Advances in Artificial Intelligence and Machine Learning; Research 5 (4) 4645-4674 Received 30-08-2025; Accepted 28-11-2025; Published online 05-12-2025

A Graph-Attention Frontend for ORB-SLAM2 under Large Viewpoint Changes

Giuseppe Macario gm@mit.edu.it
Universitas Mercatorum
Piazza Mattei, 10
00186 Rome, Italy

Corresponding Author: Giuseppe Macario

Abstract

This paper presents a visual SLAM system that augments the ORB-SLAM2 frontend with learned detection and matching while preserving the standard backend. A fully convolutional feature extractor with a position–prior head predicts one keypoint per grid cell with sub-cell refinement, producing uniformly distributed and repeatable features under illumination changes and low texture. A graph-attention matcher jointly reasons over intra-image neighborhoods and cross-image relations; a Sinkhorn-based assignment with a dustbin yields calibrated, approximately one-to-one correspondences. A lightweight optimization layer prunes and reweights candidates before geometric verification, and a dynamic frontend policy activates the learned components only when viewpoint change or inlier support indicate difficult frames, otherwise falling back to fast ORB tracking. The backend, i.e. local mapping, loop closing (DBoW2), and bundle/pose-graph optimization, remains unchanged, with learned descriptors used for 3D–2D association. Experiments on HPatches, TUM RGB-D, KITTI odometry, and real-world robot runs show improved detector repeatability and matching mAP, as well as reduced trajectory error compared to ORB-SLAM2, DX-SLAM, and GCNv2-SLAM, while maintaining real-time throughput on modest hardware. These results indicate that attention-based matching and position-aware detection provide a practical path to robust SLAM under large viewpoint and appearance changes without sacrificing efficiency.

Keywords: Simultaneous Localization and Mapping (SLAM), Visual SLAM, Feature extraction, Keypoint detection, Feature matching, Graph neural networks, Attention mechanisms, Optimal transport, Pose estimation, Real-time systems.

1. INTRODUCTION

Simultaneous Localization and Mapping (SLAM) is the process by which a robot moving in an unknown environment estimates its own motion while incrementally building a map of the scene [1, 2].

Citation: Giuseppe Macario. A Graph-Attention Frontend for ORB-SLAM2 under Large Viewpoint Changes. **Advances in Artificial Intelligence and Machine Learning**. 2025;5(4):258. https://dx.doi.org/10.54364/AAIML.2025.54258

Classical visual SLAM systems rely on handcrafted local features. Keypoints are detected and described in each frame, matched across views, and used to recover camera motion through geometric estimation; loop-closure detection provides global drift correction. Bundle adjustment and pose-graph optimization then refine structure and trajectory [3].

Among these systems, ORB-SLAM2 (Oriented FAST and Rotated BRIEF) [4], is a widely adopted baseline. It uses ORB keypoints and binary descriptors [5], for fast, rotation- and scale-robust matching (Hamming-distance NN, typically accelerated and guided via DBoW2-inverted file lists and epipolar/geometric constraints), organizes the pipeline into three parallel threads (tracking, local mapping, and loop closing) and performs geometric estimation with RANSAC (essential/fundamental matrix or PnP when 3D points are available), followed by local and global optimization. Place recognition for loop closure is handled by a bag-of-words engine (DBoW2) [6].

Despite their efficiency, handcrafted features degrade under severe illumination change, motion blur, repetitive patterns, and large viewpoint changes, which reduce match consistency and can trigger tracking loss.

Deep learning has improved the robustness of visual features. Convolutional neural networks (CNNs) have been used to produce local detectors and descriptors (for example, SuperPoint-like approaches, R2D2, DISK, LF-Net, Key.Net, ALIKE) [7–12], and global image descriptors for place recognition (for example, DenseVLAD, NetVLAD, or AP-GeM-style embeddings) [13].

Recent SLAM variants integrate learned features into the ORB-SLAM2 framework. DX-SLAM employs CNN-based local features together with global descriptors for loop closure [14]. GCNv2 provides a learned binary local descriptor and detector implemented with CNNs (despite the name, it is not a graph-convolutional network), improving efficiency but still struggling in texture-poor or highly oblique views [15].

Graph neural networks (GNNs) have emerged as an effective tool for feature matching: instead of processing descriptors independently, they reason over keypoint neighborhoods and cross-image relations. SuperGlue, [16–19], for example, performs attention-based message passing over keypoint graphs and formulates assignment with a learnable dustbin for unmatched points, yielding strong 2D–2D correspondences under viewpoint and illumination changes. Nevertheless, performance still depends on the quality and spatial coverage of detected keypoints, which can be a bottleneck in low-texture or low-light scenes.

To address these limitations, this work proposes a visual SLAM system that augments the frontend of ORB-SLAM2 with two learned components: a keypoint detector and descriptor with a position-prior head to encourage uniform, repeatable coverage, and a graph-attention matcher that aggregates intra- and inter-frame context to produce reliable correspondences. The standard ORB-SLAM2 backend (local mapping, loop closing, and optimization) is retained. This integration aims to deliver accurate localization and mapping under large viewpoint changes and challenging illumination while preserving real-time performance.

2. RELATED WORK

Research on visual SLAM spans classical pipelines built on handcrafted local features and recent approaches that replace or augment the frontend with learned components while preserving mature geometry and optimization backends. ORB-SLAM2 popularized a modular architecture based on ORB keypoints and descriptors [5], Hamming-distance matching, RANSAC for essential or PnP estimation, loop detection with DBoW2 [6], and graph-based optimization with g2o [3], and it remains a widely used baseline [4]. ORB-SLAM3 extended the family to visual–inertial and multi-map settings without altering these core principles [20]. Robust minimal solvers such as the five-point algorithm for relative pose [21], and EP*n*P for absolute pose [22], together with RANSAC variants like PROSAC [23], and learned guidance [24], underpin geometric verification in these systems.

Learned local features have improved detection repeatability and descriptor distinctiveness under appearance changes. Self-supervised and weakly supervised detectors and descriptors include SuperPoint [7], R2D2 [8], DISK [9], LF-Net [10], Key.Net [11], and ALIKE [12]. These methods typically predict interest points and descriptors with convolutional encoders and train with homography-based or multi-view consistency. Despite strong results, performance can degrade in texture-poor regions or under extreme viewpoint changes when spatial coverage or descriptor context is limited.

Matching with joint context reasoning has been advanced by graph and transformer models. SuperGlue performs message passing with attention over keypoint graphs and computes correspondences with a Sinkhorn-normalized assignment that includes a dustbin for unmatched points [16, 25]. LightGlue improves efficiency with dynamic early exiting while retaining attention-based reasoning [17]. Detector-free or dense alternatives such as LoFTR [18], and MatchFormer [19], aggregate context across scales to increase recall under strong viewpoint and illumination changes. Practical systems often couple attention layers with approximate nearest-neighbor retrieval to constrain candidate pairs, for example using FAISS [26].

