2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Struct-Loc: Confidence-Aware Structural Localization via Hierarchical Point Cloud Registration

Csaba M. Józsa, Attila Bóta, Krisztián Zs. Varga and Ferenc Kovács

***Abstract*— Localization systems often rely heavily on visual information, which can degrade under challenging conditions such as variable lighting, dynamic objects, or repetitive textures. To enhance robustness beyond single-image methods, we model localization as a structural point cloud registration problem, leveraging motion continuity and geometric consistency over time. This formulation reduces sensitivity to transient occlusions and appearance changes, enabling the system to resolve ambiguities that single-image techniques often cannot.**

In this work, we introduce Struct-Loc, a localization framework that advances structural point cloud registration through confidence-aware hierarchical localization. By estimating the reliability of structural regions and incorporating it into the matching process, Struct-Loc generates robust descriptors tailored for pose estimation.

To achieve near real-time performance, Struct-Loc combines efficient point convolutional encoders, a caching mechanism, and a hierarchical coarse-to-fine matching strategy that progressively narrows the search space. It consistently outperforms strong baselines in both accuracy and runtime, while achieving a 100× compression of the global map compared to COLMAP, significantly improving storage efficiency. We validate Struct-Loc on the LaMAR benchmark, demonstrating its effectiveness and robustness under real-world conditions.

I. Introduction

Localization entails estimating an agent's precise 6 degree-of-freedom (DoF) pose within an environment by determining the transformation between its local coordinate frame and a global map frame. Accurate localization is critical for applications such as autonomous vehicles, robotics, and augmented reality, where precise positioning supports effective navigation and task execution. Despite its importance, localization remains challenging.

Variations in illumination cause inconsistent feature detection and matching due to shadows, reflections, and lighting changes. Furthermore, camera heterogeneity—arising from differences in sensors, lenses, and calibrations [1]—introduces discrepancies in captured images, impacting feature extraction and pose estimation accuracy. Thus, localization algorithms must be robust to hardware variations. As environments grow complex and dynamic, identifying repeatable and distinctive features, alongside extracting global image descriptors for retrieval, is essential. Many methods rely on single-image techniques for simplicity but often lack sufficient context to resolve ambiguities in repetitive or feature-sparse scenes.

All authors are with Nokia Bell Labs, Budapest, Hungary. {csaba.jozsa, attila.bota, krisztian.varga, ferenc.2.kovacs}@nokia-bell-labs.com.

The authors used NokiaGPT, an internal generative-AI tool developed at Nokia, for language refinement and grammar checking. The authors take full responsibility for the content of the paper.

Robustness can be improved using probabilistic methods like particle filtering [2], [3], which maintain multiple pose hypotheses to handle ambiguity. However, this comes at high computational cost, especially in large-scale environments requiring numerous particles.

A more scalable direction, explored in several prior works, is to formulate localization as a *point cloud registration problem* [4], [5]. Instead of estimating poses from single frames, structural point clouds are constructed from sequences and then registered to a pre-built global map, leveraging motion continuity and geometric consistency. This approach improves robustness and reduces ambiguity, but existing methods often treat all candidate correspondences uniformly, regardless of their reliability for pose estimation.

We propose Struct-Loc, a localization framework that advances this registration paradigm with *confidence-aware hierarchical localization*. For each coarse superpoint, Struct-Loc predicts both a feature descriptor and a confidence score that reflects the reliability of that region. After coarse correspondences are established, fine-level superpoints within each patch are matched and assigned correspondence scores. Each fine correspondence is then weighted by: the coarse superpoint confidences, the similarity of coarse descriptors, and the fine-level correspondence score. Pose estimation is performed using a weighted least-squares solver, and the confidences are trained end-to-end by directly supervising the predicted pose against ground truth. This design adaptively increases the influence of structurally distinctive regions while downweighting ambiguous ones.

The primary target application of Struct-Loc is robust, scene-specific localization in large indoor environments such as warehouses, industrial halls, or office buildings. In these scenarios, a global map is constructed once and the model is trained specifically for that environment. This assumption enables Struct-Loc to achieve both robustness and efficiency in practical deployments, which are key requirements for many commercial and industrial applications.

An important property of Struct-Loc is that its recall improves as more information about the environment is accumulated along a trajectory. Localization can therefore be deferred until the uncertainty of dead-reckoning grows too large, at which point Struct-Loc realigns the trajectory and eliminates drift. This is especially critical in environments with many simultaneous clients, such as fleets of robots or smartphones, where high recall directly determines how many users can be served reliably. In contrast, single-frame solutions exhibit much lower recall, leading to wasted computation on failed localization attempts.

979-8-3315-8160-2/26/$31.00 ©2026 IEEE

For real-time alignment in large-scale settings, Struct-Loc pre-encodes and caches the global map after training, removing the need to repeatedly process its features during inference. By combining architectural efficiency with confidence-aware hierarchical matching, Struct-Loc delivers precise, high-recall localization at low runtime and achieves substantial map compression, making it well-suited for real-world deployment in dynamic, complex environments where vision-only methods often fail.

II. Related Work

Learning-Based Localization The three main components of single image-based localization are global image descriptor computation, feature point extraction and feature matching. Global image retrieval [6], [7], [8], [9] is the key step for localization. Deep learning models like [6] and [7] are effectively aggregating local features into compact global descriptors using contrastive learning [10], [11]. In [9] global and local descriptors are integrated together, employing global retrieval methods for coarse candidate selection and local feature matching techniques for fine-grained pose estimation. This multi-stage process balances computational efficiency with the precision needed for large-scale environments.

While classical feature point extraction algorithms rely on hand-crafted criteria and gradient-based statistics [12], [13], [14], recent research has increasingly focused on applying Convolutional Neural Networks (CNNs) for both feature detection [15], [16], [17], [18] and descriptor learning [9], [19]. At the local level, methods such as SuperPoint and D2-Net [15], [17] extract fine-grained features that are robust to variations in scale, rotation, and viewpoint. Some approaches emphasize precise localization [13], while others aim to optimize both repeatability and distinctiveness to enhance performance in correspondence-based tasks [15], [16]. Additional efforts focus on invariance to specific transformations [20] or the extraction of high-fidelity descriptors that remain reliable under significant perspective changes and in scenes containing ambiguous or unreliable regions [8], [18], [17].

