SG2LOC: SEQUENTIAL VISUAL LOCALIZATION ON 3D SCENE GRAPHS

Anonymous authors
Paper under double-blind review

ABSTRACT

Visual localization in complex indoor environments remains a critical challenge for robotics and AR applications. Sequential localization, where pose estimates are refined over time, is important for autonomous agents. However, traditional methods often require storing extensive image databases or point clouds, leading to significant overhead. This paper introduces a novel, lightweight approach to sequential visual localization using 3D scene graphs. Our method represents the environment with a compact scene graph, where nodes represent objects (with coarse meshes) and edges encode spatial relationships. For each image in the localization phase, we extract per-patch semantic features, predicting object identities. Localization is performed within a particle filter framework. Each particle, representing a camera pose, projects the coarse object meshes from the scene graph into the image, assigning object identities to patches based on visibility. The similarity of the per-patch features, in the input image, and object features from the scene graph determines the weight of a particle. Subsequent images are incorporated sequentially, refining the pose estimate. By leveraging a compact scene graph and efficient semantic matching, our method significantly reduces storage while maintaining performance on real-world datasets. The code will be public.

1 INTRODUCTION

Visual localization is a fundamental capability in robotics and augmented reality. Accurate pose estimation (orientation and position of an agent) enables autonomous navigation, scene understanding, and user-interaction tasks. Over the years, single-image localization and Simultaneous Localization and Mapping (SLAM)-based techniques have demonstrated remarkable progress (Arandjelovic et al., 2016; DeTone et al., 2018; Berton & Masone, 2025; Mur-Artal et al., 2015; Sattler et al., 2018; Murai et al., 2025), but they often demand large storage resources for image databases or 3D point clouds equipped with visual features. As environments grow both in scale and complexity, this burden becomes impractical for memory-constrained devices or applications with low bandwidth requirements.

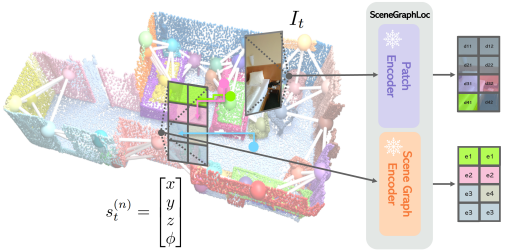

Figure 1: **Observation model** for a particle $\mathbf{s}_t^{(n)} = (x, y, z, \phi)$, matching object descriptors predicted by (Miao et al., 2024) from the query image I_t to projected object labels from the semantically segmented coarse mesh.

Sequential localization strategies mitigate these challenges by incorporating temporal cues, refining pose estimates over multiple frames rather than treating each query image independently (Barrau & Bonnabel, 2014; Leutenegger et al., 2015; Mur-Artal & Tardós, 2017; Maggio et al., 2022). Some works combine single-image pose estimation with SLAM to reduce drift and improve robustness (Lynen et al., 2020; Leutenegger et al., 2015). Nonetheless, global pose alignment typically depends on retrieving a dense 3D model or querying large-scale maps. Such approaches can be prohibitive in extended deployments, especially when frequent map updates or transfers are required.

Scene graphs present a promising alternative. Storing the environment as a graph of objects and their spatial relationships offers compactness and a semantically rich representation. Early research on scene graphs primarily targeted scene understanding, reconstruction, and retrieval (Armeni et al.,

2019; Johnson et al., 2015; Zhang et al., 2017; Lee et al., 2021; Wu et al., 2020). More recently, SceneGraphLoc (Miao et al., 2024) leveraged a scene graph to achieve cross-modal place recognition, highlighting their potential to reduce memory footprints while facilitating efficient localization.

Building on these insights, we introduce a novel method for sequential visual localization that combines a lightweight 3D scene graph with a particle filter. The proposed system relies on *coarse* object meshes, rather than dense point clouds, and *semantic* descriptors, significantly reducing storage overhead compared to the state of the art. We formulate sequential localization for indoor environments as a particle filtering problem, where each particle observes a portion of the scene and identifies object categories (*e.g.*, *table* or *chair*). By recognizing objects, the system can evaluate whether the configuration of a particle (its pose) aligns with the input sequence by simply verifying whether the same objects appear in both the particle view and the query image (see Fig. 1). The main contributions of this paper are:

- A new 3D scene-graph-based framework for sequential visual localization that jointly models semantic cues, geometric and photometric constraints.
- A particle filter approach that leverages semantic object identities to refine camera poses iteratively, without requiring large image databases or point-cloud maps.
- As a technical contribution, we adapt SceneGraphLoc to work with image sequences (Sec. 4).

2 Related Work

Visual localization is a long-standing problem in computer vision and robotics, with roots in early works on Structure-from-Motion (Kruppa, 1913; Moravec, 1980; Fischler & Bolles, 1981). Modern approaches can be categorized into single-image localization, SLAM-based methods, and sequential localization methods. Our work falls into the last category, but leverages a novel scene graph representation, distinguishing it from prior art.

Single-image localization. Many recent methods rely on a two-stage approach: coarse localization (place recognition) followed by pose estimation. Coarse localization treats the problem as image retrieval, comparing a query image to a database of geo-tagged ones. Methods like NetVLAD (Arandjelovic et al., 2016), AP-GeM (Chum & Matas, 2005) and MegaLoc (Berton & Masone, 2025) provide global image descriptors for this purpose. Fine localization typically involves establishing 2D-3D matches between image features and a 3D model (often a point cloud) of the scene, followed by pose estimation using RANSAC. Feature detectors and descriptors like SuperPoint (DeTone et al., 2018), R2D2 (Revaud et al., 2019), and D2-Net (Dusmanu et al., 2019) are commonly used. While effective, these methods require storing large image databases or 3D point clouds equipped with visual features, leading to substantial storage demands. Alternative approaches, such as scene coordinate (Brachmann et al., 2017; Brachmann & Rother, 2018; Sattler et al., 2011; Li et al., 2012; Sattler et al., 2016; 2017; Brachmann et al., 2023; Wang et al., 2024) and absolute pose regression (Walch et al., 2017; Brachmann & Rother, 2018; Sattler et al., 2016; Kendall et al., 2015), directly predict 3D coordinates or camera poses from the image. However, these often struggle with complex, large-scale environments (Sattler et al., 2018; 2016; 2017; Wang et al., 2024). Our work is fundamentally different, as it avoids storing image databases or point clouds by using a compact scene graph.

Simultaneous localization and mapping (SLAM) systems (Barrau & Bonnabel, 2014; Engel et al., 2014; Gao et al., 2018; Mur-Artal et al., 2015; Mur-Artal & Tardós, 2017; Murai et al., 2025) provide local tracking of the trajectory of a camera. Popular examples include PTAM (Klein & Murray, 2007), LSD-SLAM (Engel et al., 2014), ORB-SLAM (Mur-Artal et al., 2015; Mur-Artal & Tardós, 2017), and LDSO (Gao et al., 2018). While these systems excel at tracking relative motion, they are prone to drift over time and typically do not perform global localization (loop closure) without additional mechanisms. Some systems integrate inertial measurements (Lynen et al., 2020; Qin et al., 2018) for improved robustness. Our work builds upon SLAM systems, leveraging their estimated camera trajectory while providing lightweight global localization capabilities.