Several SLAM variants integrate learned features into ORB-SLAM–style frameworks. DX-SLAM replaces ORB with CNN-based local features inside a standard tracking, mapping, and loop-closure pipeline [14]. GCNv2-SLAM deploys a compact CNN detector and binary descriptor to retain speed with improved robustness relative to handcrafted features [15]. CNN-SLAM combines monocular SLAM with learned dense depth to enable real-time dense mapping [13]. These systems demonstrate that learned frontends can be inserted into classical backends with gains in challenging scenarios. However, when matching relies on independent nearest-neighbor decisions, ambiguities from repetitive patterns or large baselines can still reduce inlier support and trigger track loss.

The present study relates to these lines of work by coupling a position-aware, fully convolutional detector–descriptor with a graph-attention matcher that uses Sinkhorn-based assignment and an explicit dustbin. The design follows the trend of retaining a standard ORB-SLAM2 backend [3, 4, 6], while improving the frontend with learned components [7–12, 16–19]. In contrast to pipelines that depend solely on descriptor similarity, attention-based message passing and optimal-transport normalization provide calibrated, approximately one-to-one correspondences that increase robustness under large viewpoint and appearance changes, and that integrate naturally with robust geometric verification [21–24].

3. METHOD

The system comprises three main components: a learned feature extraction module (detector and descriptor) with a position–prior head that encourages uniform, repeatable keypoint coverage, a graph-attention matcher that reasons jointly over intra-image neighborhoods and cross-image relations, and the standard ORB-SLAM2 backend (tracking, local mapping, loop closing and optimization) for map maintenance and trajectory refinement.

In the feature extraction module, a lightweight position-prior head is added on top of a fully convolutional encoder. The network is trained in a self-supervised manner using multi-homography augmentation over large-scale image datasets, so that detection and description remain stable across viewpoint and illumination changes. The head predicts sub-cell offsets and confidences, which are used to select the top-v keypoints while promoting spatial coverage.

During operation, many frame pairs exhibit small inter-frame baselines where classical ORB features already track reliably and at low cost. To preserve real-time performance, a dynamic frontend policy is used: a fast ORB track is attempted first (Hamming-distance nearest-neighbor matching followed by RANSAC on the essential matrix or PnP when 3D points exist). If the inlier ratio or count drops below a threshold, the estimated median parallax or rotation indicates a large viewpoint change (for example, $\mathrm{parallax} > p_{\mathrm{hi}}$ or $\mathrm{rotation} > \theta_{\mathrm{hi}}$), or the reprojection error spikes, the system switches to the learned detector and descriptor and invokes the graph-attention matcher. Conversely, when the median parallax is small and inlier support is stable, the fast ORB track is maintained. This gating ties the use of the learned components to genuinely hard viewpoints rather than to raw match counts alone.

For matching, a bipartite keypoint graph is constructed with self-attention edges within each image to aggregate local context and suppress ambiguous points, and cross-attention edges between images to exchange information across tentative correspondences. A multi-layer message-passing GNN produces match scores that are converted into a doubly stochastic assignment with an additional dustbin (unmatched) class via Sinkhorn normalization. The output is a set of geometrically consistent correspondences with calibrated confidences.

Global trajectory and mapping follow the ORB-SLAM2 backend: new keyframes are inserted, redundant ones are culled, and local bundle adjustment minimizes reprojection error. Loop closure is detected via place recognition. We retain DBoW2 for loop closure and still extract ORB on keyframes (only on keyframes, not every frame) to index them in the vocabulary. Each keyframe and map point also stores a learned descriptor; map points keep a learned descriptor (e.g., the median over observations) used for 3D $\leftrightarrow$ 2D association and for PnP/BA during tracking and local mapping, while ORB descriptors are used solely for DBoW2 indexing and as a fallback. This keeps BoW costs bounded while making 3D $\leftrightarrow$ 2D matching consistent with the learned frontend. Pose-graph optimization and a final BA pass yield the global trajectory and map. The overall framework is illustrated in FIGURE 1.

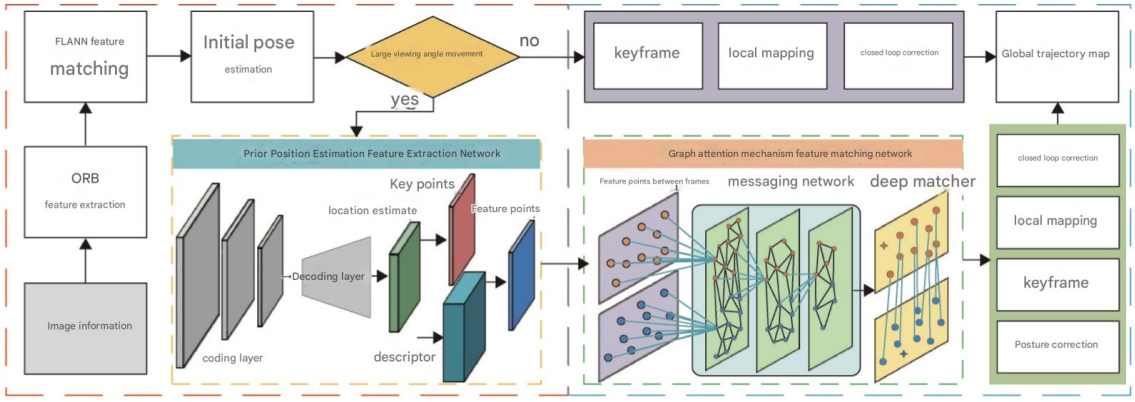

Figure 1: Framework of the proposed algorithm.

4. POSITION-PRIOR FEATURE EXTRACTOR

Classical feature extractors rely on handcrafted detectors and binary or floating-point descriptors; keypoints are then paired with descriptors to form local features. While efficient, the number and spatial coverage of such keypoints can vary strongly with illumination, blur, and viewpoint, which degrades match consistency and downstream geometry.

This work addresses the problem with a fully convolutional, self-supervised feature extractor equipped with a position–prior head. The module comprises a shared encoder and three lightweight prediction heads. The first head is a position module (two 1×1 convolutions, 256→2) that outputs raw offsets $\hat{\delta}_{uv} \in \mathbb{R}^2$; these offsets are mapped inside the cell with a sigmoid $\delta_{uv} = \sigma(\hat{\delta}_{uv}) \in (0,1)^2$ and clamped to $[\varepsilon, 1-\varepsilon]$ with $\varepsilon \approx 10^{-3}$. The corresponding image coordinates are $\ell_{uv} = s\big(u + \delta^x_{uv},\ v + \delta^y_{uv}\big)$. The second head is a confidence module that scores candidate locations. The third head is a descriptor module that produces ℓ_2-normalized descriptors. The overall structure is shown in FIGURE 2.

The backbone processes an input image of size $H \times W$ with a compact encoder (VGG/ResNet-lite blocks with stride) to produce a feature map $F \in \mathbb{R}^{\frac{H}{s}\times\frac{W}{s}\times C}$ at output stride $s = 8$. Strided convolutions are used instead of repeated max pooling to retain localization fidelity and reduce aliasing, and ReLU activations follow all intermediate convolutions.

