.tum.de/rgbd/dataset/freiburg2/rgbd_dataset_freiburg2_xyz.tgz, https://vision.in.tum.de/rgbd/dataset/freiburg3/rgbd_dataset_freiburg3_long_office_household.tgz, and ScanNet [2615] (https://github.com/ScanNet/ScanNet). Replica is a good high-quality indoor scanning dataset with realistic environments and camera trajectories. The Replica dataset is a high-precision 3D reconstruction dataset of many typical indoor environments that includes trajectories of RGB-D sensors and is suitable for research in localisation and mapping. To further confirm the actual performance of the above framework, the well-known TUM-RGBD and ScanNet datasets are also used as test cases. Precise camera poses in the TUM-RGBD dataset are obtained from an external motion-capture system, and poses in the ScanNet dataset are estimated by the BundleFusion method with high reliability. Therefore, six scenes from the Replica dataset are selected as follows: Room0 (R0), Room1 (R1), Room2 (R2), Office0 (O0), Office1 (O1), and Office4 (O4). Three scenes have been selected from the TUM RGBD dataset: rgbd_dataset_freiburg1_desk (fr1/desk), rgbd_dataset_freiburg2_xyz (fr2/xyz), and rgbd_dataset_freiburg3_long_office_household (fr3/office). Three scenes from the ScanNet dataset are selected as the examples: 0000_00 (0000), 0106_00 (0106), 0169_00 (0169), and 0207_00 (0207).

4.1.2 Metrics

As shown in Table 1, the quantity to measure the difference in pixels between the reconstructed picture and the original picture is peak signal-to-noise ratio (PSNR). The images are more similar, and thus the PSNR is higher.

SSIM is an indicator of how close two pictures are in terms of alterations to light, shadow, contrast and structure, etc. The range of its values is [0, 1]. A higher value is closer to 1, and thus the two structures of the images are more similar.

Learning Perceptual Image Patch Similarity (LPIPS) is a way that deep neural networks are used to extract image features and evaluate the perceptual distance between two images. A relatively small LPIPS value indicates that the reconstructed image is close to the original perceptually.

Absolute Trajectory Error (ATE) and Root Mean Square Error (RMSE) are employed to measure how far off the estimated path of SLAM is from the actual path. A lower value indicates that the position of the camera is relatively closer to the ideal. All the experimental data shown in this paper are the averages of five experiments. The data in bold are the best-performing results.

Table 1: Evaluation Metrics and Comparison of Baseline SLAM Methods

	Metrics	Results	Comparison of Baseline Methods
Rendering Performance	PNSR	Up	NICE-SLAM, Point-SLAM, SplaTAM, Gaussian-SLAM
	SSIM	Up	
	LPIPS	Up	
Tracking accuracy	ATE(RMSE)	Up	NICE-SLAM, Point-SLAM, SplaTAM, Gaussian-SLAM

4.1.3 Parameter Settings

Table 2 shows the key implementation parameters and configuration settings of the SLAM system, as well as specific values and optimisation methods for memory management, keyframe strategy, candidate point filtering, neighbourhood search, and other parts.

Table 2: Implementation Parameters and Settings of the SLAM System

Parameter category	Parameter name	Value
Memory management	Maximum limit for Gaussian point caching (Replica)	600,000
	Maximum limit for Gaussian point caching (TUM/ScanNet)	100,000
Number of iterative optimizations	Initial keyframe optimization (Replica)	1000
	Initial keyframe optimization (TUM/ScanNet)	100
Keyframe strategy	Keyframe Sampling Frequency	Sample 1 frame every five frames
Candidate Point Filtering	Depth confidence threshold α	0.6
Neighborhood Search	FAISS Acceleration	Accelerating Nearest Neighbor Search with the GPU
	Neighborhood search radius	0.01m
Sub-map strategy	Subgraph trigger threshold: translation	0.5m
	Subgraph trigger threshold: angle	50o
Optimization Weight Adjustment	Color, regularization loss weight	1
	Depth, perceptual loss, weight	exponential decay function
New keyframe iteration	New keyframe minimum mapping iteration ratio	>= 40%
Reconstruction	TSDF fusion element size	1cm
	TSDF Fusion Frame Selection	Every five frames

4.2 Rendering performance and Tracking accuracy Evaluation

Render and Tracking Performance of Gaussian Sputtering-based Dense SLAM Against Current State-of-the-Art Methods: NICE-SLAM, Point-SLAM, SplaTAM and Gaussian-SLAM. Among them, the data for NICE-SLAM and Point-SLAM are derived from the results in the Gaussian-SLAM [23] paper.

4.2.1 Rendering Performance

Visual quality evaluation using the Replica, TUM RGBD and ScanNet datasets in this paper. First, it compares the proposed method with that of a general-purpose tool to show that the new one is better for the R2 and O0 scenes of Replica.