A powerful local feature matching approach was proposed in [21], [22], combining the expressive capabilities of Transformers [23] with optimal transport theory [24] to solve the partial assignment problem between two sets of interest points in a data-driven manner. Building on similar principles, methods such as LoFTR and MatchFormer [25], [26] bypass traditional keypoint detection and descriptor computation by directly establishing pixel-wise dense correspondences. While this significantly enhances robustness, it comes at the cost of increased computational runtime.

Sequence-Based Localization While single-image representation has been extensively explored in the literature, the use of temporal or sequential information for visual positioning remains relatively underdeveloped. Single-image localization techniques often struggle in visually ambiguous or dynamic environments. Sequence-based methods address this limitation by leveraging temporal continuity to enhance robustness. SeqSLAM introduced a foundational approach that matches sequences of images rather than individual frames, enabling localization under significant appearance changes due to lighting or weather conditions [27]. Its ability to exploit sequential dependencies makes it effective in environments with drastic visual variations.

Building on this idea, SeqNet [28] employs a hierarchical sequence-matching architecture, learning robust descriptors for sequence-based retrieval. By leveraging temporal structure, SeqNet outperforms traditional single-image methods, especially in environments with repetitive patterns or sparse features. These approaches underscore the importance of temporal context for resolving visual ambiguities and improving localization robustness. However, they do not incorporate structural information.

Point Cloud Registration Recent advancements in deep learning have significantly improved registration techniques [29], [30], [31], [32], [33], [34]. GeoTransformer [35] leverages a transformer-based architecture to model long-range geometric dependencies. Its attention mechanisms capture complex relationships between local and global geometric features, enabling accurate alignment even in the presence of partial occlusions or noise. However, the method's computational demands can be prohibitive for real-time applications. PREDATOR emphasizes robust local correspondence estimation. By focusing on overlap regions and mitigating the effects of noise and outliers, PREDATOR achieves reliable results in sparse and challenging environments. Its lightweight design also lends itself to integration with other registration pipelines [36]. CoFiNet [34], HRegNet [37], Regformer [38], and [39] adopts a coarse-to-fine strategy, initially aligning global features before refining them at a local level. This hierarchical approach ensures robustness to scale variations and improves registration accuracy in dynamic and large-scale scenes.

While prior work has also considered confidence measures, for example through overlap or saliency prediction, these are typically used to guide local correspondence filtering rather than integrated directly into pose estimation. Struct-Loc advances this direction with *confidence-aware hierarchical matching*, where coarse-level confidences are regressed, propagated through fine-level matching, and optimized end-to-end with pose supervision. Beyond this architectural novelty, our deployment setting enables strong caching mechanisms, substantial global map compression, outperforming traditional approaches such as COLMAP [40] in storage efficiency, real-time inference, and robust accuracy in large-scale environments—making Struct-Loc both a methodological and practical step forward for reliable localization.

III. Method

We formulate localization as a hierarchical point cloud registration problem between a pre-built global structural map $\mathcal{P}^S = \{p_i\}_{i=1}^{N_S}$ and an online local query map $\mathcal{Q}^S = \{q_j\}_{j=1}^{M_S}$. The global map is constructed offline using dense SLAM from camera or LiDAR scans, while the local map is obtained online from short sequences with lightweight visual or visual-inertial SLAM.

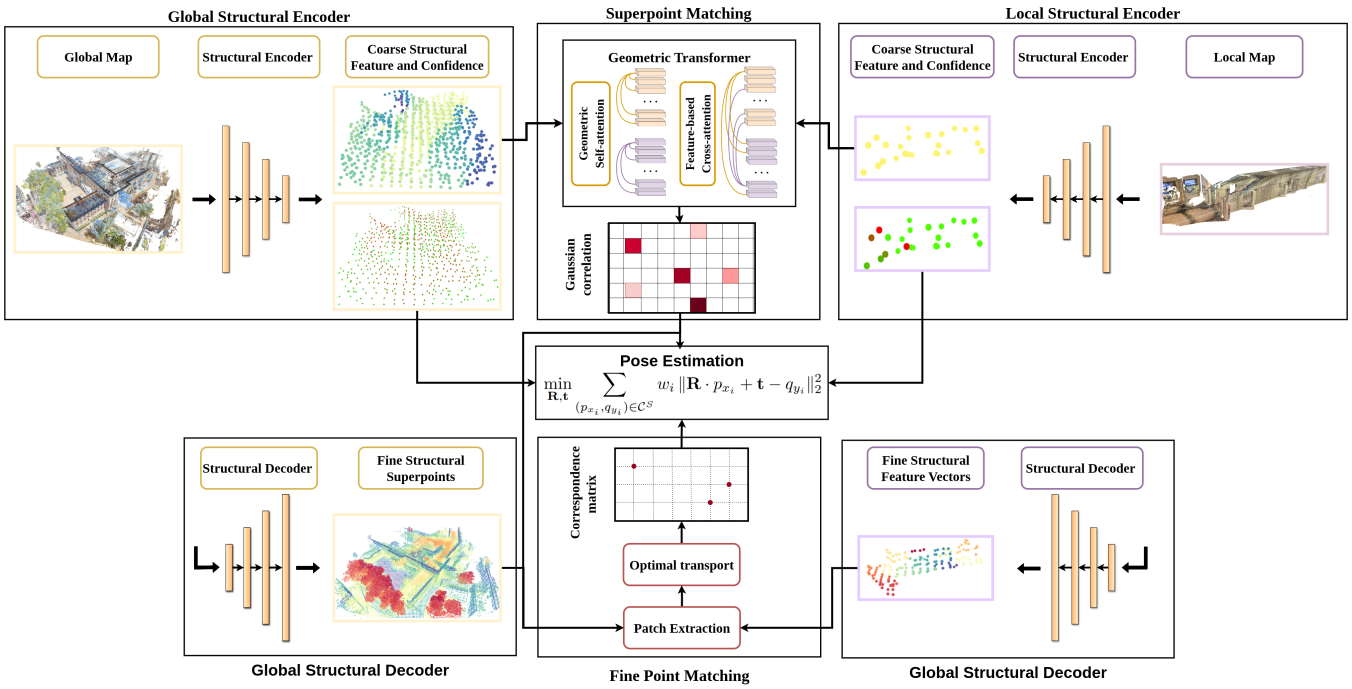

Fig. 1: Struct-Loc system architecture.