For each spatial cell (u, v) on F, the position head outputs raw offsets $\hat{\delta}_{uv}$ that are squashed with a sigmoid into $\delta_{uv} = \sigma(\hat{\delta}_{uv}) \in (0,1)^2$ (optionally clamped), and the confidence head outputs a scalar score $c_{uv} \in [0,1]$. The corresponding image coordinates are

$$\ell_{uv} = (x_{uv}, y_{uv}) = s\left(u + \delta^x_{uv}, v + \delta^y_{uv}\right),$$

which map each cell to a precise pixel location without resorting to an argmax over heatmaps. This formulation encourages uniform coverage (one candidate per cell) while allowing subpixel refinement.

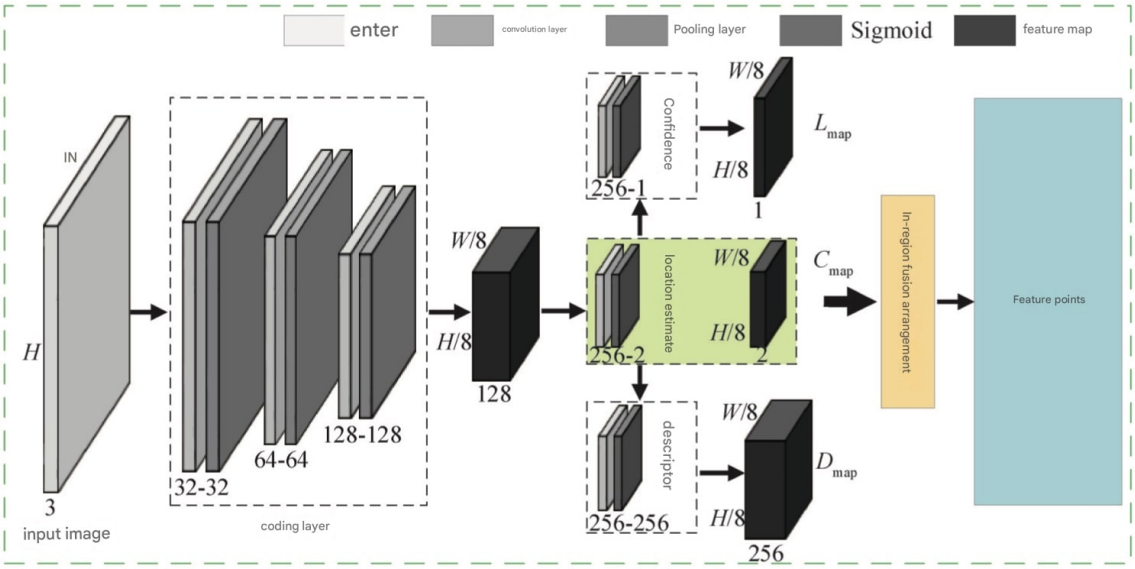

Figure 2: Structure of the Prior Position Estimation Feature Extraction Network.

The descriptor head produces an embedding $e_{uv} \in \mathbb{R}^d$ at each cell, optionally refined via bilinear sampling at ℓ_{uv} on an intermediate feature tensor to reduce quantization. Descriptors are ℓ_2-normalized, enabling cosine (dot product) similarity and stable gradient flow.

The three heads operate on the same backbone features and share early layers to avoid redundant computation. Denote the collections

$$L_{\text{map}} = \{\ell_{uv}\}, \quad C_{\text{map}} = \{c_{uv}\}, \quad D_{\text{map}} = \{e_{uv}\},$$

each defined over the $\mu = \frac{H}{s} \times \frac{W}{s}$ grid cells. Candidates are ranked by c_{uv} and a lightweight non-maximum suppression in image coordinates is applied to avoid local clustering; the top-v survivors yield the position matrix $L \in \mathbb{R}^{v\times 2}$, confidence vector $C \in \mathbb{R}^v$, and descriptor matrix $D \in \mathbb{R}^{v\times d}$ used by the matcher.

Predicting one candidate per cell with subcell refinement replaces expensive dense keypoint search with a constant-cost pass over the feature grid. The position–prior head stabilizes the number and spatial distribution of keypoints across frames, mitigating the effect of illumination and texture sparsity on downstream matching. During inference, this also enables restricting candidate gathering and descriptor sampling to local neighborhoods, lowering memory traffic and improving throughput.

The extractor is trained in a self-supervised manner with multi-homography augmentation: pairs of views are synthesized by random planar transforms, and consistency between (L, D, C) across the two views provides supervisory signals for detection (repeatability and coverage via confidence ranking) and description (contrastive objectives over correspondences). Detailed losses are introduced in the following section. All components are differentiable end to end through the offset-based coordinate mapping. During training, detection losses are applied densely on per-cell outputs; for

descriptor contrastive terms we sample top-v candidates with a stop-gradient (non-differentiable) selection. At inference, we perform top-v selection followed by NMS.

4.1 Model

The encoder consists of six 3×3 convolutions with channel configuration $[32, 32, 64, 64, 128, 128]$. Downsampling to an output stride $s = 8$ is achieved with three stride-2 operations implemented as strided convolutions placed after layers 2, 4, and 6. Each convolution is followed by ReLU; no max pooling is used to preserve localization fidelity. Given an input of size $H \times W$, the encoder outputs a feature tensor $F \in \mathbb{R}^{\frac{H}{8} \times \frac{W}{8} \times 128}$.

On top of F the three lightweight heads share an initial 1×1 expansion layer that maps $128 \to 256$ with ReLU. The position head then applies a 1×1 convolution $256 \to 2$ and outputs raw offsets $\hat{\delta}_{uv} \in \mathbb{R}^2$ that are squashed to $\delta_{uv} = \sigma(\hat{\delta}_{uv}) \in (0, 1)^2$ and optionally clamped to $[\varepsilon, 1 - \varepsilon]$. The confidence head applies a 1×1 convolution $256 \to 1$ with a sigmoid and outputs $c_{uv} \in [0, 1]$. The descriptor head applies a 1×1 convolution $256 \to d$ and produces an embedding $e_{uv} \in \mathbb{R}^d$, later ℓ_2-normalized. Here (u, v) indexes a spatial cell on the $\frac{H}{8} \times \frac{W}{8}$ grid. The corresponding image coordinates are computed as

$$\ell_{uv} = (x_{uv}, y_{uv}) = s\left(u + \delta^x_{uv}, v + \delta^y_{uv}\right),$$

which map each cell to a precise pixel location in a differentiable manner. When finer localization is required, descriptors are bilinearly sampled at ℓ_{uv} from an intermediate descriptor map before normalization.