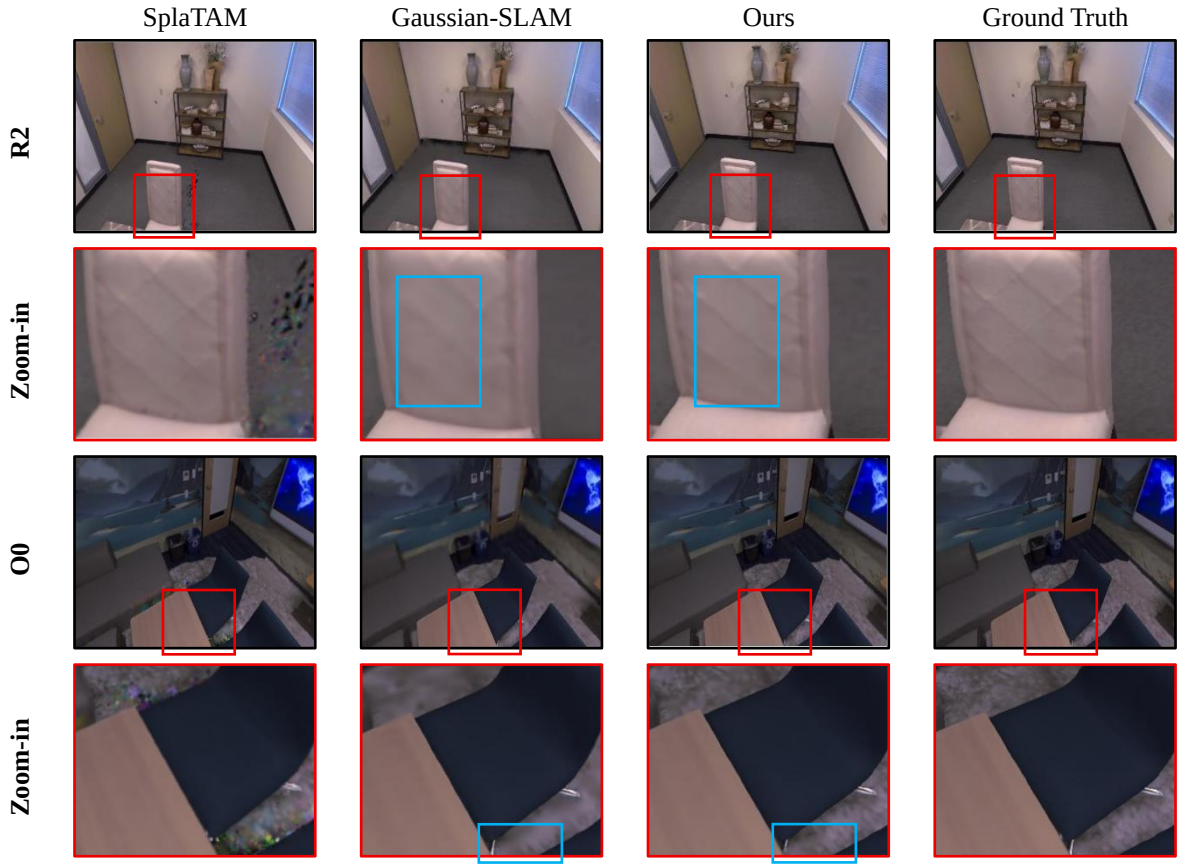


Figure 2: Comparison of Rendering Performance for Several SLAM Methods in the R2 and O0 Scenes.

As shown in the R2 and O0 scenes, the proposed method has improved both the clarity of chair seams and the realism of floor textures over SplaTAM and Gaussian SLAM (red and blue boxes). The increase in lightness and the enhancement of contrast in fine-structure and textured Areas are relatively larger; otherwise, the seams and texture distortions would be less obvious. To solve the above problems, the three new features of this system are introduced: adaptive sampling to expand sampling areas in areas with rich feature density and optimize sampling intensity in less feature-rich zones. Exponential decay weighting to give more weight to recent observations for improved reconstruction and tracking in dynamic scenarios, and a perceptual loss function to use semantic features for better preservation of details and surface quality. Together, these modules have improved the quality of detail preservation, texture clarity and boundary alignment compared with the previous ones.

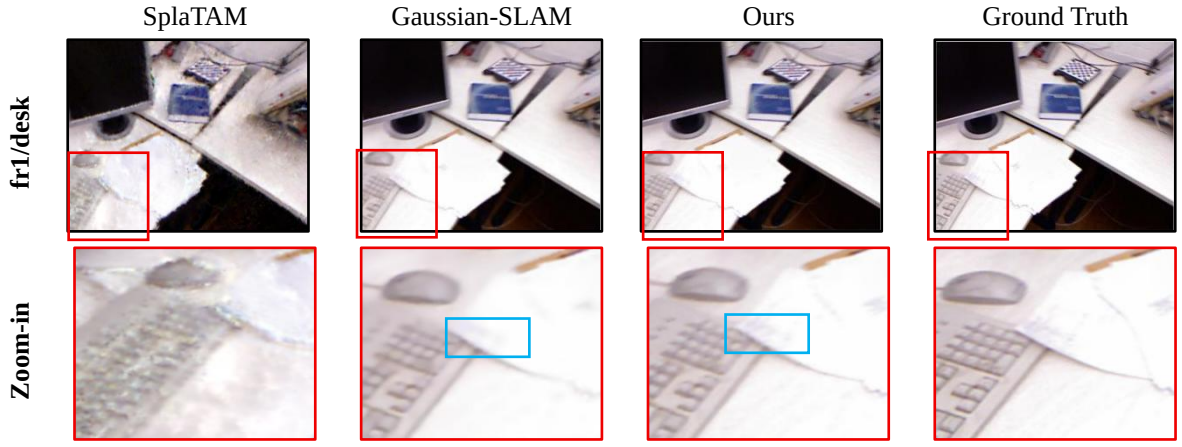
Table 3 is the rendering performance on Replica.