Sequential localization. Several approaches combine single-image localization with SLAM for improved robustness (Barrau & Bonnabel, 2014; Forster et al., 2014; Bloesch et al., 2015; Sattler et al., 2016; Leutenegger et al., 2015; Qin et al., 2018; Mur-Artal & Tardós, 2017), often using visual-inertial SLAM for local tracking and periodically querying a server or compressed map for global localization (Lynen et al., 2020; Bloesch et al., 2015; Leutenegger et al., 2015). Maplab (Schnei-

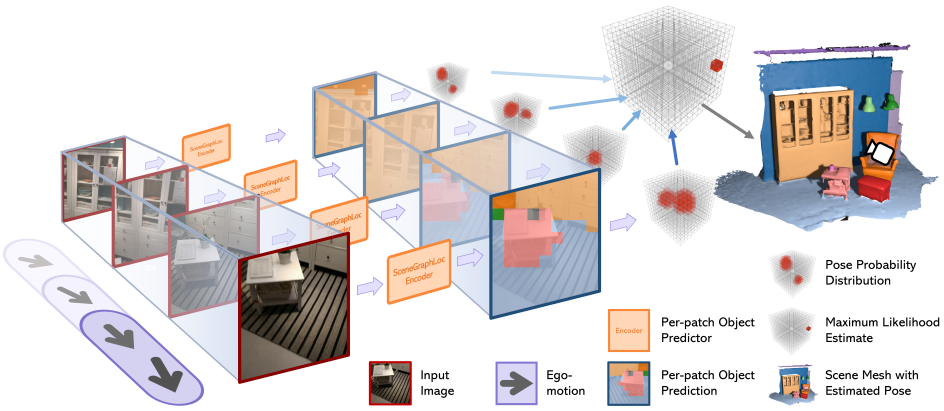

Figure 2: **Sequential localization pipeline.** Given the current image I_t and its ego-motion, the pipeline updates the particle state while leveraging previously processed images $I_0, \ldots, I_{t-1}$. The current image is passed through the SceneGraphLoc (Miao et al., 2024) encoder to predict object labels for each image patch. These predictions inform a 4D probability distribution over the camera pose (3D position and rotation around the vertical axis), which is then integrated into the posterior distribution, refining the estimated camera pose as new images become available.

der et al., 2018) provides a framework for benchmarking, while KFNet (Zhou et al., 2020) offers a learning-based alternative. Other methods incorporate temporal information by modeling image sequences, enabling joint localization across multiple frames. More recently, MASt3R-SLAM (Murai et al., 2025) and VGGT-Slam (Maggio et al., 2025) use feed-forward 3D geometry for feature matching and mapping. In contrast, NeRF- and Gaussian Splat-based methods (Maggio et al., 2022; Adamkiewicz et al., 2022; Khatib et al., 2025; Meng et al., 2025) assume a pre-built dense reconstruction and localize by minimizing photometric error of rendered and observed views. While offering high-fidelity appearance matching, they are memory- and compute-intensive and remain sensitive to scene changes. We instead propose lightweight scene graphs with semantic and depth cues, providing compact maps and constraints in addition to photometric consistency.

3D scene graphs for localization and retrieval have emerged as a powerful representation for capturing scene understanding, extending geometric representations with semantic and relational knowledge. Early work on scene graphs focused on scene understanding and reconstruction from point clouds or RGB-D data (Armeni et al., 2019; Lee et al., 2021). More recent work has explored their use in various tasks, including image retrieval (Johnson et al., 2015; Wu et al., 2020), visual question answering (Zhang et al., 2017), and navigation (Zhou et al., 2021). The recent SceneGraphLoc (Miao et al., 2024) introduced the novel problem of cross-modal localization of a query image within a database of 3D scene graphs, demonstrating significant storage savings and faster query times compared to image-based methods. While not direct localization, it showed promise for place recognition. Our approach builds upon the strengths of sequential localization methods and the light-weight nature of scene graphs to provide a new localization direction.

3 Sequential Localization with Particle Filter

We aim to estimate a 4 degree-of-freedom camera pose $\mathbf{s}_t = (x, y, z, \phi)$, where ϕ denotes rotation around the known gravity axis, from a sequence of images $\{I_1, \ldots, I_T\}$. We maintain a set of N particles $\mathbf{S}_t = \{\mathbf{s}_t^{(1)}, \ldots, \mathbf{s}_t^{(N)}\}$ that evolves over time using a particle filter. The filter integrates a *coarse*, labeled 3D mesh (derived from a 3D scene graph) with the pre-trained SceneGraphLoc model (Miao et al., 2024) to compute observation likelihoods. The pipeline is visualized in Fig. 2.

Note that the gravity direction and camera intrinsics are usually accessible from robot sensors, smartphones, and head-mounted devices. If this information is not readily available, methods such as GeoCalib (Veicht et al., 2024) can be used to estimate both the intrinsics and gravity direction.

Scene representation. We model the environment as a 3D scene graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each vertex $v_i \in \mathcal{V}$ corresponds to an object instance o_i. Each object node o_i is associated with:

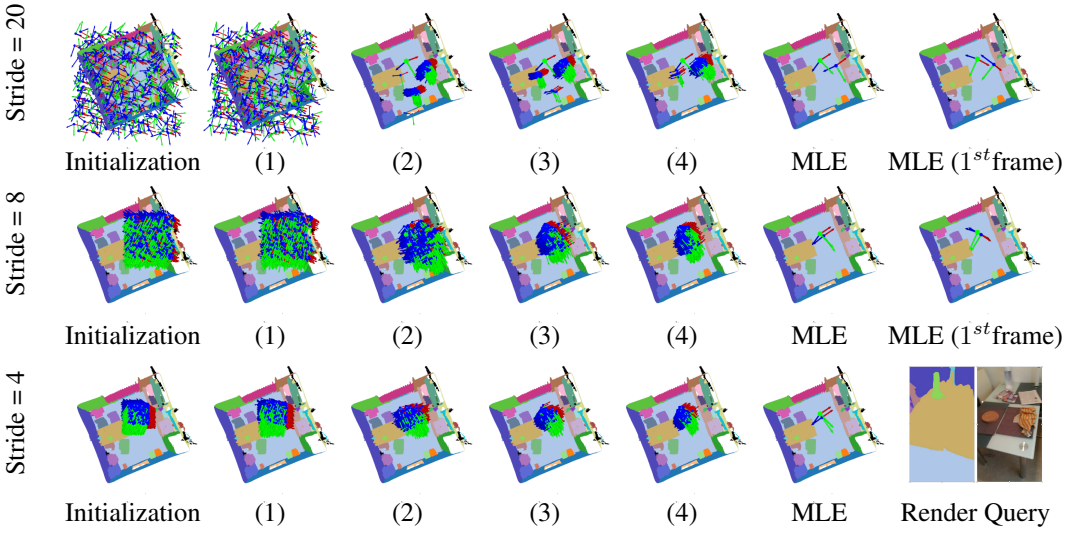

Figure 3: **Multi-round particle filter** on a 5-image sequence, first running with a stride of 20, 8, and finally 4 (the stride controls the rate of downsampling the images). The first column shows the initial random particle distribution, gradually narrowing the search space in subsequent rounds. Each next column represents particle updates after integrating the n^{th} image (indicated below each plot). Then, the Maximum Likelihood Estimate (MLE) is shown in blue, and the GT pose in green, back-propagated to the 1^{st} frame for the next optimization round.

- A compact embedding $\mathbf{e}_i \in \mathbb{R}^d$, obtained from the SceneGraphLoc object encoder (Miao et al., 2024), which combines multi-modal inputs (*e.g.*, RGB images, point clouds, textual annotations, relationships to other objects).
- A coarse 3D mesh $\mathcal{M}_i$ that captures the approximate geometry of the object for raycasting. The mesh needs only be detailed enough for visibility and occlusion checks.[1]

Edges in $\mathcal{E}$ encode spatial relationships (*e.g.*, adjacency) among objects, though our localization pipeline primarily relies on object embeddings $\{\mathbf{e}_i\}$ and coarse meshes $\{\mathcal{M}_i\}$. By storing these low-dimensional descriptors and approximate geometries instead of dense image databases or large point clouds, we greatly reduce the memory footprint while still enabling accurate pose estimation.

Particle filter. To approximate posterior $p(\mathbf{s}_t \mid I_{1:t})$, we maintain a set of weighted particles as $p(\mathbf{s}_t \mid I_{1:t}) \approx \sum_{n=1}^{N} w_t^{(n)}\, \delta(\mathbf{s}_t - \mathbf{s}_t^{(n)})$, where δ is the Dirac delta function and $w_t^{(n)}$ denotes the normalized weight of the n-th particle at time t. At each timestep, the filter performs (i) a prediction step (motion model), (ii) an update step (observation model), and (iii) optional resampling, as described below in the next section.

3.1 Initialization

Patch embeddings from SceneGraphLoc. We subdivide each incoming image I_t into a grid of $n \times m$ rectangular patches $\{p_{r,c}\}$, where $r \in [0, n)$ and $c \in [0, m)$. We follow SceneGraphLoc (Miao et al., 2024) and use a 14×14 grid, yielding 196 patches. We apply the SceneGraphLoc image encoder to obtain a set of patch embeddings $\{\hat{\mathbf{e}}_{r,c}\}$. Each $\hat{\mathbf{e}}_{r,c}$ is a semantic descriptor indicating which object (from the scene graph vocabulary) is most likely visible in the patch.