The following sets are defined:

$$L_{\text{map}} = \{\ell_{uv}\}, \quad C_{\text{map}} = \{c_{uv}\}, \quad D_{\text{map}} = \{e_{uv}\},$$

each with $\mu = \frac{H}{8} \times \frac{W}{8}$ elements, one per cell. At inference time candidates are ranked by c_{uv} and a non-maximum suppression in image space with radius r avoids local clustering. The top-v survivors form the position matrix $L \in \mathbb{R}^{v \times 2}$, the confidence vector $C \in \mathbb{R}^v$, and the descriptor matrix $D \in \mathbb{R}^{v \times d}$ that are passed to the matcher.

For subsequent graph-based matching the visual descriptors are enriched with geometric context. Let $\tilde{\ell}_i = (x_i/W, y_i/H)$ be normalized coordinates and let $\phi(\tilde{\ell}_i, c_i)$ be a small multilayer perceptron embedding of $(\tilde{\ell}_i, c_i)$. The initial node feature used by the GNN is

$$h_i^{(0)} = W_p\left[e_i \,\|\, \phi(\tilde{\ell}_i, c_i)\right] + b_p,$$

where $[\cdot\|\cdot]$ denotes concatenation and W_p, b_p are learnable parameters. This preserves the discriminative power of e_i while injecting spatial priors and confidence into the representation.

Predicting one candidate per grid cell with subcell refinement yields approximately uniform spatial coverage and a stable candidate count across frames, properties that benefit repeatability under illumination change and texture sparsity. The fully convolutional design shares computation among the three heads and avoids costly dense keypoint searches. Inference reduces to a single forward pass over F, followed by ranking and non-maximum suppression. During training, losses are applied on dense per-cell outputs before top-v selection or non-maximum suppression, since the selection itself is non-differentiable.

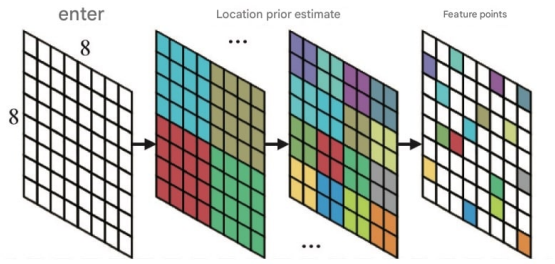

Figure 3: Module for estimating sub-cell keypoint positions and confidences. The position head predicts offsets within each $s \times s$ cell; coordinates are then mapped to pixels as $s(u+\delta^x, v+\delta^y)$.

4.2 Matcher Loss

Given two images G_A and G_B, the extractor provides keypoint coordinates $L^A = \{\ell_i^A\}_{i=1}^M$, $L^B = \{\ell_j^B\}_{j=1}^N$, confidences C^A, C^B, and descriptors $D^A = \{e_i^A\}$, $D^B = \{e_j^B\}$. After K message-passing layers, the matcher outputs pairwise logits $Z \in \mathbb{R}^{M \times N}$. This matrix is augmented with a dustbin row and column to model unmatched keypoints. Per-keypoint logits $z_{i,\emptyset}$ and $z_{\emptyset,j}$ yield the extended matrix $\bar{Z} \in \mathbb{R}^{(M+1)\times(N+1)}$ with indices $i \in \{1,\dots,M\} \cup \{\emptyset\}$ and $j \in \{1,\dots,N\} \cup \{\emptyset\}$, and with $\bar{Z}_{\emptyset,\emptyset} = 0$.

An optimal-transport normalization with explicit marginals and a dustbin is then applied. Let $a \in \mathbb{R}^{M+1}$ with $a_i = \frac{1}{M}$ for $i \le M$ and $a_\emptyset = 1$, and let $b \in \mathbb{R}^{N+1}$ with $b_j = \frac{1}{N}$ for $j \le N$ and $b_\emptyset = 1$. A log-domain Sinkhorn algorithm is run on $\bar{Z}/\tau$ (with $\bar{Z}_{\emptyset,\emptyset} = 0$) to obtain

$$\bar{P} = \text{Sinkhorn}\left(\frac{\bar{Z}}{\tau}; a, b\right) \quad \text{s.t.} \quad \sum_j \bar{P}_{ij} = a_i, \sum_i \bar{P}_{ij} = b_j.$$

The real block for $i \le M$ and $j \le N$ yields match probabilities, while $\bar{P}_{i,\emptyset}$ and $\bar{P}_{\emptyset,j}$ are the unmatched probabilities. The $(\emptyset,\emptyset)$ entry is ignored in the loss; an optional learnable dustbin bias can be applied before Sinkhorn.

During training a pair (G_A, G_B) is synthesized by warping G_A with a random homography T. A keypoint ℓ_i^A is labeled as matched to $j^\star$ if $\|\pi(T\,\tilde{\ell}_i^A) - \ell_{j^\star}^B\|_2 \le \varepsilon_{\text{corr}}$ with nearest-neighbor choice; otherwise it is labeled unmatched. Here $\tilde{\ell}$ denotes homogeneous coordinates and π the dehomogenization. This procedure yields a sparse one-hot target matrix $\bar{Y}$ over the augmented domain.

The matching loss is defined as the negative log-likelihood of the assignment:

$$L_{\text{match}} = -\sum_{\substack{i \le M,\, j \le N \\ \bar{Y}_{ij}=1}} \log \bar{P}_{ij} - \sum_{\substack{i \le M \\ \bar{Y}_{i,\emptyset}=1}} \log \bar{P}_{i,\emptyset} - \sum_{\substack{j \le N \\ \bar{Y}_{\emptyset,j}=1}} \log \bar{P}_{\emptyset,j}.$$

This loss is fully differentiable through attention layers and Sinkhorn and it naturally handles occlusions or missing detections through the dustbin terms.

To bias scores toward geometrically plausible pairs, a robust reprojection error under the training homography is minimized, weighted by the soft assignment:

$$r_{ij} = \|\pi(T\tilde{\ell}_i^A) - \ell_j^B\|_2, \qquad L_{\text{geo}} = \frac{1}{\sum_{ij}\bar{P}_{ij}} \sum_{i\le M}\sum_{j\le N} \bar{P}_{ij}\,\rho(r_{ij}),$$

where $\rho(\cdot)$ is the Huber loss and $\mathcal{H}(\cdot)$ denotes Shannon entropy. This encourages high probability on pairs that agree with the known planar mapping while remaining tolerant to localization noise.

Although the matcher can learn from $L_{\text{match}} + L_{\text{geo}}$, an additional symmetric contrastive term stabilizes early training:

$$L_{\text{desc}} = -\sum_{(i,j)\in\mathcal{P}} \log \frac{\exp(\langle e_i^A, e_j^B\rangle/\tau_d)}{\sum_{j'}\exp(\langle e_i^A, e_{j'}^B\rangle/\tau_d)} - \sum_{(i,j)\in\mathcal{P}} \log \frac{\exp(\langle e_i^A, e_j^B\rangle/\tau_d)}{\sum_{i'}\exp(\langle e_{i'}^A, e_j^B\rangle/\tau_d)},$$

where $\mathcal{P}$ are homography-consistent positives and τ_d is a temperature.