Table 3: Rendering Performance Comparison of Replica

Method	Metric	R0	R1	R2	O0	O1	O4	Avg
NICE-SLAM [20]	PSNR	22.12	22.47	24.52	29.07	30.34	24.94	25.57
	SSIM	0.689	0.757	0.814	0.874	0.886	0.856	0.812
	LPIPS	0.330	0.271	0.208	0.229	0.181	0.198	0.236
Point-SLAM [21]	PSNR	32.40	34.08	35.50	38.26	39.16	33.49	35.48
	SSIM	0.974	0.977	0.982	0.983	0.986	0.979	0.980
	LPIPS	0.113	0.116	0.111	0.100	0.118	0.142	0.116
SplaTAM [22]	PSNR	32.65	33.51	35.29	38.10	38.85	32.14	35.09
	SSIM	0.975	0.967	0.984	0.981	0.981	0.951	0.973
	LPIPS	0.071	0.100	0.072	0.089	0.092	0.150	0.095
Gaussian-SLAM [23]	PSNR	38.80	41.63	42.20	46.20	45.20	42.56	42.76
	SSIM	0.992	0.995	0.996	0.997	0.996	0.996	0.995
	LPIPS	0.020	0.016	0.019	0.012	0.013	0.017	0.016
Ours	PSNR	39.30	42.07	42.63	46.63	45.62	42.95	43.20
	SSIM	0.994	0.998	0.997	0.998	0.997	0.998	0.997
	LPIPS	0.019	0.014	0.016	0.010	0.010	0.014	0.013

The R2 scene has a small texture on the wall and floor, and because of a large change in viewing angle, there is a significant reconstruction problem. SplaTAM often has blurring and misalignment in the textured areas; its LPIPS is 0.072 and PSNR is 35.29 dB. Gaussian SLAM is generally good, but it lacks a limited-update mode and is prone to gradient instability and local discontinuities after abrupt viewpoint changes. The three innovations of the new way are as follows. Adaptively sample to reduce the number of sampling points for complex areas in dense textures. An exponential decay weight is given to the most recent observations, and stability of reconstruction and tracking after a sudden change is enhanced. Perceptual loss has captured semantic information and maintains fine details. Based on the results of the experiments, the proposed method has achieved a PSNR of 42.63 dB and a reduction in LPIPS of 0.016; it is better than others in regions with rich textures and during swift changes in view angle.

Second, it performs a comparative experiment with the main method to verify the superiority of the proposed approach in the fr1/desk and fr3/office scenes of TUM_RGBD.



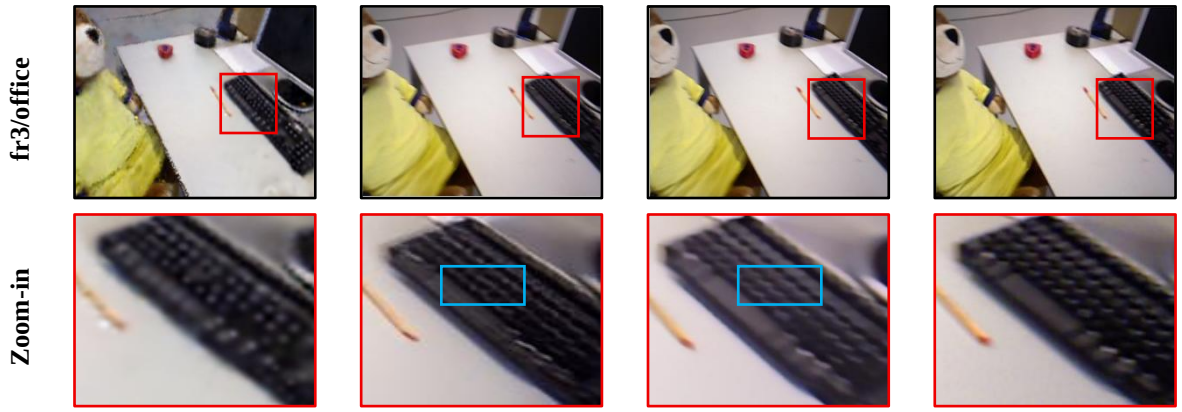


Figure 3: Comparison of Rendering Performance for Several SLAM Methods in the fr1/desk and fr3/office Scenarios.

In the fr1/desk and fr3/office scenes, the proposed method can reconstruct clearer boundaries and finer details than SplaTAM and Gaussian-SLAM, especially in the areas of the keyboard and documents. The above reasons explain why this is not possible: the environment of the scene contains numerous small and tiny items or mirrors that complicate the application of these technologies. The reasons for the good performance of the above method are as follows: an adaptive sampling strategy; the use of exponential decay weights; and a perceptual loss function. The above can improve the accuracy and detail retention of the reconstruction.

Table 4 shows the rendering performance results for TUM RGB-D.

Table 4: Rendering Performance Comparison on TUM RGB-D

Method	Metric	fr1/desk	fr2/xyz	fr3/office	Avg
NICE-SLAM [20]	PSNR	13.83	17.87	12.89	14.86
	SSIM	0.569	0.718	0.554	0.613
	LPIPS	0.482	0.344	0.498	0.441
Point-SLAM [21]	PSNR	13.87	17.56	18.43	16.62
	SSIM	0.627	0.708	0.754	0.696
	LPIPS	0.544	0.585	0.448	0.525
SplaTAM [22]	PSNR	21.79	22.87	21.92	22.19
	SSIM	0.852	0.858	0.873	0.861
	LPIPS	0.242	0.192	0.207	0.213
Gaussian-SLAM [23]	PSNR	25.12	23.31	26.14	24.85
	SSIM	0.937	0.918	0.939	0.931
	LPIPS	0.146	0.203	0.126	0.158
Ours	PSNR	25.60	23.85	26.97	25.47
	SSIM	0.942	0.929	0.946	0.939
	LPIPS	0.133	0.170	0.105	0.136

Frequent depth changes in the fr2/xyz scene cause blurring in SplaTAM, and Gaussian-SLAM fails to perform adaptive sampling allocation, thus resulting in incomplete structures. The reasons for the good results of the above three reasons are as follows: 1) an adaptive sampling strategy that improves the detail reconstruction accuracy; 2) an exponentially decaying weight function for stable optimisation; and 3) a perceptual loss function that keeps semantic and edge information. Therefore, a PSNR of 23.85 dB and an LPIPS of 0.170 have

been achieved; both surpass those of the baseline method in terms of accuracy and perceptual quality.