Particle distribution. We then uniformly distribute particles in a 3D bounding region Ω spanning four approximate heights $\{1.50, 1.60, 1.70, 1.80\}$ meters above the floor. Concretely, we partition Ω into grid cells of size 0.2 meters, sample three poses per cell, and randomly assign yaw angles $\phi \in [-\pi, \pi]$. All particles receive equal initial weight, determined as follows $p(\mathbf{s}_0) = \frac{1}{|\Omega|}$ if $\mathbf{s}_0 \in \Omega$ else 0. This broad initialization ensures coverage of plausible poses before the sequential estimation begins.

[1] On average, the 3RScan (Wald et al., 2019) meshes have 863 vertices per object and ScanNet (Dai et al., 2017) 11681 vertices per object in the experiments.

3.2 PREDICTION (MOTION MODEL)

To propagate each particle $\mathbf{s}_t^{(n)}$ from timestep t to $t+1$, we apply the camera's ego-motion $\mathbf{t}_t = [\,t_{x,t},\, t_{y,t},\, t_{z,t},\, t_{\phi,t}\,]$, augmented by i.i.d. Gaussian noise $\boldsymbol{\omega}_t = [\,\omega_{x,t},\, \omega_{y,t},\, \omega_{z,t},\, \omega_{\phi,t}\,]$:

$$\mathbf{s}_{t+1}^{(n)} \;=\; \mathbf{s}_t^{(n)} \oplus \big(\mathbf{t}_t + \boldsymbol{\omega}_t\big). \tag{1}$$

Here, $\oplus$ applies a 4-DoF transformation (3D trans./rot. around the gravity axis). We set $\sigma_{\text{trans}} = 0.05$ m and $\sigma_{\text{rot}} = 0.05$ rad as the standard deviations for translation and yaw noise, respectively. This accounts for modeling uncertainty in the motion estimates while maintaining gravity alignment. In practice, ego-motion is obtained by the employed SLAM system, such as (Teed & Deng, 2021).

3.3 UPDATE (OBSERVATION MODEL)

After predicting the particle set $\mathbf{S}_{t+1}$, we incorporate the new image I_{t+1} to refine particle weights.

Raycasting. For each particle pose $\mathbf{s}_{t+1}^{(n)}$, we project the coarse 3D meshes $\{\mathcal{M}_i\}$ from the scene graph into the image. This identifies which objects (nodes in $\mathcal{G}$) are visible in each of the 14×14 patches, assigning an object embedding $\mathbf{e}_i$ to each patch $\hat{p}_{r,c}$ for that particle.

Patch similarity. We then compare the patch embedding $\hat{\mathbf{e}}_{r,c}$ from the query image I_{t+1} (Section 3.1) with the object embedding $\mathbf{e}_i$ assigned by raycasting. We compute the cosine similarity and retain it if the predicted object matches the one determined by raycasting. Summing these valid similarities across all 144 patches and normalizing by the total patch count yields a score $s \in [0, 2]$, where the cosine similarity values in $[-1, 1]$ are shifted to the positive range.

Particle weighting. We model the likelihood using a Gaussian centered on $s = 2$ as $L\big(\mathbf{s}_{t+1}^{(n)}\big) = \exp(-(2-s)^2/(2\,\sigma^2))$, where $\sigma = 0.2$. The unnormalized weight of each particle becomes $\tilde{w}_{t+1}^{(n)} = w_t^{(n)} \cdot L(\mathbf{s}_{t+1}^{(n)})$, and we normalize by $\sum_m \tilde{w}_{t+1}^{(m)}$ to obtain $w_{t+1}^{(n)}$. Particles with higher similarity scores receive greater weight, driving the distribution toward accurate poses in subsequent steps.

3.4 ADDITIONAL SUPERVISION SIGNALS

In addition to semantics, we use photometric and geometric cues to guide the particle filter.

Color supervision. As we are given an RGB sequence as input, we can leverage photometric losses to further improve our particle state quality measurements. Let us assume that our map representation is a coarse textured mesh. For each particle $\mathbf{s}_{t+1}^{(n)}$, we project the meshes $\{\mathcal{M}_i\}$ of the visible objects into the image plane. Note that this step does not require additional computations as the mesh has already been projected for semantic supervision. Now, instead of assigning object identities, we render an RGB image $I_p^{(n)}$ for the particle. Given the next image I_{t+1} in the input sequence, we calculate the photometric score $s_i^{(n)}$ of the particle by calculating the structural similarity (SSIM) as $s_i^{(n)} = L_{\text{SSIM}}(I_{t+1}, I_p^{(n)})$, where L_{SSIM} returns values between 0 and 1.

Depth supervision. When depth information is available, either from an RGB-D sensor or a depth estimator applied to the image sequence, we can further incorporate this information to enhance accuracy. We achieve this by adding a depth-based score.

Depth map projection. For each particle $\mathbf{s}_{t+1}^{(n)}$, we project the coarse 3D meshes of the visible objects into the image plane. Now, instead of assigning object identities, we render a depth map $D_p^{(n)}$ for the particle. This depth represents the distance from the camera plane to the projected mesh surfaces, according to the pose of the particle. We use the same resolution for $D_p^{(n)}$ as the input image (or the downsampled version, as described in Sec. 3.5). Let D_I be the depth map corresponding to the input image I_{t+1}. If I_{t+1} is an RGB-D image, D_I is directly obtained from the sensor. If only RGB data is available, D_I can be estimated using a monocular or stereo depth estimation method.

Depth score calculation. We compute the depth score $s_d^{(n)}$ for particle $\mathbf{s}_{t+1}^{(n)}$ by calculating the L1 difference of the projected depth map $D_p^{(n)}$ and the input D_I. The score is calculated as follows: $s_d^{(n)} = 1 - \frac{1}{R \cdot C} \sum_{r=1}^{R} \sum_{c=1}^{C} |D_p^{(n)}(r,c) - D_I(r,c)|$, where R and C are the number of rows and columns of the depth maps, respectively, and $D_p^{(n)}(r,c)$ and $D_I(r,c)$ are the depth values at pixel (r,c) in the projected and input depth maps, respectively. The score is designed such that 1 is best and lower values are worse.

Table 1: **Localization accuracy** on the 3RScan and ScanNet datasets. We report the average and median position error (m) and average and median rotation error (°) for sequences of length 5, 10, 25, and 50 frames. ACE is shown faded on ScanNet since its encoder was trained on this dataset. Lower values indicate better localization performance.

	Method	5 frames				10 frames				25 frames				50 frames			
		Mean		Median		Mean		Median		Mean		Median		Mean		Median	
		Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)
3RScan	HLoc	2.05×10^{12}	46.68	**0.18**	6.32	5.10×10^{10}	36.85	**0.13**	4.15	1030.90	**22.74**	**0.08**	2.44	732.20	16.09	0.07	2.06
3RScan	MeshLoc	4.51×10^{5}	22.81	0.47	**2.65**	2042.24	**15.72**	0.44	**2.29**	26.15	**9.46**	0.42	**1.74**	15.29	8.99	0.40	1.70
3RScan	Loc-NeRF	**1.34**	38.42	1.19	21.63	**1.35**	40.01	1.02	22.00	**1.24**	31.47	0.79	14.08	1.12	23.09	0.59	7.46
3RScan	**SG2Loc (Ours)**	2.43	**21.24**	0.20	3.27	2.30	20.50	0.18	2.96	6.15	22.93	0.14	2.73	9.48	27.43	0.10	2.62
3RScan	ACE	2.99	61.14	1.66	42.38	**2.45**	52.52	0.93	27.04	1.69	40.05	0.39	10.31	1.15	30.10	0.19	5.33
3RScan	ACE + GS-CPR	3.33	65.82	1.61	50.13	2.93	56.05	0.75	20.88	2.40	42.71	0.18	5.93	2.06	29.78	0.14	4.37
ScanNet	HLoc	6.16×10^{4}	14.30	**0.09**	2.63	603.05	10.48	**0.08**	2.45	492.00	7.27	**0.06**	2.12	1348.28	7.12	**0.07**	1.94
ScanNet	MeshLoc	7.62	**4.14**	0.31	**1.90**	8.54	**3.57**	0.30	**1.90**	**0.30**	**2.28**	0.29	**1.84**	**0.29**	**2.06**	0.28	1.77
ScanNet	Loc-NeRF	1.32	33.33	1.09	17.60	1.45	29.59	1.11	15.10	1.36	20.58	0.77	9.02	1.18	12.02	0.46	8.15
ScanNet	**SG2Loc (Ours)**	**0.53**	11.55	0.12	2.55	**0.44**	8.14	0.10	2.30	0.32	4.55	0.09	2.29	2.10	12.27	0.08	**1.70**
ScanNet	ACE	0.23	4.43	0.08	1.99	0.18	3.85	0.08	1.93	0.13	2.77	0.07	1.70	0.10	2.78	0.06	1.81