Both maps are processed by a KPConv-FPN backbone[41], [42], producing coarse superpoints with descriptors and regressed confidences, as well as fine-level superpoints linked to their coarse parents. Localization proceeds in a coarse-to-fine manner. Coarse correspondences are first established between the global and local superpoints, and the top-k matches are retained according to descriptor similarity and confidence. For each coarse match, the associated fine-level patches are extracted, and one-to-one correspondences are computed using optimal transport.

The final rigid transformation $\mathbf{T} = \{\mathbf{R}, \mathbf{t}\} \in SE(3)$ aligning $\mathcal{Q}^S$ to $\mathcal{P}^S$ is estimated by solving a weighted least-squares problem

$$\min_{\mathbf{R},\mathbf{t}} \sum_{(p_{x_i},q_{y_i})\in\mathcal{C}^S} w_i \left\|\mathbf{R}\cdot p_{x_i}+\mathbf{t}-q_{y_i}\right\|_2^2, \tag{1}$$

where $\mathcal{C}^S$ is the set of fine-level correspondences. Each correspondence is assigned a hierarchical weight

$$w_i = c_u \cdot c_v \cdot s_{uv}^{\text{coarse}} \cdot s_i^{\text{fine}}, \tag{2}$$

with c_u and c_v the confidences of the matched coarse superpoints, s_{uv}^{coarse} their descriptor similarity, and s_i^{fine} the fine-level correspondence score. This hierarchical formulation ensures that reliable regions dominate the alignment, while ambiguous areas are downweighted. The next subsections describe the encoder/decoder, hierarchical matching, and training objectives in detail.

A. Structural Feature Encoder/Decoder

To extract hierarchical features from unstructured 3D point clouds, we adopt a KPConv-FPN backbone [41], [42] that leverages kernel point convolutions KPConv across multiple resolution levels. KPConv operates on point sets by convolving over local neighborhoods defined around a set of query points. In our setting, the query points—referred to as superpoints—are obtained by voxel-based downsampling of the point cloud at each encoder layer. The convolution itself aggregates features from support points, which are the input points at that layer. As the encoder deepens, voxel sizes grow progressively, allowing superpoints to capture increasingly coarse geometric structures. Let the original point cloud be $\mathcal{P}^{(0)} = \{\mathbf{p}_i^{(0)} \in \mathbb{R}^3\}_{i=1}^{N}$, where each point $\mathbf{p}_i^{(0)}$ has an associated feature vector $\mathbf{f}_i^{(0)}$. At encoder layer $l \in \{1, \ldots, L\}$, a voxel grid with voxel size $v^{(l)} = 2^{l-1}v^{(1)}$ partitions the input point set $\mathcal{P}^{(l-1)}$ into disjoint voxel cells $V_j^{(l)}$, indexed by j. The superpoints $\mathcal{S}^{(l)} = \{\mathbf{s}_j^{(l)}\}_{j=1}^{N^{(l)}}$ at layer l are defined as the centroids of the points in each voxel: $\mathbf{s}_j^{(l)} = \frac{1}{|V_j^{(l)}|} \sum_{\mathbf{p}_i^{(l-1)} \in V_j^{(l)}} \mathbf{p}_i^{(l-1)}$.

To compute features on superpoints, KPConv is applied using the input point set $\mathcal{P}^{(l-1)}$ as support points and $\mathcal{S}^{(l)}$ as query points. Each query point $\mathbf{s}_j^{(l)}$ aggregates local geometric and semantic context via a radius-based patch: $\text{Patch}(\mathbf{s}_j^{(l)}) = \left\{ \mathbf{p}_i^{(l-1)} \in \mathcal{P}^{(l-1)} \;\middle|\; \|\mathbf{p}_i^{(l-1)} - \mathbf{s}_j^{(l)}\| \le r^{(l)} \right\}$, where $r^{(l)}$ is the convolution radius. The feature at $\mathbf{s}_j^{(l)}$ is obtained by applying KPConv over its patch: $\mathbf{f}_j^{(l)} = \text{KPConv}\left(\mathbf{s}_j^{(l)}, \{(\mathbf{p}_i^{(l-1)}, \mathbf{f}_i^{(l-1)}) \in \text{Patch}(\mathbf{s}_j^{(l)})\}\right)$. The set of features at layer l is denoted by $\mathcal{F}^{(l)} = \{\mathbf{f}_j^{(l)}\}_{j=1}^{N^{(l)}}$. In the decoder, we define $L-1$ layers indexed by $l' = L+1, \ldots, 2L-1$, each aimed at progressively reconstructing finer superpoint features. The superpoint positions in decoder

layer $l' = L + i$ are inherited from encoder layer $L - i$, i.e., $\mathcal{S}^{(l')} = \mathcal{S}^{(L-i)}$, with the finest decoder layer at $l' = 2L - 1$ using the original fine-level superpoints $\mathcal{S}^{(1)}$. We denote the coarse superpoints at the final encoder layer as $\hat{\mathcal{P}} = \mathcal{S}^{(L)}$, with features $\hat{\mathcal{F}} = \mathcal{F}^{(L)}$, and the fine superpoints at the final decoder layer as $\tilde{\mathcal{P}} = \mathcal{S}^{(2L-1)} = \mathcal{S}^{(1)}$, with features $\tilde{\mathcal{F}} = \mathcal{F}^{(2L-1)}$. To link the hierarchical structure, we define a mapping $\pi^{(l)} : \mathcal{S}^{(l-1)} \to \mathcal{S}^{(l)}$ such that $\pi^{(l)}(\mathbf{s}_i^{(l-1)}) = \mathbf{s}_j^{(l)}$ if $\mathbf{s}_i^{(l-1)} \in V_j^{(l)}$, i.e., each superpoint at layer $l-1$ is assigned to the voxel centroid $\mathbf{s}_j^{(l)}$ that contains it.

The overall hierarchical mapping from fine to coarse superpoints is then given by the composition: $\Pi = \pi^{(L)} \circ \pi^{(L-1)} \circ \cdots \circ \pi^{(2)}$, which associates each fine superpoint in $\tilde{\mathcal{P}} = \mathcal{S}^{(1)}$ to a coarse superpoint in $\hat{\mathcal{P}} = \mathcal{S}^{(L)}$. Accordingly, the fine patch corresponding to a coarse superpoint $\mathbf{s}_j^{(L)}$ is defined as

$$G(\mathbf{s}_j^{(L)}) = \left\{ \mathbf{s}_i^{(1)} \in \mathcal{S}^{(1)} \;\middle|\; \Pi(\mathbf{s}_i^{(1)}) = \mathbf{s}_j^{(L)} \right\}. \tag{3}$$

This construction enables each coarse superpoint in the final encoder layer to be associated with a set of semantically and spatially related fine superpoints in the final decoder layer.