Finally, it is compared with the above two typical ways to demonstrate that the proposed method is superior in the 0000 and 0207 scenes of ScanNet.

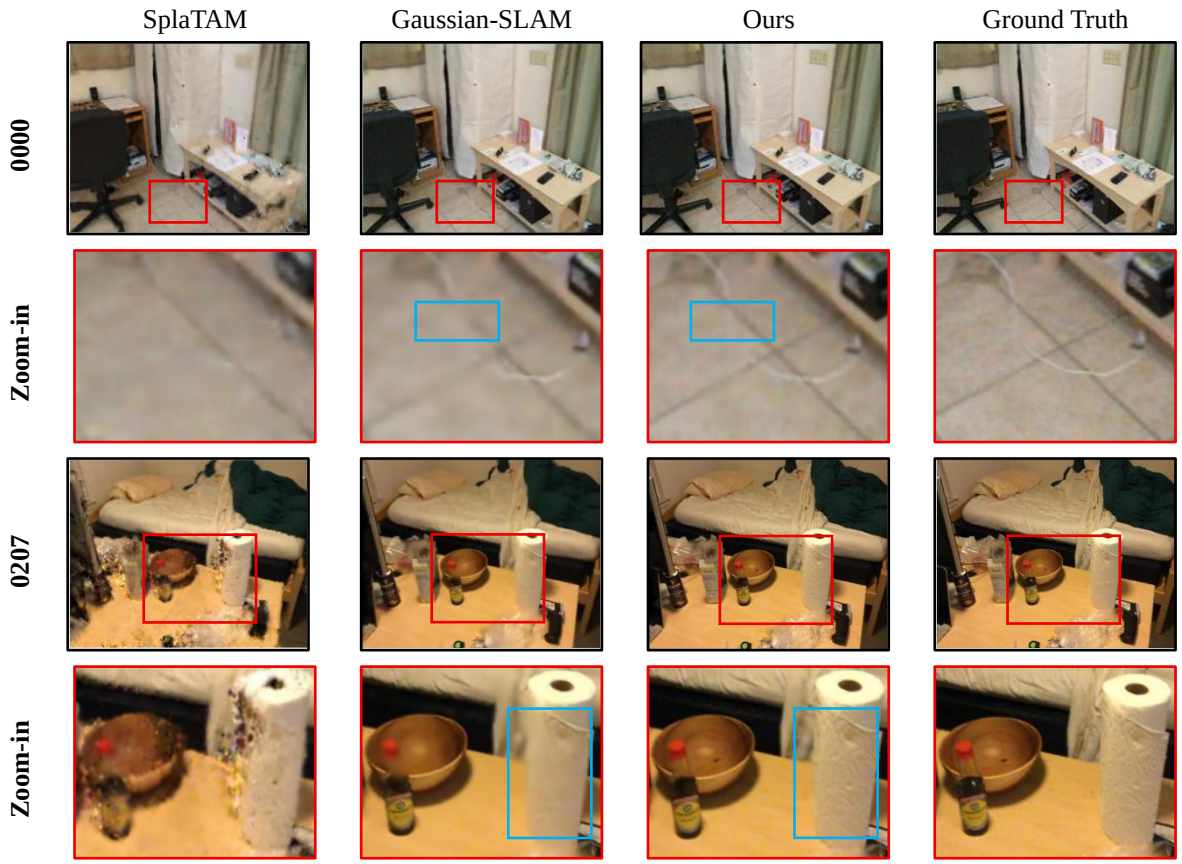


Figure 4: Comparison of Rendering Performance of Multiple SLAM Methods in the 0000 and 0207 Scenes.

In the 0000 and 0207 scenes, the proposed method achieves a finer-grained reconstruction of textures for electrical wires, desktop items, and toilet paper than SplaTAM and Gaussian SLAM. The analysis of the above scenes is difficult because there are many thin structures, cluttered objects, and reflective surfaces. These factors often cause a lack of clarity in the other ways. By combining adaptive sampling strategies, exponential decay weights and perceptual loss functions, it has been shown that the accuracy of the method can be improved, the stability of the optimisation process can be guaranteed, and finer details can be retained.

Table 5 shows the rendering performance results for ScanNet.

Table 5: Rendering Performance Comparison for ScanNet

Method	Metric	0000	0169	0207	Avg
NICE-SLAM [20]	PSNR	18.71	18.75	18.38	18.61
	SSIM	0.641	0.629	0.646	0.638
	LPIPS	0.561	0.534	0.552	0.549
Point-SLAM [21]	PSNR	21.30	18.53	20.56	20.13
	SSIM	0.806	0.686	0.750	0.747
	LPIPS	0.485	0.542	0.544	0.523
SplaTAM [22]	PSNR	18.01	22.56	19.63	20.06
	SSIM	0.621	0.794	0.683	0.699
	LPIPS	0.464	0.271	0.343	0.359
Gaussian-SLAM [23]	PSNR	28.50	28.53	28.34	28.45
	SSIM	0.923	0.914	0.907	0.914
	LPIPS	0.273	0.229	0.296	0.266
Ours	PSNR	29.04	29.30	29.20	29.18
	SSIM	0.927	0.923	0.912	0.920
	LPIPS	0.250	0.195	0.264	0.236

SplaTAM has a low performance in the 0169 scene because of a high density of structures, frequent occlusion and narrow passages; therefore, fine-detail reconstruction is difficult. Gaussian-SLAM has a fixed sampling scheme and is therefore also constrained. The three reasons for the three main innovations are as follows: (1) to improve the accuracy of edge reconstruction, an adaptive sampling strategy is introduced; (2) to address sudden changes in viewpoints more effectively, exponentially decaying weights are employed; (3) to maintain semantic consistency and fine texture, a perceptual loss function is added.