Combined score. The final weight is then computed by combining the semantic similarity score s (from Sec. 3.3), the depth $s_d^{(n)}$ and color scores $s_i^{(n)}$ as:

$$L\big(\mathbf{s}_{t+1}^{(n)}\big) = \exp\Big(-\frac{(6-(s+\lambda_1 s_d^{(n)}+\lambda_2 s_i^{(n)}))^2}{2\,\sigma^2}\Big), \tag{2}$$

where λ_1 and λ_2 are weighting factors that balance the contribution of the depth and color scores. The values of λ_1, λ_2 are set empirically. For our experiments, all losses are weighted equally, the weights are fixed and all parameters are kept fixed across all experiments. The unnormalized weight becomes $\tilde{w}_{t+1}^{(n)} = w_t^{(n)} \cdot L\big(\mathbf{s}_{t+1}^{(n)}\big)$. We normalize these weights as before: $w_{t+1}^{(n)} = \tilde{w}_{t+1}^{(n)} / \sum_m \tilde{w}_{t+1}^{(m)}$. This combined scoring mechanism leverages semantic, geometric and appearance information, leading to more robust and accurate localization, especially in cases where the semantic information alone might be ambiguous. The depth and color scores provide additional constraints based on geometric and photometric consistency, helping to disambiguate poses that might have similar semantic projections.

3.5 Coarse-to-Fine Optimization

We refine the pose estimate over multiple rounds, gradually increasing raycasting resolution and narrowing the search region. We observed that this process speeds up the localization and leads to higher accuracy. In the first round, we downsample the input image with a stride of 20 pixels, enabling a coarse but efficient search over the entire scene. Particles are initialized uniformly within the full environment bounds, allowing a rapid, coarse localization. Next, we apply an 8-pixel stride, restricting the particle initialization to a 1.6^3 m region centered around the maximum likelihood estimate (MLE) pose from the previous round. This second pass refines localization by focusing on a smaller region. Finally, we downsample the image with stride 4 and use an even tighter bounding box of 0.8^3 m around the new MLE pose. This final pass incorporates high-resolution visibility checks and yields the most precise pose estimate.

Adaptive resampling. Following each update, we dynamically adjust the particle count using an adaptive scheme based on KLD-sampling (Fox, 2001). Letting ϵ and δ be bounds on the Kullback–Leibler divergence and its confidence level, respectively, we compute $n = \frac{1}{2\epsilon}\left(1 - \frac{2}{9(k-1)} + \sqrt{\frac{2}{9(k-1)}}\, z_{1-\delta}\right)^3$, where k is the bin number in the state histogram used for divergence estimation, and $z_{1-\delta}$ is the $(1-\delta)$-quantile of the normal distribution. We then apply stratified resampling to draw n new particles, ensuring that the MLE distribution remains an accurate approximation of the true posterior. An example visualization is shown in Fig. 3.

3.6 Pose Refinement with PnP

To refine the final pose from the particle filter, we render six synthetic views from the mesh. One from the MLE pose estimated in the previous steps and five sampled within a range of $\pm 45°$ yaw around the MLE. We match each view with the query image using RoMa (Edstedt et al., 2024) and establish 2D-3D correspondences through a ray-mesh intersection. Pose refinement is performed using RANSAC-based PnP from PoseLib (Larsson & contributors, 2020). For each sequence, we apply this process per frame by backpropagating the MLE pose (obtained from the last frame) to earlier images. We select the pose with the highest inlier count as the final estimate.

Table 2: **Pose recalls** on the 3RScan and ScanNet datasets. We report position, rotation and joint recalls at thresholds: Pos. R@0.25m / Rot. R@2° / R@0.25m, 2° for sequences of length 5, 10, 25 and 50. Recall measures the fraction of errors that fall below those thresholds. ACE is shown faded on ScanNet since its encoder was trained on this dataset. Higher values indicate better performance.

	Method	5 frames	10 frames	25 frames	50 frames
3RScan	HLoc	**0.54** / 0.28 / 0.27	**0.62** / 0.32 / 0.32	**0.74** / 0.43 / **0.43**	0.76 / 0.50 / **0.50**
	MeshLoc	0.04 / **0.41** / 0.03	0.05 / **0.45** / 0.03	0.05 / **0.54** / 0.03	0.06 / **0.55** / 0.04
	Loc-NeRF	0.03 / 0.05 / 0.00	0.06 / 0.04 / 0.01	0.11 / 0.13 / 0.04	0.16 / 0.14 / 0.02
	SG2Loc (Ours)	0.53 / 0.38 / **0.32**	0.55 / 0.38 / **0.33**	0.59 / 0.43 / 0.39	0.65 / 0.43 / 0.43
	ACE	0.26 / 0.11 / 0.11	0.33 / 0.14 / 0.14	0.45 / 0.21 / 0.21	0.55 / 0.29 / 0.29
	ACE + GS-CPR	0.33 / 0.14 / 0.14	0.42 / 0.19 / 0.19	0.54 / 0.27 / 0.27	0.64 / 0.36 / 0.36
ScanNet	HLoc	**0.81** / 0.38 / **0.38**	**0.86** / 0.41 / **0.41**	**0.91** / 0.47 / **0.47**	**0.98** / 0.52 / 0.52
	MeshLoc	0.25 / **0.53** / 0.13	0.25 / **0.53** / 0.13	0.29 / **0.56** / 0.15	0.29 / **0.65** / 0.13
	Loc-NeRF	0.03 / 0.08 / 0.00	0.05 / 0.08 / 0.01	0.21 / 0.14 / 0.08	0.31 / 0.14 / 0.03
	SG2Loc (Ours)	0.73 / 0.40 / 0.36	0.80 / 0.40 / 0.39	0.86 / 0.43 / 0.42	0.90 / 0.61 / **0.61**
	ACE	0.91 / 0.50 / 0.50	0.94 / 0.52 / 0.51	0.94 / 0.60 / 0.59	0.97 / 0.53 / 0.53

4 Sequential Scene Retrieval

SceneGraphLoc (Miao et al., 2024) was originally designed to select the correct scene (represented as a 3D scene graph) from a set of candidate scene graphs $\{\mathcal{G}_i\}$ given a query image as input. In this section, we adapt SceneGraphLoc to leverage not just one image but the entire input sequence when retrieving the current scene from the database.

For each image I_t in the sequence, we extract a set of image patches $\mathcal{Q}_t$. For each patch $q \in \mathcal{Q}_t$, we compute its embedding e_q using the pre-trained encoder. Similar to (Miao et al., 2024), we then compare e_q with the embeddings e_v of nodes $v \in \mathcal{V}_i$ in each scene graph $\mathcal{G}_i$ in our database using a similarity metric as $\text{similarity}(q, v) = \cos(e_q, e_v)$. We assign a score to each scene graph $\mathcal{G}_i$ based on the similarity scores of its nodes to the image patches as: $\text{score}_t(\mathcal{G}_i, I_t) = \frac{1}{|\mathcal{Q}_t|} \sum_{q \in \mathcal{Q}_t} \max_{v \in \mathcal{V}_i} \text{similarity}(q, v)$. To incorporate information from the sequence, we aggregate the scores from all images as $\text{score}(\mathcal{G}_i) = \sum_{t=1}^{T} \text{score}_t(\mathcal{G}_i, I_t)$. The scene graph with the highest final score is selected as the correct match for the sequence. This process allows us to better find the correct scene in a database of maps.

5 Experiments

Datasets. The *3RScan* dataset (Wald et al., 2019) contains 1,335 annotated indoor scenes across 432 spaces, with 1,178 scenes (385 rooms) for training and 157 (47 rooms) for validation. Each scene is represented by a semantically annotated 3D point cloud, with multiple captures over months to reflect environmental changes. Since the test set lacks scene graph annotations, we follow (Miao et al., 2024) and reorganize the validation set into 34 scenes (17 rooms) for validation and 123 scenes (30 rooms) for testing. To ensure realistic evaluation, query sequences are localized against maps from different temporal states, yielding 9,445 images. We note that 3RScan is explicitly designed to evaluate localization under significant scene changes, including object rearrangements, removal, and occlusions. SG2Loc is evaluated on every frame of this dataset in a cross-temporal setting, where query sequences are matched against maps captured some time apart. For evaluation without GT scene graphs, we also use *ScanNet* (Dai et al., 2017). Following (Miao et al., 2024), we generate scene graphs with SceneGraphFusion (Wu et al., 2021) and adopt the same split of 57 scan pairs (captured at different times), sampling one image every 25 frames for 4,088 queries. This split will be released publicly.