B. Superpoint Confidence Regression

Let $\hat{\mathcal{P}} = \{\hat{\mathbf{p}}_u\}$ and $\hat{\mathcal{F}} = \{\hat{\mathbf{f}}_u\}$ denote the coarse superpoint positions and features (from the encoder). We regress a scalar confidence $c_u \in [0, 1]$ per coarse superpoint via two KPConv layers that aggregate from coarse neighbors.

Each coarse superpoint u indexes a fine patch $G(u)$ (Eq. 3), but only a subset of its fine points is informative for localization. The confidences $\{c_u\}$ are trained end-to-end via the differentiable weighted Procrustes solver and an SE(3) pose loss, so superpoints whose patches contain more geometrically distinctive (pose-supporting) fine points naturally attain higher c_u, while ambiguous regions are downweighted in pose estimation.

C. Coarse Superpoint Matching

Superpoint matching is performed at a coarse level to establish initial correspondences between regions of the point clouds. Inspired by [35], [34], [21], [25], before computing correlation scores between the coarse global superpoints $\hat{\mathcal{P}}^S$ with their input features $\hat{\mathcal{F}}^{\hat{\mathcal{P}}}$ and coarse local superpoints $\hat{\mathcal{Q}}^S$ with features $\hat{\mathcal{F}}^{\hat{\mathcal{Q}}}$, we apply the geometric transformer [35] attention mechanism to the coarse feature vectors.

This attention updates the input feature matrices $\hat{\mathcal{F}}^{\hat{\mathcal{P}}}$ and $\hat{\mathcal{F}}^{\hat{\mathcal{Q}}}$ into spatial matching descriptor matrices $\hat{H}^{\mathcal{P}}$ and $\hat{H}^{\mathcal{Q}}$, respectively. Each row vector $\hat{h}_i^{\mathcal{P}}$ of $\hat{H}^{\mathcal{P}}$ corresponds to the descriptor for the i-th global superpoint $\hat{\mathbf{p}}_i$, and each row $\hat{h}_j^{\mathcal{Q}}$ of $\hat{H}^{\mathcal{Q}}$ corresponds to the j-th local superpoint $\hat{\mathbf{q}}_j$. The descriptors are then ℓ_2-normalized row-wise: $\tilde{h}_i^{\mathcal{P}} = \frac{\hat{h}_i^{\mathcal{P}}}{\|\hat{h}_i^{\mathcal{P}}\|_2}$, $\tilde{h}_j^{\mathcal{Q}} = \frac{\hat{h}_j^{\mathcal{Q}}}{\|\hat{h}_j^{\mathcal{Q}}\|_2}$, where $\tilde{h}_i^{\mathcal{P}}$ and $\tilde{h}_j^{\mathcal{Q}}$ remain row vectors representing normalized descriptors of superpoints $\hat{\mathbf{p}}_i$ and $\hat{\mathbf{q}}_j$, respectively.

Next, the Gaussian correlation matrix $C \in \mathbb{R}^{|\hat{\mathcal{Z}}^{\hat{\mathcal{P}}}| \times |\hat{\mathcal{Z}}^{\hat{\mathcal{Q}}}|}$ is computed between the normalized descriptors: $C_{i,j} = \exp\left(-\frac{1}{2\sigma^2}\left\|\tilde{h}_i^{\mathcal{P}} - \tilde{h}_j^{\mathcal{Q}}\right\|_2^2\right)$, where σ controls the bandwidth of the Gaussian kernel, affecting sensitivity to feature differences. To improve discriminative power and suppress ambiguous matches, a dual-normalization [43], [25] is applied to C, producing the normalized matrix $\tilde{C}$ that emphasizes mutual correspondences. Finally, for each local superpoint, the best corresponding global superpoint is selected, and only the top-k matches are retained, resulting in a filtered set of superpoint correspondences $\hat{C}$. These correspondences serve as input to the fine matching module.

D. Fine Point Matching

After establishing coarse correspondences between superpoints, we refine them by identifying dense point correspondences within each matched superpoint pair. For each coarse match $\hat{C}_i = (\hat{\mathbf{p}}_{x_i}, \hat{\mathbf{q}}_{y_i})$, fine patches $G_{x_i}^{\mathcal{P}}$ and $G_{y_i}^{\mathcal{Q}}$ are extracted as described in Eq.3. We compute a cost matrix $C_i \in \mathbb{R}^{|G_{x_i}^{\mathcal{P}}| \times |G_{y_i}^{\mathcal{Q}}|}$, which quantifies the similarity between fine points. Let $\tilde{F}_{x_i}^{\mathcal{P}}$ and $\tilde{F}_{y_i}^{\mathcal{Q}}$ denote the global and local feature matrices corresponding to the points in $G_{x_i}^{\mathcal{P}}$ and $G_{y_i}^{\mathcal{Q}}$, respectively. The cost matrix is computed as $C_i = \frac{\tilde{F}_{x_i}^{\mathcal{P}} (\tilde{F}_{y_i}^{\mathcal{Q}})^\top}{\sqrt{\tilde{d}}}$, where $\tilde{d}$ is the feature dimensionality used for normalization.

To handle unmatched points and outliers, we augment C_i by adding a dustbin row and column filled with a learnable parameter α, resulting in the extended matrix $\bar{C}_i$. The Sinkhorn algorithm [44] is then applied to $\bar{C}_i$ to compute a soft assignment matrix, which approximates the optimal transport plan and provides confidence scores for potential correspondences, while accounting for unmatched points via the dustbin entries. After removing the dustbin entries, we retain only reliable correspondences using mutual top-k selection, ensuring consistency and robustness in the final matches. Based on the retained fine point matches, a weighted Procrustes solver [45] is applied to estimate a pose matrix for each coarse superpoint pair. The final pose is selected using the pose selection strategy from [35].