4.2.2 Tracking Accuracy Performance

Evaluation of Tracking Accuracy with Replica and TUM RGBD Datasets. First, it performs a comparative experiment with the main methods to verify the superiority of the proposed approach in the R2 and O0 scenes of Replica.

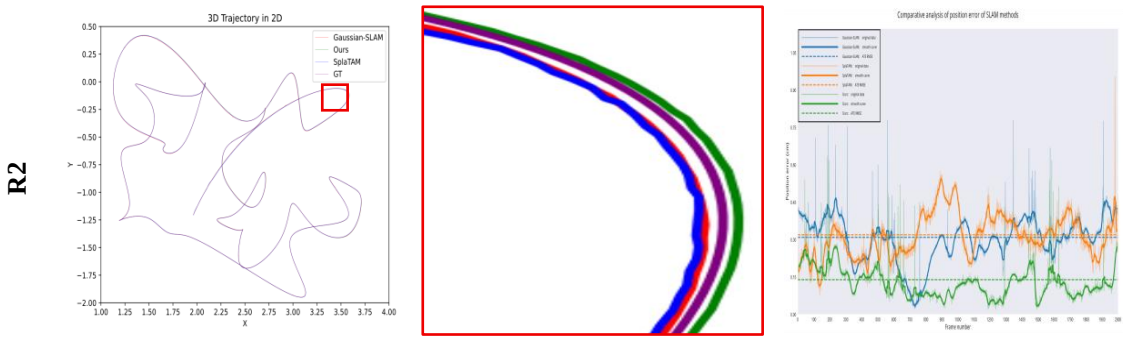


Figure 5: Comparison of tracking accuracy and position error per-frame for multiple SLAM methods in the R2 and O0 scenes.

The proposed method has better tracking accuracy and stability than SplaTAM and Gaussian SLAM in the R2 and O0 scenes. As shown in Figure 54, the trajectories are closer to the ground truth and the error curves are relatively small and stable for all methods; others have deviated more significantly and shown large fluctuations. Complex motion, occlusion and rapid changes in viewpoint are typical causes of drift and instability for traditional SLAM. The problems mentioned above have been solved by adaptive sampling to improve reconstruction quality in textured and cluttered areas, exponential decay weighting to prioritise recent reliable observations, and perceptual loss to retain fine details such as walls, edges and small objects. Together, these new additions will make the tracking more reliable in difficult circumstances. Table 6 is the tracking accuracy results for Replica.

Table 6: Tracking Accuracy Comparison on Replica ATE (cm)

Method	R0	R1	R2	O0	O1	O4	Avg
NICE-SLAM [20]	1.69	2.04	1.55	0.99	0.90	3.08	1.70
Point-SLAM [21]	0.61	0.41	0.37	0.38	0.48	0.63	0.48
SplaTAM [22]	0.27	0.47	0.32	0.54	0.25	0.54	0.39
Gaussian-SLAM [23]	0.45	0.32	0.31	0.33	0.23	0.52	0.36
Ours	0.40	0.27	0.14	0.29	0.18	0.32	0.26

Existing methods are generally unable to retain the finer details in the R2 scene due to its complex textures and sudden changes in the viewing direction. SplaTAM has an ATE (RMSE) of 0.32 cm, but blurring and misalignment of high-texture areas still occur. Gaussian SLAM has a relatively large overall ATE (RMSE) of 0.31 cm and is more likely to become unstable after an abrupt change in viewpoint. On the other hand, our proposed method has achieved the lowest ATE (RMSE) of 0.14 cm by introducing the following three new ideas: (1) An adaptive sampling strategy that increases the sampling density in texture-rich areas; (2) An exponential decay weighting scheme to enhance tracking accuracy by giving more weight to recent observations; (3) A perceptual loss function that improves the reconstruction of fine details in the semantic feature space. The above innovations are to be introduced to maintain high tracking accuracy in complex scenes.

Second, it is compared with the mainstream methods to verify the advantage of the proposed approach in the fr1/desk and fr3/office scenes of TUM_RGBD.