Regularization is introduced by defining $\tilde{P}_{i,:} = \bar{P}_{i,:}/a_i$ and $\tilde{P}_{:,j} = \bar{P}_{:,j}/b_j$ and applying an entropy penalty on the normalized distributions:

$$L_{\text{ent}} = -\frac{1}{M}\sum_{i\le M}\mathcal{H}(\tilde{P}_{i,:}) - \frac{1}{N}\sum_{j\le N}\mathcal{H}(\tilde{P}_{:,j}).$$

This keeps the meaning of entropy invariant under the non-unit marginals a and b. Since L_{total} includes the term $\lambda_{\text{ent}}L_{\text{ent}}$ with $\lambda_{\text{ent}} > 0$, minimizing L_{total} maximizes the entropy of $\bar{P}$ and prevents premature overconfidence. An annealing schedule gradually reduces λ_{ent} to zero, for example $\lambda_{\text{ent}}(t) = \lambda_{\text{ent}}^{(0)}\max\left(0,\, 1 - \frac{t}{T_{\text{stop}}}\right)$ with $T_{\text{stop}} = 50\text{k}$ steps, so that in later stages the model can converge toward sharper assignments.

The full objective for the matcher is

$$L_{\text{total}} = \lambda_{\text{match}}L_{\text{match}} + \lambda_{\text{geo}}L_{\text{geo}} + \lambda_{\text{desc}}L_{\text{desc}} + \lambda_{\text{ent}}L_{\text{ent}}.$$

In contrast to LP-based assignment, the Sinkhorn-based formulation is computationally efficient, fully differentiable end-to-end, and well suited to real-time operation within the present SLAM framework.

5. GRAPH-ATTENTION MATCHER

Conventional nearest-neighbor matching of local descriptors is brittle under large viewpoint, illumination, or texture changes: the overlap between views shrinks, descriptors become ambiguous, and simple ratio or RANSAC filtering may fail, causing track loss. To increase robustness, a graph-attention matcher is used that reasons jointly over keypoints within each image and across the image pair.

Instead of scoring matches independently (as in independent NN pipelines, for example brute-force Hamming or ℓ_2 k-NN with a ratio test), an explicit keypoint graph is built. Nodes encode

visual descriptors together with normalized image coordinates and confidences; edges capture self-attention relations within each image (local neighborhoods) and cross-attention relations between images (candidate correspondences). A stack of attention-based message-passing layers aggregates context, suppresses ambiguous or repetitive structures, and outputs match logits that reflect both appearance and spatial consistency.

From these logits, a soft, approximately one-to-one assignment is obtained by applying Sinkhorn normalization with an additional "dustbin" row and column for unmatched keypoints, yielding a differentiable optimal-transport layer. High-confidence pairs are then validated geometrically with a minimal solver inside RANSAC (essential matrix for 2D–2D; PnP only with monocularly triangulated map points; no depth is used in the RGB-only experiments), and the inlier mask serves as supervision for the attention layers during training.

To maintain real-time performance, cross-attention is restricted to a top-k candidate set per keypoint obtained via approximate nearest-neighbor (ANN) search over descriptors (rather than exhaustive similarity), while self-attention operates within a fixed-radius neighborhood. This reduces quadratic costs while preserving the benefits of global context aggregation. The resulting correspondences improve pose estimation and tracking stability in the presence of large viewpoint changes. The overall architecture is depicted in FIGURE 4.

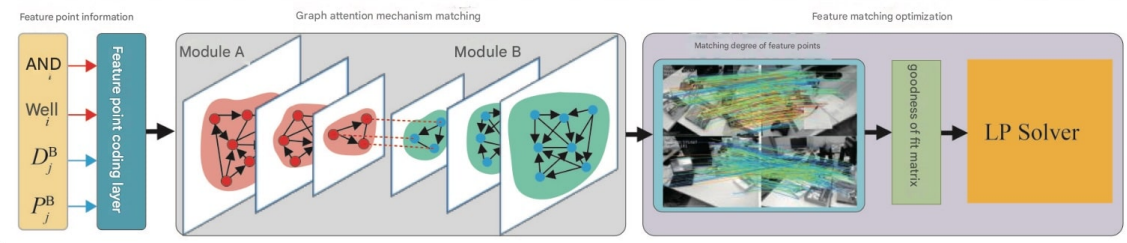

Figure 4: Structure of the graph-attention matcher. Self-attention aggregates intra-image context; cross-attention exchanges information across images; a Sinkhorn layer with a dustbin converts logits to assignments; geometric verification returns the final matches.

Compared to dense, set-wise matching, this approach integrates aggregation, updating, and selection of node information within a single message-passing framework, capturing richer global cues and improving the accuracy and resilience of camera tracking in challenging, large-viewpoint scenarios.

5.1 Model

Let $Q_1 = \{(\ell_i^1, e_i^1, c_i^1)\}_{i=1}^{v_1}$ and $Q_2 = \{(\ell_j^2, e_j^2, c_j^2)\}_{j=1}^{v_2}$ denote the keypoints from the two images, with pixel coordinates $\ell \in \mathbb{R}^2$, ℓ_2-normalized descriptors $e \in \mathbb{R}^d$, and confidences $c \in [0, 1]$. Each node is initialized as

$$h_i^{1(0)} = W_p\left[e_i^1 \| \phi(\tilde{\ell}_i^1, c_i^1)\right] + b_p, \quad h_j^{2(0)} = W_p\left[e_j^2 \| \phi(\tilde{\ell}_j^2, c_j^2)\right] + b_p,$$

where ϕ embeds normalized coordinates and confidence through a small multilayer perceptron and $[\cdot\|\cdot]$ denotes concatenation.

We construct two directed neighborhoods (attention is directional). Self-attention neighborhoods within each image connect every node to its k_s nearest neighbors in image space (fixed k_s or radius); cross-attention candidate neighborhoods link each node to its top-k_c descriptors in the other image via an approximate nearest-neighbor (ANN) index over the dot-product $e^\top e'$ (ties broken by image-space distance). Optionally, cross links can be symmetrized by keeping only mutual top-k_c pairs, but message passing remains directed.

Attention-based message passing is applied for $l = 0, \ldots, L-1$. With d_a the attention width and temperature τ_a, define the query, key, and value projections as $Q = W_q h$, $K = W_k h$, and $V = W_v h$ with layer-dependent weights. For node i in image 1, the self-attention message is

$$\alpha_{ij}^{(l)} = \mathrm{softmax}_{j \in \mathcal{N}_i^{\mathrm{self}}}\left(\frac{Q_i^{1(l)\top} K_j^{1(l)}}{\sqrt{d_a}\tau_a}\right), \qquad m_{i,\mathrm{self}}^{1(l)} = \sum_{j \in \mathcal{N}_i^{\mathrm{self}}} \alpha_{ij}^{(l)} V_j^{1(l)}.$$

The cross-attention message from image 2 is

$$\beta_{ij}^{(l)} = \mathrm{softmax}_{j \in \mathcal{N}_i^{\mathrm{cross}}}\left(\frac{\tilde{Q}_i^{1(l)\top} \tilde{K}_j^{2(l)}}{\sqrt{d_a}\tau_a}\right), \qquad m_{i,\mathrm{cross}}^{1(l)} = \sum_{j \in \mathcal{N}_i^{\mathrm{cross}}} \beta_{ij}^{(l)} \tilde{V}_j^{2(l)},$$

with independent projections $\tilde{W}_q$, $\tilde{W}_k$, and $\tilde{W}_v$ for cross-attention. The update rule is

$$h_i^{1(l+1)} = h_i^{1(l)} + \mathrm{MLP}\left(\left[h_i^{1(l)} \| m_{i,\mathrm{self}}^{1(l)} \| m_{i,\mathrm{cross}}^{1(l)}\right]\right),$$

and an analogous formulation is applied to nodes in the second image. Residual connections and layer normalization stabilize training, and multi-head attention can be used when required.