E. Loss Functions

Struct-Loc is trained with three objectives: a weighted circle loss for coarse descriptors, a negative log-likelihood loss for fine correspondences, and an $SE(3)$ geodesic loss for the estimated pose. The total loss is

$$\mathcal{L}_{\text{total}} = \lambda_{\text{wc}}\, \mathcal{L}_{\text{wc}} + \lambda_{\text{fine}}\, \mathcal{L}_{\text{fine}} + \lambda_{\text{pose}}\, \mathcal{L}_{\text{pose}}, \tag{4}$$

where $\lambda_{\text{wc}}, \lambda_{\text{fine}}$, and λ_{pose} control the relative contributions of each term.

Coarse Level — Weighted Circle Loss: At the coarse level a weighted variant of circle loss is adopted [35], [46] to learn discriminative superpoint descriptors. Rather than treating all pairs equally, the contribution of each pair is modulated by its geometric overlap, so that informative positives and challenging negatives have a stronger impact during training.

The anchor set $\mathcal{A}$ is defined as the patches in $\mathcal{P}$ that have at least one positive patch in $\mathcal{Q}$. A pair $(\mathcal{G}_i^P, \mathcal{G}_j^Q)$ is considered positive if their overlap ratio is at least 10%, and negative

otherwise. For each anchor $i \in \mathcal{A}$, let ε_p^i and ε_n^i denote its positive and negative sets.

The weighted circle loss is given by

$$\mathcal{L}_{\text{wc}} = \frac{1}{|\mathcal{A}|} \sum_{i \in \mathcal{A}} \log \left[1 + \Big(\sum_{j \in \varepsilon_p^i} e^{\lambda_i^j \gamma (s_{ij} - \delta_p)} \Big) \Big(\sum_{k \in \varepsilon_n^i} e^{\gamma (\delta_n - s_{ik})} \Big) \right], \tag{5}$$

where s_{ij} is the similarity between descriptors, δ_p and δ_n are positive/negative margins, and $\lambda_i^j = (o_i^j)^\alpha$ is an overlap-based weight with overlap ratio $o_i^j \in [0, 1]$ and modulation strength $\alpha \geq 0$. The final loss is obtained by averaging the symmetric terms from both global to local and local to global directions.

Fine Level - Optimal Transport: For finer levels, a negative log-likelihood loss is applied to improve precision in the established correspondences. During training, the ground-truth superpoint correspondences are used instead of the predicted ones. For each ground-truth superpoint point pair, a set of ground-truth fine point correspondences $\mathcal{C}_i$ is extracted. The sets of unmatched points in the two patches are denoted as $\mathcal{I}_i$ and $\mathcal{J}_i$. The fine-level loss function is defined as:

$$\mathcal{L}_{\text{fine}} = - \sum_{(p_i, q_j) \in \mathcal{C}_i} \log \bar{c}_{i,j} - \sum_{p_i \in \mathcal{I}_i} \log \bar{c}_{i, m_i + 1} - \sum_{q_j \in \mathcal{J}_i} \log \bar{c}_{n_i + 1, j} \tag{6}$$

where $\bar{c}_{ij}$ is the similarity score from the score matrix $\bar{C}_i$ between points $\mathbf{p}_i$ and $\mathbf{q}_j$. This loss function encourages highly accurate matches by penalizing misaligned correspondences, leading to robust alignment across varying conditions and levels of detail.

Pose Loss: The predicted supervise rigid transformation is supervised using a pose loss defined on the Lie group $SE(3)$. Let $\mathbf{T}_{\text{pred}} = (\mathbf{R}_{\text{pred}}, \mathbf{t}_{\text{pred}}) \in SE(3)$ be the pose estimated from weighted Procrustes alignment. The regressed confidence scores c_u, c_v of the coarse superpoints directly influence the resulting transformation by modulating the relative importance of their associated fine correspondences.

Given the ground-truth pose $\mathbf{T}_{\text{gt}} = (\mathbf{R}_{\text{gt}}, \mathbf{t}_{\text{gt}})$, we compute the relative transformation $\Delta\mathbf{T} = \mathbf{T}_{\text{pred}}^{-1} \mathbf{T}_{\text{gt}} \in SE(3)$. The pose loss is defined as the squared geodesic distance on $SE(3)$:

$$\mathcal{L}_{\text{pose}} = \| \log(\Delta\mathbf{T}) \|_2^2, \tag{7}$$

where $\log : SE(3) \to \mathfrak{se}(3)$ maps the transformation to the Lie algebra, producing a 6D residual vector of rotational and translational errors. This geodesic formulation provides a smooth, rotation-aware metric that balances translation and rotation errors. Superpoints whose patches consistently yield reliable fine correspondences are assigned higher confidence, while those in ambiguous regions are naturally downweighted.

IV. Results

For training and evaluating Struct-Loc, we used the CAB scene from the LaMAR dataset [47], which spans 12,000 m² and consists of a multi-floor office building containing small and large offices, a kitchen, storage rooms, and two courtyards. The structural global map was constructed using 7728 NavVis image keyframes. Local maps were created using the HoloLens (HL) sessions. To generate these maps, we rendered depth images from the NavVis mesh using the front-left camera views from the HL recordings. To simulate realistic sensor conditions, a stereo noise model [48] is applied to the rendered ground-truth depth images.

Accuracy performance To evaluate the performance of our model, we compared it against the best-performing single-image and sequential localization baselines presented in [47]. The single-image model, referred to as **HLoc**, uses SuperPoint [15] for keypoint detection and description. Global image descriptors are computed by fusing NetVLAD [6] and APGeM [7], following the robust fusion approach proposed in [49], which has been shown to be representative of state-of-the-art performance [50]. Based on the global descriptors, the top 5, 10, or 20 most relevant reference images are retrieved. Local features are then matched using SuperGlue [21]. The matched keypoints in the reference images are lifted to 3D using the reference mesh, and the camera pose is estimated using PnP with RANSAC. The Sequential HLoc (**Seq HLoc**) baseline extends this approach to sequences of images. The single-image localization steps are applied independently to each image, followed by a rigid alignment and pose graph optimization (PGO) step [51] to refine the poses collectively. The Extended Sequential HLoc (**Seq HLoc+**) further builds on Seq HLoc by performing a second round of retrieval and localization. This second retrieval is guided by the poses estimated in the first PGO step, and is followed by a second pose graph optimization. The full details are provided in [47].