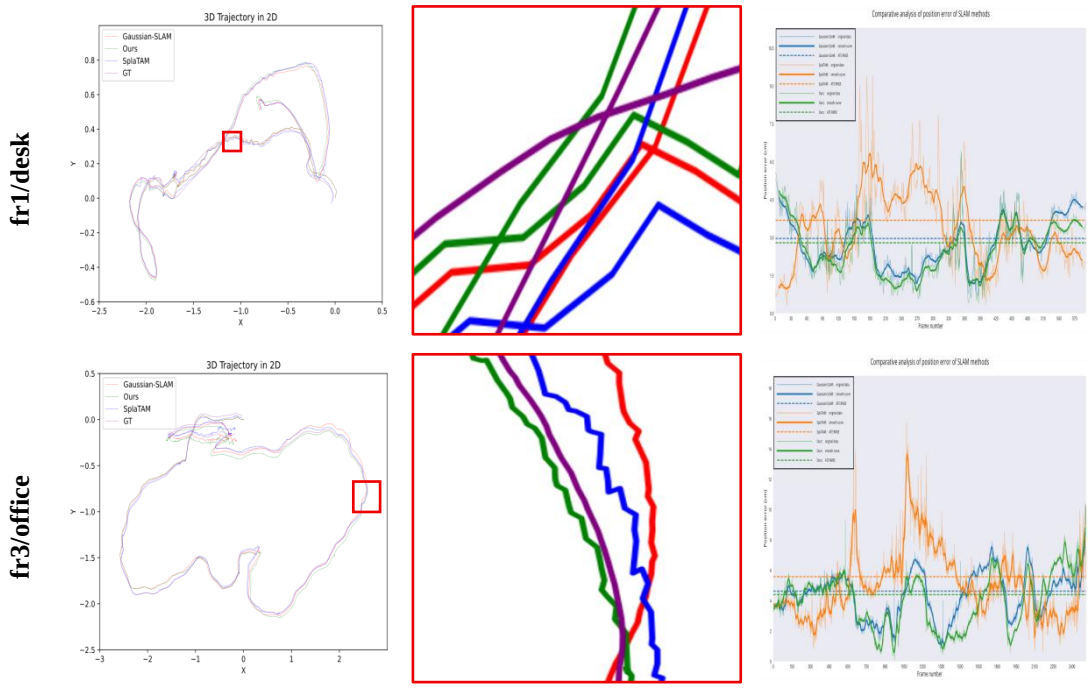


Figure 6: Comparison of tracking accuracy and position error per-frame for multiple SLAM methods in the fr1/desk and fr3/office scenes.

In the fr1/desk and fr3/office scenes, the trajectories generated by the proposed method are closer to the true trajectories than those from SplaTAM and Gaussian-SLAM, with smaller and more stable errors. The existing methods have problems in difficult environments and frequent occlusions; therefore, feature matching and optimisation fail. Three first-class improvements have been made by the above method. First, a dynamic sampling mode will be used to improve the accuracy of reconstruction in the above areas. Second, to improve the accuracy of tracking, an exponential-decaying weight will be used. Thirdly, a perceptual loss function is added to keep the fine details and visual quality of the images. The above improvements are expected to have a high-accuracy tracker in difficult scenarios.

Table 7 shows the tracking accuracy results in TUM RGB-D.

Table 7: Tracking Accuracy Comparison on TUM RGB-D ATE (cm)

Method	fr1/desk	fr2/xyz	fr3/office	Avg
NICE-SLAM [20]	4.30	31.7	3.90	13.3
Point-SLAM [21]	4.30	1.30	3.50	3.03
SplaTAM [22]	3.69	1.34	5.59	3.54
Gaussian-SLAM [23]	2.97	1.46	4.64	3.02
Ours	2.80	1.44	4.41	2.88

Frequently change the view in the fr3/office scene, and large-scale structures are also present. SplaTAM had an error of 5.59cm, and Gaussian-SLAM had an error of 4.64cm. Both of these measures are also subject to distortion and drift. By adding adaptive sampling, an optimized exponential decay weight function, and a loss function that maintains the structure and texture of the region, this way can reduce the error and increase the adaptability of the method in complex scenes.

Table 8 shows the average values of all methods' various metrics based on all the datasets. Performance improvement is the average increase of the proposed method over each of the

comparison methods in terms of various indicators: NICE-SLAM (NS), Point-SLAM (PS), SplaTAM (ST), Gaussian-SLAM (GS).

Table 8: Improved Performance in All Indicators

Metric	NS [20]	PS [21]	ST[22]	GS [23]	Ours	Improvement (Avg)
PSNR	19.68	24.07	25.78	32.02	32.61	32.38%
SSIM	0.687	0.807	0.844	0.946	0.952	17.48%
LPIPS	0.408	0.388	0.222	0.146	0.128	47.57%
ATE(RMSE)	7.500	1.755	1.965	1.690	1.570	29.20%

4.3 Ablation Evaluation

To assess the impact of the innovations proposed in this paper on system performance, three ablation studies are conducted here to determine how much each module contributes to the overall results by omitting it and observing any changes in performance after its exclusion, without altering the other components. It then re-runs the above studies to compare the differences, such as adaptive sampling (as), exponential decay weighting (edw), perceptual loss (pl), etc.

Table 9 shows the detailed rendering performance of ablation studies on Replica.

Table 9: Rendering Performance Comparison of Ablation Study on Replica

Method	Metric	R0	R1	R2	O0	O1	O4	Avg
Ours (no as)	PSNR	38.51	41.71	42.45	46.39	45.37	42.47	42.81
	SSIM	0.991	0.995	0.996	0.997	0.996	0.996	0.995
	LPIPS	0.020	0.016	0.018	0.011	0.013	0.017	0.015
Ours (no edw)	PSNR	38.50	41.48	42.47	46.44	45.20	42.66	42.79
	SSIM	0.991	0.995	0.996	0.997	0.996	0.996	0.995
	LPIPS	0.021	0.018	0.019	0.011	0.015	0.017	0.016
Ours (no pl)	PSNR	38.87	41.42	42.61	46.22	45.42	42.53	42.84
	SSIM	0.992	0.995	0.996	0.997	0.997	0.996	0.995
	LPIPS	0.019	0.017	0.018	0.011	0.013	0.016	0.015
Ours	PSNR	39.30	42.07	42.63	46.63	45.62	42.95	43.20
	SSIM	0.994	0.998	0.997	0.998	0.997	0.998	0.997
	LPIPS	0.019	0.014	0.016	0.010	0.010	0.014	0.013

The R0 scene has several objects and rich textures, and with the application of the proposed method, the best reconstruction results have been obtained, with a PSNR of 39.30 dB, an SSIM of 0.994, and an LPIPS of 0.019. This paper shows that removing adaptive sampling reduces PSNR by 0.79 dB, and there are slight declines in SSIM and LPIPS; thus, it can be seen that the allocation of sampling points to complex geometric and textured areas is not optimal. Without exponential decay weighting, both PSNR is 0.8 dB lower and LPIPS is 0.021 higher; therefore, to maintain the reconstruction accuracy of a difficult scene, we need to weigh more recent data more heavily. Excluding the perceptual loss results in a 0.43 dB drop in PSNR and less semantic consistency; thus, fine-texture preservation and a high-quality visual are still needed. Based on the above results, it can be seen that adaptive sampling, exponential decay weighting and perceptual loss can be used together to improve the accuracy and visual quality of reconstruction.