Baselines. We compare with HLoc (Sarlin et al., 2019), MeshLoc (Panek et al., 2022), Loc-NeRF (Maggio et al., 2022), ACE (Brachmann et al., 2023) and ACE poses refined with GS-CPR (Liu et al., 2024). HLoc is a hierarchical method combining image retrieval (50 nearest neighbors per query) with pose estimation from 2D–3D correspondences, requiring a large image database. MeshLoc localizes via a 3D mesh, storing a depth map per image generated from the mesh. We use SuperPoint (DeTone et al., 2018) and SuperGlue (Sarlin et al., 2020) for local feature matching. ACE is a scene coordinate regression network trained per scene enabling accurate localization. GS-CPR refines ACE by using rendered depth from Gaussian Splats and MASt3R correspondences to refine camera poses. For sequential evaluation, we run HLoc, MeshLoc, ACE, and ACE+GS-CPR per image and select the pose with the most RANSAC inliers as the sequence result. Loc-NeRF follows a similar filtering approach as we do, using only photometric error. We replaced the original NeRF map with a Gaussian Splat for faster localization. HLoc and MeshLoc both require substantial storage, limiting scalability.

Metrics. We evaluate the performance of our method and baselines using standard metrics for visual localization: average and median position (in meters), rotation errors (in degrees) and recall. Additionally, we report the storage requirements of each method, including the size of the database and any additional structures. The rotation error is calculated as the geodesic distance between the estimated rotation matrix $\hat{R}$ and the ground truth rotation matrix R_{gt} on the SO(3) manifold. It is calculated as follows: $\epsilon_R = \arccos((1/2)(\text{tr}(R_{gt}^T \hat{R}) - 1))$.

Sequential visual localization. Table 1 reports average and median position (m) and rotation (°) errors for sequences of length 5, 10, 25, and 50. SG2Loc achieves lower mean errors than HLoc, with comparable median position errors and substantially better rotation accuracy. While methods such as MeshLoc yield strong rotation estimates, their position errors remain well above those of HLoc and SG2Loc. ACE performs worse than HLoc and other baselines on 3RScan, but achieves high accuracy on ScanNet, likely due to its encoder being trained on ScanNet, unlike the other methods. The GS-CPR refinement of ACE poses (ACE+GS-CPR) improves ACE, but our method still achieves higher accuracy. All methods benefit from longer sequences, which consistently improve performance. Table 2 shows that our method achieves the best combined recalls for 5/10 frames and the second-best for 25/50 on 3RScan. On ScanNet it achieves best combined recall for length 50.

HLoc and MeshLoc rely on significantly more information by leveraging large-scale databases, yet SG2Loc performs on par with them, sometimes slightly worse, and sometimes better. Crucially, our approach achieves comparable accuracy, while requiring *one* order of magnitude less storage than HLoc and *two* less than MeshLoc on 3RScan, and *one* order less than both on ScanNet. Storage in MB is shown in Table 3. This significant reduction in storage makes SG2Loc well-suited for on-device localization, where efficient map storage and low-bandwidth transmission are critical constraints. Our method uses roughly 2500 particles on 3RScan and 3200 particles on ScanNet for the results reported in Table1 and Table 2. Raycasting for all particles is run fully in parallel on GPU, which keeps runtime manageable even with larger particle sets.

Table 3: **Average storage** per scene in MB.

Method	3RScan	ScanNet
HLoc	294.4	283.3
MeshLoc	1433.6	589.7
Loc-NeRF	143.9	66.6
SG2Loc (Ours)	<u>9.8</u>	<u>28.2</u>
ACE	**4.2**	**4.2**

The mapping and per-frame processing times (in seconds) are reported in Table 4. For the proposed method, the mapping time includes constructing a kd-tree for raytracing with particles and computing object embeddings for the scene graph. For HLoc, it includes extracting image embeddings and 2D-3D correspondences for the database, while for MeshLoc, it involves computing both image embeddings and depth maps. Loc-NeRF and ACE require training a separate map representation for each scene. The localization for SG2Loc consists of state transitions, particle updates (embeddings are computed per query image and raycasting is performed from all particles), and resampling steps, before post-processing with PnP. The localization time represents the average time required to estimate the camera pose for a single frame. The proposed method requires substantially less computation during mapping than the baselines. At inference, it runs at an average of 2.9 seconds per frame, followed by pose optimization, which is suitable for sequential localization.

Keyframing. In practice, localization need not be performed on every frame, as high frame rates introduce redundancy. To simulate a realistic scenario, we integrate frames at a rate matched to our runtime (Table 6), where the agent moves in real time and incoming frames must be processed sequentially. Our method significantly outperforms HLoc on the same inputs, demonstrating its practicality for robot localization.

Table 4: **Average runtime on 3RScan** per scene (offline, seconds) for frame integration and final optimization by PnP (runs once). The localization time is averaged over sequences and accounts for the *per-frame* integration.

Method	Mapping	Localization	Final Opt.
HLoc	3520.0	**0.01**	–
MeshLoc	2208.0	<u>0.07</u>	–
Loc-NeRF	1490.0	7.50	–
SG2Loc (Ours)	**1.6**	2.90	5.5
ACE	<u>134.2</u>	0.08	–

Sequential scene retrieval. In this section, we evaluate the extension proposed in Section 4, which enables SceneGraphLoc (Miao et al., 2024) to operate in a sequential setting. We compare this extension to the original method, which performs localization using only the first image of the sequence. Additionally, we benchmark against state-of-the-art image retrieval methods, AnyLoc (Keetha et al., 2023) and CVNet (Lee et al., 2022).

Table 5: **Cross-modal sequential scene retrieval** on the 3RScan dataset (Wald et al., 2019), where the goal is to identify the correct scene from a set of 10 or 50 candidates. We evaluate SceneGraphLoc (Miao et al., 2024) operating on single frames and our proposed extension, which enables (Miao et al., 2024) to process sequences. We also show results of standard techniques (copied from (Miao et al., 2024)) CVNet (Lee et al., 2022) and AnyLoc (Keetha et al., 2023). Tests are conducted on sequences of length 5, 10, 25, and 50. We report scene retrieval recall in the temporal setting (R^t) at ranks 1, 3, and 5, indicating whether the correct scene appears among the top-k predictions. Additionally, we present inference time (in milliseconds) and storage requirements for map representation, demonstrating the efficiency of our approach.

	10 scenes				50 scenes				Storage
# of frames	R^t@1	R^t@3	R^t@5	Time (ms)	R^t@1	R^t@3	R^t@5	Time (ms)	MB
1 ((Miao et al., 2024))	0.82	0.94	**0.98**	**0.3**	0.69	0.79	0.84	**1.5**	**5.4**
5	0.84	0.95	**0.98**	1.5	0.72	0.84	0.88	7.5	**5.4**
10	0.86	0.95	**0.98**	3.0	0.75	0.85	0.89	15.0	**5.4**
25	0.88	**0.96**	**0.98**	7.5	0.78	0.87	0.91	37.5	**5.4**
50	**0.89**	**0.96**	**0.98**	15.0	**0.81**	**0.89**	**0.92**	75.0	**5.4**
CVNet (Lee et al., 2022)	0.79	0.91	0.95	60.0	0.67	0.77	0.82	311.1	239.1
AnyLoc (Keetha et al., 2023)	0.88	0.95	**0.98**	1826.4	**0.81**	0.87	0.90	1451.1	5720.3

Table 6: **Keyframing on 3RScan.** We report avg. and median pos. (m) and rot. error (°), and recall at $(25\text{cm}, 2^\circ)$ for 100-frame sequences.

	HLoc	Ours w/ keyframing
Mean pos. (m) ↓	**0.94**	2.36
Mean rot. (°) ↓	33.99	**20.08**
Med. pos. (m) ↓	0.27	**0.18**
Med. rot. (°) ↓	3.67	**2.75**
Pos. R@0.25m ↑	0.48	**0.54**
Rot. R@2° ↑	0.35	**0.43**
R@25cm, 2° ↑	0.35	**0.40**

Table 7: **Ablation study on 3RScan.** We report recall at $(10\text{cm}, 5^\circ)$ and $(25\text{cm}, 10^\circ)$ for 5-frame sequences. Higher recall is better.