We evaluate the accuracy of the methods with the **image/trajectory recall** at a threshold of (5°, 1 m) and (3°, 0.5 m). To compute trajectory recall, we sample trajectory chunks of lengths 5, 10, 15, and 20 meters. Each chunk undergoes a keyframe selection process, and localization is performed on the selected keyframes. A trajectory chunk is considered successfully recalled if the query keyframe within the chunk meets the recall threshold.

Figures 2, 3 compare Struct-Loc to HLoc, Seq HLoc and Seq HLoc+. Struct-Loc and all the sequential models significantly outperform the single image baseline HLoc. Struct-Loc outperforms Seq HLoc and Seq HLoc+ in trajectory recall for both recall thresholds, especially for the longer trajectories. For the $(5°, 1, \text{m})$ recall threshold, Struct-Loc achieves an improvement of 4–5% for trajectory lengths between 10 and 20 meters, while under the stricter $(3°, 0.5, \text{m})$ threshold the improvement ranges from 2–9% in the same trajectory interval.

Runtime performance To enable near real-time inference, we cache the pre-computed coarse fine and coarse superpoints of the global map. Since the global map remains static, it can be treated as a constant and preprocessed offline. During inference, only the local maps are passed through the encoder, decoder layers. Correspondence matching is then performed using the cached global features and the locally extracted fine points and superpoints. Localization begins by matching the query device's local descriptors to the cached coarse features of the global map. Once coarse correspondences are established, only the fine-level representations corresponding to those matched regions are loaded, enabling detailed matching at progressively finer scales, restricted to the

TABLE I: Runtime breakdown (in seconds) for sequential baselines using retrieval with 10 images.

Model	Traj. (m)	Trajectory		Feature Extraction		Image Retrieval		Feature Matching		Pose Estimation		Pose Graph Optimization	Total	
		Avg. #Img	Avg. Chunk (s)	GPU 1/img	GPU 1/chunk	GPU 1/10img	GPU 1/chunk	GPU 1/10img	GPU 1/chunk	CPU 1/img	CPU 1/chunk	CPU 1/chunk	GPU only	GPU+ CPU
Seq. HLoc	5	8.6 ± 2.6	7.6 ± 2.8	0.008	0.067	0.066	0.564	0.352	3.024	0.506	4.350	0.072	0.425	1.002
	10	16.1 ± 3.8	15.3 ± 4.2	0.008	0.125	0.066	1.055	0.352	5.662	0.506	8.144	0.127	0.425	1.058
	15	23.4 ± 4.4	22.5 ± 5.0	0.008	0.182	0.066	1.533	0.352	8.229	0.506	11.837	0.197	0.425	1.128
	20	31.1 ± 4.8	30.2 ± 5.4	0.008	0.241	0.066	2.037	0.352	10.929	0.506	15.722	0.277	0.425	1.208
Seq. HLoc+	5	8.6 ± 2.6	7.6 ± 2.8	0.008	0.067	0.089	0.761	0.550	4.726	0.506	4.350	0.128	1.152	1.280
	10	16.1 ± 3.8	15.3 ± 4.2	0.008	0.125	0.089	1.425	0.550	8.847	0.506	8.144	0.220	1.152	1.372
	15	23.4 ± 4.4	22.5 ± 5.0	0.008	0.182	0.089	2.072	0.550	12.859	0.506	11.837	0.327	1.152	1.478
	20	31.1 ± 4.8	30.2 ± 5.4	0.008	0.241	0.089	2.752	0.550	17.079	0.506	15.722	0.434	1.152	1.585

TABLE II: Runtime breakdown (in seconds) of Struct-Loc with and without fusion. All times are measured on GPU.

Model	Trajectory (m)	Avg. #Img	Avg. Chunk (s)	Feature Extraction + Fusion	Fine + Coarse Matching	Pose Estimation	Total
Struct-Loc	5	8.6 ± 2.6	7.6 ± 2.8	0.0274	0.0625	0.0075	0.0974
	10	16.1 ± 3.8	15.3 ± 4.2	0.0261	0.0659	0.0077	0.0996
	15	23.4 ± 4.4	22.5 ± 5.0	0.0263	0.0697	0.0082	0.1042
	20	31.1 ± 4.8	30.2 ± 5.4	0.0256	0.0732	0.0082	0.1070

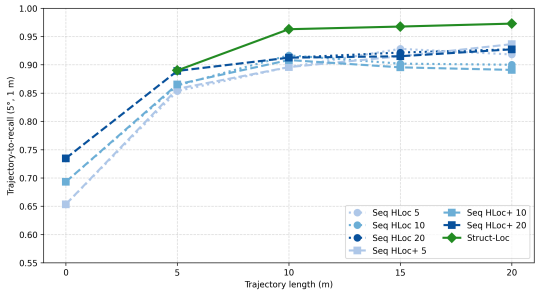

Fig. 2: Trajectory recall comparison (5° / 1 m acceptance threshold)

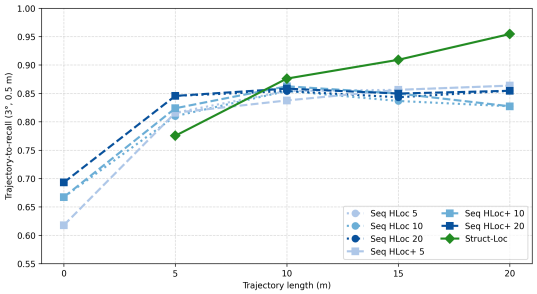

Fig. 3: Trajectory recall comparison (3° / 0.5 m acceptance threshold)

identified local patches. This hierarchical strategy significantly reduces memory usage, as only the relevant portions of the global map are accessed during alignment. It also avoids the need to load or process the entire global point cloud.

In order to measure the runtime performance we used a RTX 3090 NVidia GPU and a AMD EPYC 7413 24-Core CPU. In Tab. 1 the runtime breakdown of the sequential baseline models is given. We report two different total times. We report the GPU only time, that measures the runtime of the deep learning models used for feature extraction, image retrieval and feature matching. The total time is measured when the batch of images encountered in the image processing step fit in the GPU memory and can be processed concurrently, and there are enough CPU resources to process concurrently the pose estimation and pose graph optimization steps. We used the [47] codebase to perform these measurements. The Struct-Loc runtime performance is shown in Tab. 2. When comparing Struct-Loc to the total GPU runtime of Seq HLoc, we observe approximately a $4\times$ reduction in runtime. Compared to total runtime, Struct-Loc achieves a $10\times$ speedup. When compared to the more accurate Seq HLoc+ model, Struct-Loc achieves an approximately $10\times$ reduction in runtime. Relative to the total runtime, the improvement ranges between $11 - 13\times$, highlighting the efficiency of Struct-Loc.