Table 10 shows the ablation results of rendering performance on TUM RGB-D.

Table 10: Rendering Performance Comparison of Ablation Study on TUM RGB-D

Method	Metric	fr1/desk	fr2/xyz	fr3/office	Avg
Ours (no as)	PSNR	25.07	22.95	26.48	24.83
	SSIM	0.937	0.909	0.941	0.929
	LPIPS	0.147	0.210	0.126	0.161
Ours (no edw)	PSNR	25.11	22.85	26.76	24.90
	SSIM	0.937	0.907	0.946	0.930
	LPIPS	0.139	0.196	0.107	0.147
Ours (no pl)	PSNR	25.39	23.71	26.92	25.34
	SSIM	0.940	0.927	0.942	0.936
	LPIPS	0.136	0.175	0.107	0.139
Ours	PSNR	25.60	23.85	26.97	25.47
	SSIM	0.942	0.929	0.946	0.939
	LPIPS	0.133	0.170	0.105	0.136

The PSNR is 23.85 dB, the SSIM is 0.929, and the LPIPS is 0.170 for the fr2/xyz scene, which is higher than all other decomposition variants. Adaptive Sampling increases the accuracy of sampling by focusing more sampling points on edges and textured areas, and exponentially decaying weights improve reconstruction quality and tracking performance by giving more weight to recent observations. The phenomenon of perceptual loss has been shown to enhance performance further through optimisation of semantic consistency and texture realism. Together, the modules will perform reconstruction and visualisation at a high level of quality.

Table 11 shows the ablation results of the rendering performance on ScanNet.

Table 11: Rendering Performance Comparison of Ablation Study on ScanNet

Method	Metric	0000	0106	0207	Avg
Ours (no as)	PSNR	28.52	26.46	28.81	27.93
	SSIM	0.922	0.921	0.910	0.917
	LPIPS	0.274	0.212	0.294	0.260
Ours (no edw)	PSNR	28.85	26.69	28.74	28.09
	SSIM	0.924	0.910	0.911	0.915
	LPIPS	0.254	0.197	0.265	0.238
Ours (no pl)	PSNR	29.02	26.86	28.90	28.26
	SSIM	0.926	0.920	0.911	0.919
	LPIPS	0.251	0.200	0.266	0.239
Ours	PSNR	29.04	29.30	29.20	29.18
	SSIM	0.927	0.923	0.912	0.920
	LPIPS	0.250	0.195	0.264	0.236

In the 0106 scene, the proposed method achieves a PSNR of 29.30 dB, an SSIM of 0.923 and an LPIPS of 0.195, and is thus better than all other deconvolution methods. Adaptive sampling can refine the details more precisely, significantly reduce the influence of tracking optimisation, and better maintain the texture and visual quality of perceptual loss. Together, the three reasons can achieve very good reconstruction results in complex environments.

Table 12 shows the tracking accuracy after ablation on Replica.

Table 12: Tracking Accuracy Comparison of Ablation Study on Replica ATE (cm)

Method	R0	R1	R2	O0	O1	O4	Avg
Ours (no as)	0.49	0.35	0.25	0.34	0.30	0.41	0.36
Ours (no edw)	0.42	0.31	0.18	0.35	0.22	0.38	0.31
Ours (no pl)	0.35	0.30	0.23	0.41	0.23	0.38	0.31
Ours	0.40	0.27	0.14	0.29	0.18	0.32	0.26

In the R2 scene with a complex desktop structure and occlusions, the complete model achieved the best result with an average root mean square error of 0.14 cm. It can be seen from this paper that without adaptive sampling, the error is 0.25 cm, and thus it is necessary to move to a dynamically adjusting sampling point to improve the accuracy of reconstruction for a complex structure. Remove the exponential decay weights, increase the error to 0.18 cm, and thus confirm that giving more weight to recent observations improves tracking accuracy. Excluding perceptual loss, the error is 0.23 cm; thus, semantic consistency and the preservation of fine details will be affected. Therefore, the three new technologies work together to reduce the tracking error in difficult circumstances substantially.

Table 13 shows the tracking accuracy after ablation on TUM RGB-D.

Table 13: Tracking Accuracy Comparison of Ablation Study on TUM RGB-D ATE (cm)

Method	fr1/desk	fr2/xyz	fr3/office	Avg
Ours (no as)	2.87	1.99	4.66	3.17
Ours (no edw)	2.96	2.21	5.45	3.54
Ours (no pl)	2.63	1.72	6.34	3.56
Ours	2.80	1.44	4.41	2.88

In the fr2/xyz scene with regular motion and moderate complexity, our model achieved the lowest error of 1.44 cm. Without adaptive sampling, the error reaches 1.99 cm, and the distribution of sampling points at the edges and in fine structures is not optimal. Removing the exponential decay weights results in the highest error; therefore, the weight of recent observation data needs to be increased. Exclude the perceptual loss; otherwise, the error would be 1.72 cm and it would lack semantic consistency and fine details. Together, the three new ideas will help us construct the reconstruction more precisely.