Method	R@10cm, 5°	R@25cm, 10°
Max. raycast resolution (3.5)	0.07	0.33
Uniform sampling (3.1)	0.10	0.30
w/o adaptive resampling (3.5)	0.11	0.30
SG2Loc w/ semantic (3.4)	0.11	0.35
SG2Loc w/ semantic+depth (3.4)	0.20	0.54
SG2Loc w/ semantic+depth+RGB (3.4)	0.35	0.61
SG2Loc (Ours) (3.6)	**0.50**	**0.65**

We follow the evaluation protocol used in (Miao et al., 2024), where, given a query image (or sequence), we retrieve the top-k scenes from a candidate set of either 10 or 50 scenes. This evaluation is conducted ensuring that the query image was captured at a time step different from the reference map. We report recall at ranks 1, 3, and 5, measuring how often each method retrieves the correct scene among its top-k predictions ($k \in \{1, 3, 5\}$). Additionally, we provide localization time in milliseconds and storage requirements for retrieval. For our method, storage corresponds to the scene graph embeddings, whereas for CVNet and AnyLoc, it reflects the database of image embeddings.

The results are presented in Table 5. As expected, using longer sequences for localization substantially and consistently improves retrieval recall in both the 10-scene and 50-scene settings while introducing only a minimal increase in runtime. With sequences of 25 and 50 frames, the proposed method matches the performance of storage- and computation-intensive image retrieval baselines. These findings demonstrate that the sequential extension of SceneGraphLoc is effective and serves as a valuable complement to the localization approach proposed in this paper.

Ablation Studies are conducted on the first 4 scenes of the 3RScan (Wald et al., 2019) dataset, comprising 335 sequences of 5 images. The results are presented in Table 7. The first 3 ablations evaluate components of our method relative to *SG2Loc w/ semantic*. The *Max. resolution* (stride = 1) setting removes downsampling in the final pass of multi-round optimization, using the highest resolution for raycasting. As shown in Table 7, this configuration achieves similar accuracy to *SG2Loc w/ semantic*, while achieving much lower computational cost. The *Uniform sampling* setting initializes particles on a uniform 3D grid with a resolution of 0.2m, assigning 3 random poses per grid cell. This approach results in significantly lower accuracy compared to *SG2Loc w/ semantic* with the proposed init. strategy. The *w/o adaptive resampling* configuration replaces the adaptive resampling with a fixed particle count. While this achieves similar med. errors, it significantly reduces recall, showing that dynamic particle count increases accuracy. The ablations *SG2Loc w/ semantic*, *w/ semantic+depth*, *w/ semantic+depth+RGB*, explore the impact of supervision signals without PnP. Each additional signal improves performance. The best variant (*SG2Loc w/ semantic+depth+RGB*) initializes the PnP refinement, and our full SG2Loc method (last row), yields the best overall results.

Limitations. Although the proposed method achieves substantial storage savings and competitive localization accuracy compared to the state-of-the-art HLoc and MeshLoc, it incurs higher computational cost during per-frame integration. We believe that this overhead can be significantly reduced

through further code optimizations. Moreover, the keyframing experiment highlights that SG2Loc can already be used in time-sensitive applications by processing only keyframes.

6 Conclusion

We present a lightweight approach to sequential visual localization using 3D scene graphs and a particle filter, avoiding the need for large image databases or dense point clouds. By leveraging semantic object descriptors and coarse meshes, our method efficiently refines pose estimates over time while significantly reducing storage requirements. Experiments show that SG2Loc achieves competitive accuracy with far lower storage overhead than existing methods. The proposed coarse-to-fine optimization balances efficiency and precision, making the approach practical for resource-constrained applications. SG2Loc achieves performance similar to storage-intensive baselines, sometimes it is better in accuracy, sometimes slightly worse. Future work will focus on enhancing feature representations and runtime. The code will be made public.

7 Reproducibility and Use of Large Language Models

We will release the codebase, including data set splits, to enable reproducibility of all experiments. Large Language Models (LLMs) were used for minor language editing (grammar and phrasing).

References

Michal Adamkiewicz, Timothy Chen, Adam Caccavale, Rachel Gardner, Preston Culbertson, Jeannette Bohg, and Mac Schwager. Vision-only robot navigation in a neural radiance world. *IEEE Robotics and Automation Letters*, 7(2):4606–4613, 2022.

Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5297–5307, 2016.

Iro Armeni, Soroush Sax, Amir R Zamir, and Silvio Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5664–5673, 2019.

Axel Barrau and Silvere Bonnabel. Intrinsic filtering on lie groups with applications to attitude estimation. *IEEE Transactions on Automatic Control*, 60(2):436–449, 2014.

Gabriele Berton and Carlo Masone. Megaloc: One retrieval to place them all. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 2861–2867, 2025.

Michael Bloesch, Sammy Omari, Marco Hutter, and Roland Siegwart. Robust visual inertial odometry using a direct ekf-based approach. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 298–304. IEEE, 2015.

Eric Brachmann and Carsten Rother. Learning less is more-6d camera localization via 3d surface regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4654–4662, 2018.

Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Tom Sharp, Carsten Rother, and Roberto Cipolla. Dsac-differentiable ransac for camera localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6658–6666, 2017.

Eric Brachmann, Tommaso Cavallari, and Victor Adrian Prisacariu. Accelerated coordinate encoding: Learning to relocalize in minutes using rgb and poses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5044–5053, 2023.

Ondrej Chum and Jiri Matas. Matching with prosac-progressive sample consensus. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pp. 220–226. IEEE, 2005.

Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5828–5839, 2017.

Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 224–236, 2018.

Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8092–8101, 2019.

Johan Edstedt, Qiyu Sun, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. Roma: Robust dense feature matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19790–19800, 2024.

Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pp. 834–849. Springer, 2014.

Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. In *Communications of the ACM*, volume 24, pp. 381–395. ACM, 1981.

Christian Forster, Matia Pizzoli, and Davide Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *2014 IEEE international conference on robotics and automation (ICRA)*, pp. 15–22. IEEE, 2014.

Dieter Fox. Kld-sampling: Adaptive particle filters. *Advances in neural information processing systems*, 14, 2001.

Xiang Gao, Rui Wang, Nikolaus Demmel, and Daniel Cremers. Ldso: Direct sparse odometry with loop closure. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2198–2204. IEEE, 2018.

Justin Johnson, Ranjay Krishna, Michael Stark, Li Fei-Fei, et al. Image retrieval using scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3668–3678, 2015.

Nikhil Keetha, Avneesh Mishra, Jay Karhade, Krishna Murthy Jatavallabhula, Sebastian Scherer, Madhava Krishna, and Sourav Garg. Anyloc: Towards universal visual place recognition. *IEEE Robotics and Automation Letters*, 9(2):1286–1293, 2023.

Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pp. 2938–2946, 2015.

Fadi Khatib, Dror Moran, Guy Trostianetsky, Yoni Kasten, Meirav Galun, and Ronen Basri. Gsvisloc: Generalizable visual localization for gaussian splatting scene representations. *arXiv preprint arXiv:2508.18242*, 2025.

Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, pp. 225–234. IEEE, 2007.

Erwin Kruppa. Zur ermittlung des objektes aus zwei perspektiven mit innerer orientierung. *Sitzungsberichte der Kaiserlichen Akademie der Wissenschaften in Wien. Mathematisch-Naturwissenschaftliche Klasse*, 122(2a):1939–1948, 1913.

Viktor Larsson and contributors. PoseLib - Minimal Solvers for Camera Pose Estimation, 2020. URL `https://github.com/vlarsson/PoseLib`.

Seongwon Lee, Hongje Seong, Suhyeon Lee, and Euntai Kim. Correlation verification for image retrieval. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5374–5384, 2022.

Wonkwang Lee, Whie Jung, Han Zhang, Ting Chen, Jing Yu Koh, Thomas Huang, Hyungsuk Yoon, Honglak Lee, and Seunghoon Hong. Revisiting hierarchical approach for persistent long-term video prediction. *arXiv preprint arXiv:2104.06697*, 2021.

Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. Keyframe-based visual–inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34:314–334, 2015.

Yunpeng Li, Noah Snavely, Dan Huttenlocher, and Pascal Fua. Worldwide pose estimation using 3d point clouds. In *European conference on computer vision*, pp. 15–29. Springer, 2012.

Changkun Liu, Shuai Chen, Yash Bhalgat, Siyan Hu, Ming Cheng, Zirui Wang, Victor Adrian Prisacariu, and Tristan Braud. Gs-cpr: Efficient camera pose refinement via 3d gaussian splatting. *arXiv preprint arXiv:2408.11085*, 2024.