Map compression Figure 4 shows the point cloud of the CAB scene from the LaMAR dataset, along with its corresponding fine and coarse feature representations. High-dimensional feature vectors are projected into a lower-dimensional space using t-SNE [52] to visualize the fused coarse and decoded fine features. The original global map contains approximately 8 million points, which is reduced to 86,662 fine points (each with a 256-dimensional descriptor) and 828 coarse superpoints (each with a 2048-dimensional descriptor). Including the 3D positions of these points, the total storage required for the compressed map in half precision is: $(828 \times (2048 + 3) + 86,662 \times (256 + 3)) \times 2$ bytes $\approx$ 46 MB. In comparison, the COLMAP [40] map consists of 7,728 keyframes, each with an average of 1,299 keypoints and a descriptor length of 256. Using half-precision, the total storage is approximately: $7,728 \times 1,299 \times (256 + 3) \times 2 \approx$ 4.85 GB. This results in a $100\times$ reduction in storage for the global map compared to the COLMAP representation, while still preserving the necessary structure for accurate localization.

V. Conclusion

In this work, we introduced Struct-Loc, a structural localization framework that formulates pose estimation as a hierarchical point cloud registration problem. By leveraging motion continuity and geometric consistency over time, Struct-Loc is robust to transient occlusions and appearance changes, enabling it to resolve ambiguities that single-image methods often cannot. Our main contribution is confidence-aware

Fig. 4: Visualization of a the CAB scene, together with the computed coarse confidence scores, fine and coarse extracted feature representation visualized in a lower-dimensional space using t-SNE.

hierarchical matching: each coarse superpoint is assigned a regressed confidence that propagates into fine-level correspondences and weighted pose estimation, optimized end-to-end with an $SE(3)$ geodesic loss. Struct-Loc achieves robust, accurate, and efficient localization in large-scale environments, outperforming strong baselines in both accuracy and runtime. Furthermore, it enables a $100\times$ compression of the global map compared to COLMAP, making the approach highly scalable for deployment in industrial and commercial indoor environments.

APPENDIX

Training details. Struct-Loc is implemented and evaluated using PyTorch. Training is conducted on a compute cluster equipped with 8 × NVIDIA L40S and 4 × NVIDIA H100 GPUs; inference and performance evaluation are performed on a single NVIDIA RTX 3090 GPU.

The model is trained in three stages with progressively increasing global map context. In the *first stage*, a 60×60 m crop of the global map is extracted for each local map. This crop is defined by computing the bounding box of the local structural points, taking its center, transforming it into the global frame using the ground-truth pose, and then extracting the corresponding region from the global map. The crop is then extracted from the global map accordingly. In the *second stage*, the spatial context is expanded by cropping a 100×100 m region centered at the local bounding box center. In the *third stage*, the full global map is used, allowing the model to leverage complete environmental context. The first and second stage is trained on the L40S GPUs, however, in order to train on the full map the H100 GPUs are required, because of the increased memory requirements.

Local structural maps are precomputed offline from each training session and segmented into overlapping trajectory chunks of lengths 5, 10, 15, and 20 meters; each chunk overlaps with its neighbor by one-third of the trajectory length.

A diverse set of data augmentations is applied during training to both input local an global maps to improve generalization and robustness. Gaussian noise $\mathcal{N}(0, 0.05)$ m is independently added to the coordinates of each structural point; point clouds are randomly scaled using a factor sampled uniformly from the range $[0.8, 1.2]$; and the cropping center point is randomly shifted by a vector drawn from $\mathcal{N}(0, 5)$ meters. Each structural point is independently dropped with a probability of 0.2. To improve invariance to orientation, a random 3D rotation is applied consistently to the local maps and the corresponding ground-truth transformation. The rotation angles are sampled independently from uniform distributions: yaw $\theta_z \sim \mathcal{U}(-180°, 180°)$; pitch $\theta_y \sim \mathcal{U}(-30°, 30°)$; and roll $\theta_x \sim \mathcal{U}(-10°, 10°)$.

The network is trained in the three stages for 100, 100, and 150 epochs, respectively. A batch size of 1 is used per GPU throughout all stages. Each stage employs the same optimizer configuration. The model is optimized using the Adam optimizer [53], with an initial learning rate of 1×10^{-4}. A learning rate decay factor of 0.95 is applied every 4 epochs to progressively reduce the learning rate during training. Additionally, a weight decay of 1×10^{-6} is applied to regularize the model and mitigate overfitting.

REFERENCES

[1] H. Huang, C. Liu, Y. Zhu, C. Hui, T. Braud, and S.-K. Yeung, "360loc: A dataset and benchmark for omnidirectional visual localization with cross-device queries," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[2] P. Karkus, D. Hsu, and W. S. Lee, "Particle filter networks with application to visual localization," in *Conference on Robot Learning*, 2018.

[3] S. Thrun, "Particle filters in robotics," in *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, 2002, p. 511–518.

[4] G. Elbaz, T. Avraham, and A. Fischer, "3d point cloud registration for localization using a deep neural network auto-encoder," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2472–2481.

[5] K. Wang, J. Li, Z. Chen, and J. Wang, "3d point cloud registration method based on structural matching of feature points," in *Methods and Applications for Modeling and Simulation of Complex Systems*, 12 2022, pp. 595–608.

[6] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[7] J. Revaud, J. Almazan, R. Rezende, and C. D. Souza, "Learning with average precision: Training image retrieval with a listwise loss," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 5106–5115.

[8] P.-E. Sarlin, A. Unagar, M. Larsson, H. Germain, C. Toft, V. Larsson, M. Pollefeys, V. Lepetit, L. Hammarstrand, F. Kahl, and T. Sattler, "Back to the feature: Learning robust camera localization from pixels to pose," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3246–3256, 2021.

[9] P.-E. Sarlin, C. Cadena, R. Y. Siegwart, and M. Dymczyk, "From coarse to fine: Robust hierarchical localization at large scale," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12 708–12 717, 2018.

[10] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research. PMLR, Jul 2020, pp. 1597–1607.

[11] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2, 2006, pp. 1735–1742.

[12] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International Conference on Computer Vision*, 2011, pp. 2564–2571.