After L layers the pairwise logits are

$$Z_{ij} = (W_m h_i^{1(L)})^\top (W'_m h_j^{2(L)}) + \gamma \langle e_i^1, e_j^2 \rangle + \delta \psi(\ell_i^1, \ell_j^2),$$

where ψ encodes weak geometric priors such as epipolar bearings or relative scale, and γ and δ are learnable scalars. Logits are computed only for candidate pairs $(i, j) \in \mathcal{N}^{\mathrm{cross}}$ (we do not instantiate a dense $v_1 \times v_2$ matrix); we augment this sparse support with a dustbin row and column and apply masked sparse Sinkhorn to obtain a doubly-stochastic assignment with unmatched probabilities.

At inference time, mutual high-confidence pairs are extracted from the soft assignment and validated using a minimal solver inside RANSAC. The essential matrix is used for 2D-2D matching, and PnP is applied when 3D points exist. The resulting inlier set is passed to the SLAM backend, and during training its mask can serve as an auxiliary supervision signal for the attention layers.

Algorithm 1: Graph-Attention Matcher

Input: Keypoints Q_1, Q_2 with (ℓ, e, c); hyperparameters k_s, k_c, L
Output: Verified matches $\mathcal{M}$
Build self-neighborhoods $\mathcal{N}^{\text{self}}$ in each image using a radius or k_s nearest neighbors;
For each node, build cross-candidate sets $\mathcal{N}^{\text{cross}}$ by top-k_c descriptor similarity;
Initialize $h^{1(0)}$ and $h^{2(0)}$ with descriptor and position embeddings;
for $l \leftarrow 0$ **to** $L - 1$ **do**
  Compute self-attention messages within each image;
  Compute cross-attention messages across images;
  Update $h^{1(l+1)}$ and $h^{2(l+1)}$ with residual MLPs;
Compute logits Z and apply Sinkhorn with a dustbin to obtain the soft assignment $\bar{P}$;
Extract mutual matches $\mathcal{M}_0 = \{(i, j) : \bar{P}_{ij} > \tau_p \text{ and mutual}\}$;
Verify $\mathcal{M}_0$ with RANSAC to obtain the inlier set $\mathcal{M}$;
return $\mathcal{M}$;

This model assigns each feature a unique identity through its node embedding and refines it with layered self and cross context. Using top-k descriptor candidates and k_s image-space neighbors keeps the encoder near-linear in the number of keypoints, and the masked Sinkhorn on the candidate support preserves this scaling (a dense $v_1 \times v_2$ assignment would reintroduce $O(v_1 v_2)$ costs). This joint reasoning mitigates ambiguities from repetitive textures and maintains reliable matching under large viewpoint changes, providing robust inputs to downstream geometric estimation.

Computational complexity. Let v_1, v_2 be the numbers of keypoints and $V = v_1 + v_2$, with descriptor dim. d, hidden/attention widths d_h, d_a, and L layers. We build $E_{\text{self}} \approx v_1 k_s + v_2 k_s$ and $E_{\text{cross}} \approx v_1 k_c + v_2 k_c$ directed edges. Self neighborhoods via a KD-tree cost $O(V \log V + V k_s)$ time and $O(V)$ memory (worst case $O(V^2)$ for a large radius), while the ANN over descriptors costs $O(Vd)$ to build and $O\left(V \, \text{ann\_cost}(k_c)\right)$ to query. Each layer computes projections in $O(V d_h d_a)$, attention over edges in $O\left((E_{\text{self}} + E_{\text{cross}}) d_a\right)$, and the update MLP in $O(V d_h^2)$; the encoder therefore costs $O\left(L\left((E_{\text{self}} + E_{\text{cross}}) d_a + V d_h^2\right)\right)$ time and $O(V d_h + E_{\text{self}} + E_{\text{cross}})$ memory. Pairwise logits are evaluated only on candidate pairs, for $O(E_{\text{cross}} d_h)$ time, and the masked sparse Sinkhorn on the same support costs $O(T E_{\text{cross}})$ time and $O(E_{\text{cross}})$ memory for T iterations (a dense $v_1 \times v_2$ normalization would cost $O(T v_1 v_2)$). RANSAC for the essential matrix or PnP requires $N = \frac{\log(1-p)}{\log(1-w^s)}$ iterations in expectation (s=5 for the essential matrix; s=3/4 for PnP), with $O(|\mathcal{M}_0|)$ inlier checks per iter. With fixed k_s, k_c and $d_a \sim d_h$, the end-to-end pipeline scales near-linearly in V.

5.2 Optimization Layer

The optimization layer converts the matcher's soft assignments into a compact and geometrically consistent set of correspondences suitable for pose estimation. Computing and verifying all $m \times n$ pairs is unnecessary, so the pipeline sparsifies the candidates, reweights them using local consistency, and verifies them geometrically.

Let $\bar{P}$ denote the Sinkhorn-normalized assignment that includes a dustbin. Unmatched mass is discarded and only the real block $P \in \mathbb{R}^{v_1 \times v_2}$ is kept. A mutual pruning together with a confidence

threshold yields the candidate set

$$\mathcal{M}_0 = \{(i, j) : P_{ij} \geq \tau_p,\ j = \arg\max_{j'} P_{ij'},\ i = \arg\max_{i'} P_{i'j}\}.$$

Optionally, a per-row/column top-t pruning can be applied before the mutual check in settings where mutuality is not enforced.

Some pairs in $\mathcal{M}_0$ may remain ambiguous, so each pair (i, j) is scored by comparing the relative geometry of their local neighborhoods. Let $\mathcal{N}_i^1$ be the k_s nearest neighbors of i in image 1 for which a candidate $(i', j') \in \mathcal{M}_0$ exists, and define $\mathcal{N}_j^2$ analogously around j. Define unit direction vectors $u_{ii'} = \frac{\ell_{i'}^1 - \ell_i^1}{\|\ell_{i'}^1 - \ell_i^1\|}$ and $v_{jj'} = \frac{\ell_{j'}^2 - \ell_j^2}{\|\ell_{j'}^2 - \ell_j^2\|}$. The agreement score is

$$\kappa_{ij} = \frac{1}{|\mathcal{M}_i|} \sum_{(i',j') \in \mathcal{M}_i} \max\left(0,\ u_{ii'}^\top v_{jj'}\right), \quad \mathcal{M}_i = \{(i', j') \in \mathcal{M}_0 : i' \in \mathcal{N}_i^1,\ j' \in \mathcal{N}_j^2\}.$$

This favors pairs whose local neighborhoods undergo a similar transform that is approximately rigid or a similarity.