Simon Lynen, Bernhard Zeisl, Dror Aiger, Michael Bosse, Joel Hesch, Marc Pollefeys, Roland Siegwart, and Torsten Sattler. Large-scale, real-time visual–inertial localization revisited. *The International Journal of Robotics Research*, 39(9):1061–1084, 2020.

Dominic Maggio, Marcus Abate, Jingnan Shi, Courtney Mario, and Luca Carlone. Loc-nerf: Monte carlo localization using neural radiance fields. *arXiv preprint arXiv:2209.09050*, 2022.

Dominic Maggio, Hyungtae Lim, and Luca Carlone. Vggt-slam: Dense rgb slam optimized on the sl (4) manifold. *arXiv preprint arXiv:2505.12549*, 2025.

Wugang Meng, Tianfu Wu, Huan Yin, and Fumin Zhang. Nurf: Nudging the particle filter in radiance fields for robot visual localization. *IEEE Transactions on Cognitive and Developmental Systems*, 2025.

Yang Miao, Francis Engelmann, Olga Vysotska, Federico Tombari, Marc Pollefeys, and Dániel Béla Baráth. Scenegraphloc: Cross-modal coarse visual localization on 3d scene graphs. In *European Conference on Computer Vision*, pp. 127–150. Springer, 2024.

Hans P Moravec. Obstacle avoidance and navigation in the real world by a seeing robot rover. 1980.

Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017.

Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: A versatile and accurate monocular slam system. *IEEE Transactions on robotics*, 31(5):1147–1163, 2015.

Riku Murai, Eric Dexheimer, and Andrew J Davison. Mast3r-slam: Real-time dense slam with 3d reconstruction priors. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 16695–16705, 2025.

Vojtech Panek, Zuzana Kukelova, and Torsten Sattler. Meshloc: Mesh-based visual localization. In *European Conference on Computer Vision*, pp. 589–609. Springer, 2022.

Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. In *IEEE Transactions on Robotics*, volume 34, pp. 1004–1020. IEEE, 2018.

Jérôme Revaud, César De Souza, Martin Humenberger, and Philippe Weinzaepfel. R2d2: Reliable and repeatable detector and descriptor. In *Advances in Neural Information Processing Systems*, volume 32, pp. 12405–12415, 2019.

Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12716–12725, 2019.

Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4938–4947, 2020.

Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 667–674, 2011.

Torsten Sattler, Akihiko Torii, Josef Sivic, Marc Pollefeys, and Masatoshi Okutomi. Efficient & effective prioritized matching for large-scale image-based localization. In *IEEE transactions on pattern analysis and machine intelligence*, volume 38, pp. 1744–1756. IEEE, 2016.

Torsten Sattler, Michal Havlena, Konrad Schindler, and Marc Pollefeys. Large-scale location recognition and the geometric burstiness problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4877–4886, 2017.

Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, et al. Benchmarking 6dof outdoor visual localization in changing conditions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8601–8610, 2018.

Thomas Schneider, Marcin Dymczyk, Marius Fehr, Kevin Egger, Simon Lynen, Igor Gilitschenski, and Roland Siegwart. maplab: An open framework for research in visual-inertial mapping and localization. *IEEE Robotics and Automation Letters*, 3(3):1418–1425, 2018.

Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021.

Alexander Veicht, Paul-Edouard Sarlin, Philipp Lindenberger, and Marc Pollefeys. Geocalib: Learning single-image calibration with geometric optimization. In *European Conference on Computer Vision*, pp. 1–20. Springer, 2024.

Florian Walch, Caner Hazirbas, Laura Leal-Taixé, Torsten Sattler, Frank Hutter, and Marc Pollefeys. Image-based localization using lstms for structured feature correlation. In *Proceedings of the IEEE international conference on computer vision*, pp. 6046–6055, 2017.

Johanna Wald, Armen Avetisyan, Nassir Navab, Federico Tombari, and Matthias Nießner. Rio: 3d object instance re-localization in changing indoor environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7658–7667, 2019.

Fangjinhua Wang, Xudong Jiang, Silvano Galliani, Christoph Vogel, and Marc Pollefeys. Glace: Global local accelerated coordinate encoding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21562–21571, 2024.

Ke Wu, Guanbin Wang, Wenqi Zeng, Qinghao Wang, Yang Wang, Xiaodong Huang, and Zheng Ding. Multi-modal graph neural network for joint reasoning on vision and scene graph. In *Proceedings of the 28th ACM International Conference on Multimedia*, pp. 2067–2075, 2020.

Shun-Cheng Wu, Johanna Wald, Keisuke Tateno, Nassir Navab, and Federico Tombari. Scenegraphfusion: Incremental 3d scene graph prediction from rgb-d sequences. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7515–7525, 2021.

Hanwang Zhang, Zawlin Zawadzki, Jianwei Cai, et al. Visual translation embedding network for visual relation detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6789–6797, 2017.

Kang Zhou, Chi Guo, and Huyin Zhang. Visual navigation via reinforcement learning and relational reasoning. In *2021 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/IOP/SCI)*, pp. 131–138. IEEE, 2021.

Lei Zhou, Zixin Luo, Tianwei Shen, Jiahui Zhang, Mingmin Zhen, Yao Yao, Tian Fang, and Long Quan. Kfnet: Learning temporal camera relocalization using kalman filtering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4919–4928, 2020.

A Additional Ablations on Supervision Signals

To understand the individual contribution of each supervision signal, we run the particle filter using only a single score at a time (semantic, depth, or RGB). Table 8 reports the resulting localization accuracy and pose recall on 3RScan (Wald et al., 2019). *SG2Loc w/ semantic* achieves the lowest median position error and consistently good performance in all metrics. *SG2Loc w/ depth* has the lowest median rotation error. In contrast, *SG2Loc w/ RGB* performs worse in terms of position and rotation accuracy, but interestingly achieves the highest recall at the strict threshold $(10\text{cm}, 5^\circ)$. These results suggest that all three supervision signals contribute useful information to the final localization performance.

Table 8: **Ablations on supervision signals on 3RScan**. We report the average and median position error (in meters) and average and median rotation error (in degrees) for sequences of 5 frames, where lower values indicate better performance. We also measure recall at thresholds $(10\text{cm}, 5^\circ)$ and $(25\text{cm}, 10^\circ)$, where higher values indicate better performance.

Method	Mean		Median		Recall	
	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)	R@10cm, 5°	R@25cm, 10°
SG2Loc w/ semantic	**1.20**	**21.44**	**0.66**	8.77	0.08	0.26
SG2Loc w/ depth	1.44	39.01	0.96	**6.42**	0.09	**0.32**
SG2Loc w/ RGB	1.79	44.78	1.51	17.37	**0.14**	0.20

B Ablations on ScanNet

We evaluate the contribution of different supervision signals on the ScanNet (Dai et al., 2017) dataset. Table 9 shows the mean and median localization errors, and Table 10 reports the recall at thresholds $(10\text{cm}, 5^\circ)$ and $(25\text{cm}, 10^\circ)$. Our proposed SG2Loc method performs the additional optimization initialized from the maximum likelihood estimate of *SG2Loc w/ semantic+depth+RGB* (second row in Tab. 9 and Tab. 10). It achieves the best performance across all metrics, only the rotation median is slightly better without the final optimization.

Table 9: **Ablations on ScanNet (mean and median)**. Localization accuracy on the ScanNet dataset for different supervision inputs. We report the average and median position error (in meters) and average and median rotation error (in degrees) for sequences of length 5, 10 and 25 frames. Lower values indicate better localization performance.

Method	5 frames				10 frames				25 frames			
	Mean		Median		Mean		Median		Mean		Median	
	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)
SG2Loc w/ semantic	1.08	21.21	0.73	7.40	1.00	18.55	0.57	6.28	0.86	15.69	0.43	5.95
SG2Loc w/ semantic+depth+RGB	0.56	11.92	0.15	**2.45**	0.54	10.23	0.14	**2.08**	0.46	10.42	0.13	**1.99**
SG2Loc (Ours)	**0.53**	**11.55**	**0.12**	2.55	**0.44**	**8.14**	**0.10**	2.30	**0.29**	**4.55**	**0.09**	2.29

Table 10: **Ablations on ScanNet (pose recall).** We report recall at thresholds $(10\text{cm}, 5^\circ)$ and $(25\text{cm}, 10^\circ)$ for localization sequences of length 5, 10 and 25 frames. Recall measures the fraction of errors that fall below those thresholds. Higher values indicate better performance.