[13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[14] C. G. Harris and M. J. Stephens, "A combined corner and edge detector," in *Alvey Vision Conference*, 1988.

[15] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018.

[16] J. Revaud, P. Weinzaepfel, C. D. Souza, and M. Humenberger, "R2d2: repeatable and reliable detector and descriptor," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019.

[17] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torii, and T. Sattler, "D2-Net: A Trainable CNN for Joint Detection and Description of Local Features," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8084–8093.

[18] M. J. Tyszkiewicz, P. Fua, and E. Trulls, "Disk: learning local features with policy gradient," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020.

[19] Y. Tian, X. Yu, B. Fan, F. Wu, H. Heijnen, and V. Balntas, " SOSNet: Second Order Similarity Regularization for Local Descriptor Learning ," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019, pp. 11 008–11 017.

[20] R. Pautrat, V. Larsson, M. R. Oswald, and M. Pollefeys, "Online invariance selection for local feature descriptors," in *European Conference on Computer Vision*, 2020, pp. 707–724.

[21] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superglue: Learning feature matching with graph neural networks," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 4937–4946.

[22] P. Lindenberger, P.-E. Sarlin, and M. Pollefeys, "Lightglue: Local feature matching at light speed," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 17 581–17 592.

[23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6000–6010.

[24] G. Peyré and M. Cuturi, "Computational optimal transport: With applications to data science," *Foundations and Trends in Machine Learning*, vol. 11, no. 5-6, pp. 355–607, 2019.

[25] J. Sun, Z. Shen, Y. Wang, H. Bao, and X. Zhou, "Loftr: Detector-free local feature matching with transformers," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 8918–8927.

[26] Q. Wang, J. Zhang, K. Yang, K. Peng, and R. Stiefelhagen, "Matchformer: Interleaving attention in transformers for feature matching," in *Computer Vision – ACCV 2022: 16th Asian Conference on Computer Vision, Macao, China, December 4–8, 2022, Proceedings, Part III*. Berlin, Heidelberg: Springer-Verlag, 2022, p. 256–273.

[27] M. J. Milford and G. F. Wyeth, "Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 1643–1649.

[28] S. Garg and M. Milford, "Seqnet: Learning descriptors for sequence-based hierarchical place recognition," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4305–4312, 2021.

[29] C. Choy, W. Dong, and V. Koltun, "Deep global registration," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 2511–2520.

[30] "Point tree transformer for point cloud registration," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 35, no. 7, pp. 6756–6772, 2025.

[31] Q. Zhou, S. Agostinho, A. Ošep, and L. Leal-Taixé, "Is geometry enough fornbsp;matching innbsp;visual localization?" in *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part X*, 2022, p. 407–425.

[32] J. Edstedt, A. Mateus, and A. Jaenal, "Colabsfm: Collaborative structure-from-motion by point cloud registration," in *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025, pp. 6573–6583.

[33] H. Yu, Z. Qin, J. Hou, M. Saleh, D. Li, B. Busam, and S. Ilic, "Rotation-invariant transformer for point cloud matching," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 5384–5393.

[34] H. Yu, F. Li, M. Saleh, B. Busam, and S. Ilic, "Cofinet: reliable coarse-to-fine correspondences for robust point cloud registration," in *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021.

[35] Z. Qin, H. Yu, C. Wang, Y. Guo, Y. Peng, and K. Xu, "Geometric transformer for fast and robust point cloud registration," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 11 133–11 142.

[36] S. Huang, Z. Gojcic, M. Usvyatsov, A. Wieser, and K. Schindler, "Predator: Registration of 3d point clouds with low overlap," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 4265–4274.

[37] F. Lu, G. Chen, Y. Liu, L. Zhang, S. Qu, S. Liu, R. Gu, and C. Jiang, "Hregnet: A hierarchical network for efficient and accurate outdoor lidar point cloud registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 10, pp. 11 884–11 897, 2023.

[38] J. Liu, G. Wang, Z. Liu, C. Jiang, M. Pollefeys, and H. Wang, "Regformer: An efficient projection-aware transformer network for large-scale point cloud registration," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 8417–8426.

[39] G. Chen, M. Wang, Y. Yang, L. Yuan, and Y. Yue, "Fast and robust point cloud registration with tree-based transformer," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 773–780.

[40] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4104–4113.

[41] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas, " KPConv: Flexible and Deformable Convolution for Point Clouds ," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Nov. 2019, pp. 6410–6419.

[42] H. Thomas, Y.-H. H. Tsai, T. D. Barfoot, and J. Zhang, " KPConvX: Modernizing Kernel Point Convolution with Kernel Attention ," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024, pp. 5525–5535.

[43] I. Rocco, M. Cimpoi, R. Arandjelović, A. Torii, T. Pajdla, and J. Sivic, "Neighbourhood consensus networks," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018, p. 1658–1669.

[44] M. Cuturi, "Sinkhorn distances: lightspeed computation of optimal transport," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, 2013, p. 2292–2300.

[45] J. C. Gower, "Generalized procrustes analysis," *Psychometrika*, vol. 40, no. 1, pp. 33–51, Mar 1975.

[46] Y. Sun, C. Cheng, Y. Zhang, C. Zhang, L. Zheng, Z. Wang, and Y. Wei, "Circle loss: A unified perspective of pair similarity optimization," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 6397–6406.

[47] P.-E. Sarlin, M. Dusmanu, J. L. Schönberger, P. Speciale, L. Gruber, V. Larsson, O. Miksik, and M. Pollefeys, "Lamar: Benchmarking localization and mapping for augmented reality," in *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VII*, 2022, p. 686–704.

[48] K. Khoshelham and S. O. Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors (Basel, Switzerland)*, vol. 12, pp. 1437 – 1454, 2012.

[49] M. Humenberger, Y. Cabon, N. Guerin, J. Morat, V. Leroy, J. Revaud, P. Rerole, N. Pion, C. de Souza, and G. Csurka, "Robust image retrieval-based visual localization using kapture," 2022.

[50] N. Pion, M. Humenberger, G. Csurka, Y. Cabon, and T. Sattler, "Benchmarking image retrieval for visual localization," in *2020 International Conference on 3D Vision (3DV)*, 2020, pp. 483–494.

[51] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based slam," *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010.

[52] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008.

[53] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, vol. abs/1412.6980, 2014.