Sampling weights are then formed as

$$w_{ij} = \frac{(P_{ij})^\alpha (\kappa_{ij} + \epsilon)^\beta}{\sum_{(p,q) \in \mathcal{M}_0} (P_{pq})^\alpha (\kappa_{pq} + \epsilon)^\beta},$$

and a PROSAC or NG-RANSAC estimator is run for the essential matrix in the 2D-2D case or for PnP in the 3D-2D case when map points exist, drawing minimal samples with probability w_{ij}. After estimating the model $\hat{E}$ or the pose parameters $\hat{\theta}$, Sampson or reprojection residuals r_{ij} are computed and inliers are accepted as

$$\mathcal{M} = \{(i, j) \in \mathcal{M}_0 : r_{ij} \leq \tau_{\text{geo}}\}.$$

A final confidence for downstream weighting is

$$\hat{w}_{ij} = \text{sigmoid}(\eta P_{ij}) \exp(-\rho(r_{ij})/\sigma)(\kappa_{ij} + \epsilon),$$

where ρ is a robust kernel such as Huber and η, σ are tunable parameters.

Because cross-attention is restricted to top-k_c candidates per keypoint (retrieved via ANN) and self-attention to k_s neighbors, the number of pairs considered scales as $O(v(k_c + k_s))$ after the ANN retrieval step (near $O(v \log v)$ in practice), instead of $O(MN)$. The dustbin in $\bar{P}$ naturally handles occlusions and missing detections and avoids ad-hoc correction rows or columns and linear-programming solvers. The combination of mutual pruning, neighborhood agreement, and guided RANSAC yields a compact, high-quality match set that reduces pose-estimation errors and increases robustness under large viewpoint changes.

5.3 Training Objective

This section summarizes how the Sinkhorn-based matching objective combines with the non-differentiable geometric verification used as supervision.

The graph-attention matcher, including the Sinkhorn assignment, is differentiable. The optimization layer, which performs mutual pruning, neighborhood-consistency scoring, and RANSAC or PROSAC geometric verification, is non-differentiable and is used only to produce supervision signals such as inlier masks. Gradients are stopped at the soft assignment $\bar{P}$. Let M and N be the numbers of keypoints in G_1 and G_2, respectively. The matcher produces augmented logits $\bar{Z} \in \mathbb{R}^{(M+1)\times(N+1)}$ with a dustbin row and column for unmatched points and a doubly-stochastic soft assignment

$$\bar{P} = \mathrm{Sinkhorn}\left(\frac{\bar{Z}}{\tau}\right).$$

From homography-based supervision, or from geometry-verified inliers, a sparse target matrix $\bar{Y}$ is built with three disjoint index sets:

$$\mathcal{P} = \{(i,j) : \text{matched}\}, \quad \mathcal{U}_1 = \{i : \text{unmatched in } G_1\}, \quad \mathcal{U}_2 = \{j : \text{unmatched in } G_2\}.$$

The negative log-likelihood over the augmented domain is optimized:

$$L_{\text{match}} = -\sum_{(i,j)\in\mathcal{P}} \log \bar{P}_{ij} - \sum_{i\in\mathcal{U}_1} \log \bar{P}_{i,\emptyset} - \sum_{j\in\mathcal{U}_2} \log \bar{P}_{\emptyset,j}.$$

Optionally, positives are weighted by a geometric consistency factor

$$\omega_{ij} = \exp\left(-\frac{\rho(r_{ij})}{\sigma}\right),$$

derived from the Sampson or reprojection residual r_{ij} and a robust kernel ρ, yielding

$$L^{\omega}_{\text{match}} = -\sum_{(i,j)\in\mathcal{P}} \omega_{ij} \log \bar{P}_{ij} + \cdots.$$

An entropy bonus on the rows and columns of $\bar{P}$ can be included early in training to avoid overconfident spurious assignments.

This objective penalizes low-probability mass on true correspondences and encourages correct routing to the dustbin for occluded or missing keypoints. Because gradients flow through Sinkhorn and the attention projections, the network learns to raise match confidence for geometrically consistent pairs while suppressing ambiguous matches under large viewpoint changes.

5.4 Model Pretraining

The extractor and the matcher are implemented in PyTorch and are pretrained in a self-supervised fashion on MS-COCO [27]. No manual keypoint labels are used; supervision arises from homography-induced correspondences between synthetically warped image pairs. Images are converted to grayscale and resized to 240×320. Standard photometric augmentations (brightness and contrast jitter, gamma, Gaussian noise and blur) and random erasing simulate illumination change and occlusions.

For homography synthesis, four source points are sampled within the central region of each image, perturbed, and used to compute a full projective transform T that composes perspective, rotation, and

scale. The pair $(I, T(I))$ provides dense 2D-2D correspondences under a local planar assumption. To reduce bias, multi-homography sampling over disjoint tiles and random scaling is applied.

Pretraining proceeds in two stages. Stage 1 trains the extractor alone with

$$L_{\text{ext}} = \lambda_{\text{rep}} L_{\text{rep}} + \lambda_{\text{unif}} L_{\text{unif}} + \lambda_{\text{desc}} L_{\text{desc}},$$

where L_{rep} enforces cross-view detection repeatability under synthetic homographies, L_{unif} discourages spatial clustering of detections, and L_{desc} is a contrastive descriptor loss. For descriptor contrastive terms we sample top-v candidates per image with stop-gradient selection, while detection losses remain dense. We use output stride s=8 and descriptor dimension d=256. Stage 2 attaches the graph-attention matcher and optimizes the Sinkhorn-based assignment with a dustbin using

$$L_{\text{match}} + \lambda_{\text{geo}} L_{\text{geo}} + \lambda_{\text{desc}} L_{\text{desc}} + \lambda_{\text{ent}} L_{\text{ent}},$$

while keeping the extractor unfrozen. A short joint fine-tuning can be applied to stabilize end-to-end performance.

Optimization uses Adam with $(\beta_1 = 0.9, \beta_2 = 0.999)$, weight decay 10^{-4}, an initial learning rate of 2×10^{-4} with a 10k-step warmup and cosine decay, batch size 16, and 300k training iterations. Unless stated otherwise, the coefficients are λ_{rep}=1.5, λ_{unif}=1.0, and λ_{desc}=1.2 for the extractor, and λ_{match}=1.0, λ_{geo}=0.5, λ_{desc}=0.5, and λ_{ent}=0.01 for the matcher, with temperatures τ_a=0.2 for attention and τ_d=0.07 for descriptors. Top-k neighborhoods use k_s=8 for self-attention and k_c=64 for cross-attention. This setup yields repeatable and well-distributed detections and robust correspondences under strong viewpoint and illumination changes while remaining efficient at inference time.