Method	5 frames		10 frames		25 frames	
	R@10cm, 5°	R@25cm, 10°	R@10cm, 5°	R@25cm, 10°	R@10cm, 5°	R@25cm, 10°
SG2Loc w/ semantic	0.12	0.33	0.12	0.34	0.14	0.40
SG2Loc w/ semantic+depth+RGB	0.26	0.69	0.28	0.73	0.32	0.75
SG2Loc (Ours)	**0.41**	**0.72**	**0.49**	**0.79**	**0.58**	**0.80**

C Qualitative Results

Figure 4 shows a failure case of the particle filter on a 5-frame sequence from the 3RScan dataset (Wald et al., 2019). Although the particle distribution narrows over time, the final estimate still results in a coarse localization with a position error of 1.06 meters and a rotation error of 6.6

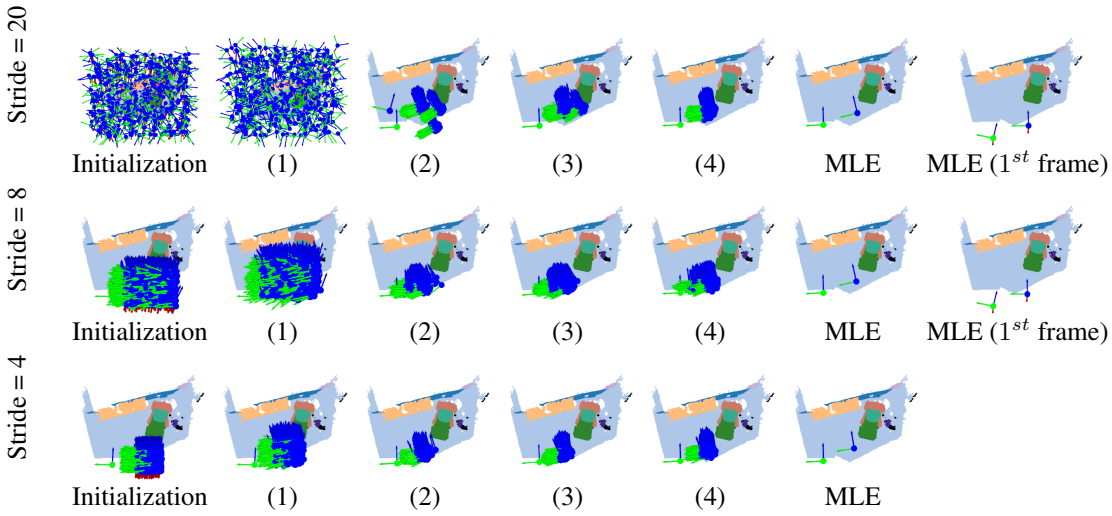

Figure 4: **A failure case** of our coarse-to-fine particle filter on a 5-image sequence, first running with a stride of 20, 8, and finally 4 (the stride controls the rate of downsampling the images). The first column shows the initial random particle distribution, gradually narrowing the search space in subsequent rounds. Each next column represents particle updates after integrating the n^{th} image (indicated below each plot). Then, the Maximum Likelihood Estimate (MLE) is shown in blue, and the GT pose in green, back-propagated to the 1^{st} frame for the next optimization round. The position error for this example is 1.06 meter and rotation error 6.60°. This failure is potentially caused by the uninformative input views (all looking very similar) visualized in Fig. 5.

degrees. This example highlights a limitation of our method. The input views in the query sequence are visually very similar (Figure 5), with little change in perspective. In such cases, the information in the image sequence is not discriminative enough to resolve pose ambiguities. As a result, the method struggles to converge to a precise pose and only returns a coarse estimate. Still, the failure is less severe than for state-of-the-art methods like HLoc (Sarlin et al., 2019), which produces an average position error above 10^{12} meters on 3RScan dataset (Wald et al., 2019).

Figure 5: **Query images and MLE view.** The five query images (a)–(e) used in Figure 4 for the course-to-fine particle filter on the 3RScan dataset (Wald et al., 2019). After the fifth image (e), we retrieve the Maximum Likelihood Estimate (MLE) pose. Image (f) shows the segmented 3D mesh projected into a virtual camera located at the MLE pose. Colors denote object instances.

D LARGE-SCALE ENVIRONMENTS

We conducted an additional experiment simulating a larger-scale environment. Since the available datasets do not provide large-scale, object-annotated environments (all scenes are apartment-sized), we merged every 3 ScanNet scenes into a single environment, initializing particles across the en-

tire space and allowing the filter to automatically determine the correct scene. We report both the retrieval accuracy (R^t@1, i.e., fraction of cases where the correct room was identified) and the final pose accuracy. The results for 50-frame sequences, shown below, demonstrate that our method can successfully localize in this scenario. It achieves slightly lower retrieval performance compared to the proposed sequential SceneGraphLoc and the original SceneGraphLoc, while still producing accurate poses. The lower recall could potentially be improved by increasing the number of initial particles. Given its efficiency, running the sequential SceneGraphLoc variant for the coarse localization stage remains preferable. As per standard practice in localization Sarlin et al. (2019), SG2Loc can provide an accurate pose after the coarse location has been identified by the proposed sequential SceneGraphLoc.

Table 11: Localization results **large-scale environment** on ScanNet.

Method	R^t@1	**Mean Pos. (m)** ↓	**Mean Rot. (°)** ↓	**Med. Pos. (m)** ↓	**Med. Rot. (°)** ↓	**R@10cm,5°** ↑	**R@25cm,10°** ↑
SGL (Miao et al., 2024)	92.4	–	–	–	–	–	–
Seq. SGL (Ours)	95.1	–	–	–	–	–	–
SG2Loc (Ours)	85.7	0.24	7.21	0.16	2.25	0.33	0.80

E SPEED-UP EXPERIMENTS

To reduce runtime, we ran the following experiment: we ran the particle filter only once per frame with the lowest resolution (stride 20). Averaged over sequence lengths 5 and 25, this achieves 1.6 seconds per frame with a modest accuracy drop. Mean and median errors and recalls are reported in Table 12 and Table 13. Compared to our coarse-to-fine method, accuracy decreases slightly but remains comparable to baselines (e.g., HLoc).

Table 12: **Localization accuracy for coarse SG2Loc** on the 3RScan dataset. We report the avg. and median position error (m) and avg. and median rotation error (°) for sequences of length 5 and 25 frames for the coarse variant of SG2Loc. This variant is only running a single particle filter pass with the lowest resolution (stride 20) to improve runtime.

Method	5 frames				25 frames			
	Mean		Median		Mean		Median	
	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)	Pos. (m)	Rot. (°)
HLoc	2.05×10^{12}	46.68	**0.18**	6.32	1030.90	**22.74**	**0.08**	**2.44**
SG2Loc (Ours)	<u>2.43</u>	**21.24**	<u>0.20</u>	**3.27**	**6.15**	<u>22.93</u>	<u>0.14</u>	<u>2.73</u>
SG2Loc (Coarse) (Ours)	**2.25**	<u>24.37</u>	0.31	<u>5.05</u>	<u>8.08</u>	27.10	<u>0.14</u>	3.44

Table 13: **Pose recalls for coarse SG2Loc** on the 3RScan dataset. We report recall at thresholds $(25\text{cm}, 2°)$ for localization sequences of length 5 and 25 frames: Position R@0.25m / Rotation R@2° / R@0.25m, 2°. Recall measures the fraction of errors that fall below those thresholds, higher values indicate better performance.

Method	5 frames	25 frames
HLoc	**0.54** / 0.28 / <u>0.27</u>	**0.74** / **0.43** / **0.43**
SG2Loc (Ours)	<u>0.53</u> / **0.38** / **0.32**	<u>0.59</u> / **0.43** / <u>0.39</u>
SG2Loc (Coarse) (Ours)	0.46 / <u>0.31</u> / <u>0.27</u>	0.58 / <u>0.38</u> / 0.36

F EXPERIMENT USING SLAM POSES

We additionally report results using DROID-SLAM Teed & Deng (2021) for the relative motion on ScanNet (sequence length 5) instead of the quasi ground truth. The performance is comparable, showing that SG2Loc is robust to moderate drift in the motion estimate:

Method	Median Pos (m)	Median Rot (°)	Recall@0.25m, 2°
HLoc	**0.09**	2.63	**0.38**
HLoc w/ SLAM poses	0.15	2.67	0.34
SG2Loc (Ours)	0.12	**2.55**	0.36
SG2Loc w/ SLAM poses	0.16	2.63	0.36

Table 14: Comparison of localization performance using DROID-SLAMTeed & Deng (2021) poses.