5 Conclusion

A dense visual SLAM system based on Gaussian segmentation is proposed in this paper, and good reconstruction quality and tracking accuracy are achieved by adopting adaptive sampling strategies, exponential decay weights and perceptual loss functions. Add an adaptive sampling strategy to move the sampling location dynamically and improve reconstruction accuracy first. Second, an exponential decay weighting function is used to give more weight to the recent observation data and thus enhance the accuracy of reconstruction and tracking. Therefore, a perceptual loss function has been introduced to use high-level semantic information for better detail recovery and to improve the visual quality of the results. According to the above experiments, the new approach has achieved notable improvements; namely, the mean increase in PSNR is 32.38%, that of SSIM is 17.48%, that of LPIPS is 47.57%, and that of ATE (RMSE) is 29.20%; thus, it has exhibited better visual effects, higher-quality environmental rendering, and more precise camera trajectory tracking. The efficacy of the above way is supported by its superior performance over other existing methods in many public datasets. The above results

have shown that the way of operating is feasible in practice, and new ideas for the development of dense visual SLAM systems have been proposed.

Discussion on Limitations and Extensions Although this method has good rendering quality and tracking accuracy for indoor scenes, it is not perfect either. First, the system is only verified in a static, stable-light environment, and its application to large-scale outdoor or dynamic scenes still needs to be improved. Second, although the perceptual loss effectively improves the quality of the image, it has a high computational cost and is thus unsuitable for lightweight deployment. Thirdly, the current adaptive sampling strategy uses manually set thresholds and is inflexible. Future research will explore data-driven sampling strategies, introduce semantic perception mechanisms, and consider expanding the system to support multimodal input and real-time applications on mobile terminals to enhance the robustness and practicality of the system further.

Acknowledgements: The authors would like to thank the Shanghai Engineering Research Centre of Pharmaceutical Intelligent Equipment for providing equipment.

Author contributions All the authors participated in the study. TWZ: Take part in research design, conduct experiments, analyze data and write papers. HLY: HLY leads this study. Constructive discussion helped revise and improve the final edition of the thesis. XDM made some revisions to the original works.

Funding for this work was provided by the Science and Technology Department of Shanghai, China, through Grants 23010501700 and 20DZ2255900, and in part by the National Natural Science Foundation of China, Grant 81960327.

All the data from this study have been included in this paper. The source code of the algorithm in this paper will be available shortly.

Declarations

Conflict of interest The authors have no conflicts of interest.

Ethics Approval This paper does not include human or animal subjects. No Ethics Approval Needed.

Consent for Publication: There are no data or figures collected from human subjects in this paper. No permission to publish the paper is required.

Funding

This work was supported in part by the Science and Technology Department of Shanghai of China under Grant 23010501700 and 20DZ2255900, and in part by the National Natural Science Foundation of China under Grant 81960327.

References

[1] Wu W, Su C, Zhu S,et al.ADD-SLAM: Adaptive Dynamic Dense SLAM with Gaussian Splatting[J]. 2025.

[2] Yu S, Cheng C, Zhou Y,et al.RGB-Only Gaussian Splatting SLAM for Unbounded Outdoor Scenes[C]//2025.DOI:10.1109/ICRA55743.2025.11128286.

[3] Zhu S, Qin R, Wang G,et al.SemGauss-SLAM: Dense Semantic Gaussian Splatting SLAM[C]//2024.DOI:10.1109/IROS60139.2025.11246462.

[4] Pak G, Kim E.VIGS SLAM: IMU-based Large-Scale 3D Gaussian Splatting SLAM[J]. 2025.

[5] Liu T, Yuan R, Ju Y,et al.GS-EVT: Cross-Modal Event Camera Tracking based on Gaussian Splatting[C]//2024.DOI:10.1109/ICRA55743.2025.11128190.

[6] Zhu Z, Fang Y, Li X,et al.Robust Gaussian Splatting SLAM by Leveraging Loop Closure[J]. 2024.

[7] Deng T, Wu W, He J,et al.VPGS-SLAM: Voxel-based Progressive 3D Gaussian SLAM in Large-Scale Scenes[J]. 2025.

[8] Wen L, Li S, Zhang Y,et al.Gassidy: Gaussian Splatting SLAM in Dynamic Environments[C]//2024.DOI:10.1109/ICRA55743.2025.11127678.

[9] Chen Z, Lu F, Yu G,et al.GSGTrack: Gaussian Splatting-Guided Object Pose Tracking from RGB Videos[J]. 2024.

[10] Zheng C, Xue L, Zarate J,et al.GauSTAR: Gaussian Surface Tracking and Reconstruction[C]//2025.DOI:10.1109/CVPR52734.2025.01542.

[11] Huang J, Li M, Sun L,et al.NGM-SLAM: Gaussian Splatting SLAM with Radiance Field Submap[J]. 2024.

[12] Matsuki H, Murai R, Kelly P H J,et al.Gaussian Splatting SLAM[J].IEEE, 2023.DOI:10.1109/CVPR52733.2024.01708.

[13] Tan Z, Chen X, Feng L,et al.TVG-SLAM: Robust Gaussian Splatting SLAM with Tri-view Geometric Constraints[J]. 2025.DOI:10.1109/LRA.2025.3641103.

[14] Zhong X, Pan Y, Jin L,et al.Globally Consistent RGB-D SLAM with 2D Gaussian Splatting[J]. 2025.

[15] Tan B, Yu R, Shen Y,et al.PlanarSplatting: Accurate Planar Surface Reconstruction in 3 Minutes[C]//2024.DOI:10.1109/CVPR52734.2025.00119.

[16] Qian Y, Sun Y, Guo Y.DynamicAvatars: Accurate Dynamic Facial Avatars Reconstruction and Precise Editing with Diffusion Models[J]. 2024.