6. EXPERIMENTS

The evaluation comprises four parts:

1. detector quality (prior position–estimation feature extraction),
2. matcher quality (graph-attention matching),
3. SLAM trajectory accuracy on public datasets,
4. real-world robot experiments.

Unless stated otherwise, experiments run on an AMD Ryzen 5 3500X (3.6 GHz), an NVIDIA RTX 3070 (8 GB), 16 GB RAM, Ubuntu 24.04.

For real-world tests, a TurtleBot3 with an Intel RealSense D435i is used. Public datasets include TUM RGB-D (indoor) [28], and KITTI odometry (outdoor) [29]. In SLAM experiments, the specific sensing modality (monocular/RGB-D) used by each method is reported to ensure fair comparisons.

6.1 Detector Repeatability on HPatches

Detector repeatability is assessed on HPatches [30], focusing on illumination changes and low-texture scenes. The comparison includes ORB, SuperPoint [7], and GCNv2 (a learned CNN detector with a binary descriptor). All methods output at most v keypoints per image with the same v and NMS radius; evaluation uses the official image pairs with provided homographies.

Repeatability follows the HPatches protocol and is defined as the fraction of detections that find a correspondence under the ground-truth homography. A detection ℓ in image 1 is deemed repeatable if its warped location $T\ell$ lies within ε pixels of any detection in image 2; the mean over both directions and all pairs in a sequence is reported. Unless noted, $\varepsilon = 3\,\mathrm{px}$ and $v = 1000$. This metric isolates the detector and is independent of descriptor quality.

Two representative illumination sequences are reported explicitly: i_castle (strong illumination gradients) and i_whitebuilding (texture sparsity), with a summary across five illumination sequences. All methods operate on grayscale 240×320 images with identical preprocessing and NMS.

In i_castle (FIGURE 5), ORB detections tend to cluster on high-contrast regions, leading to uneven spatial coverage that can hurt downstream geometry. SuperPoint and GCNv2 maintain more uniform coverage in bright frames but exhibit a reduced number of stable detections as contrast drops. The position–prior head keeps a near-constant count and spread by proposing one candidate per grid cell with sub-cell refinement.

In the texture-sparse i_whitebuilding (FIGURE 6), gradient-based saliency is weak, causing ORB and learned baselines to drop detections or concentrate them around a few structures. The grid-based position prior encourages distributed proposals, improving the chance of overlap across views.

TABLE 1, reports repeatability (higher is better). The proposed method attains the best mean in four of five sequences and remains competitive on i_fruits. Gains are most pronounced on i_whitebuilding, consistent with the qualitative behavior above.

Table 1: Detector repeatability on HPatches [30], illumination sequences (mean over pairs)

Dataset	ORB	SuperPoint	GCNv2	Proposed
i_books	0.57	0.59	0.63	**0.69**
i_castle	0.58	0.61	0.64	**0.72**
i_fruits	0.54	0.57	**0.60**	0.59
i_ski	0.56	0.58	0.60	**0.65**
i_whitebuilding	0.52	0.51	0.55	**0.62**

Because repeatability relies on homographies, it primarily probes planar or near-planar content; nevertheless, the improvements carry over to general scenes by stabilizing counts and coverage across frames. Later sections verify that these detector properties translate into stronger correspondence sets and better pose tracking within the SLAM pipeline.

Figure 5: Detector responses on HPatches i_castle under illumination change. Top: input images; bottom: top-v detections (color encodes confidence). The position–prior detector preserves coverage and repeatability as brightness decreases.

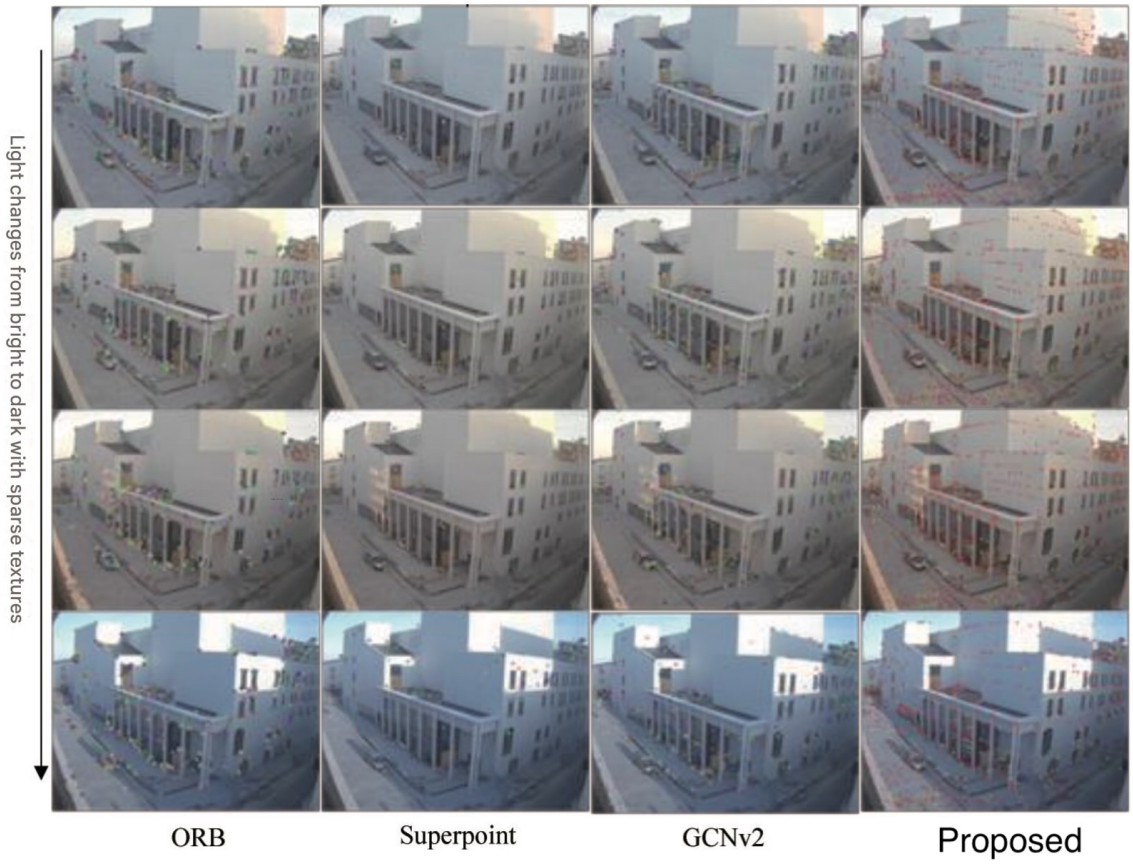

Figure 6: Detector responses on HPatches i_whitebuilding (illumination variation, texture-sparse). The detector maintains distributed keypoints despite weak gradients.

6.2 Matching MaP on TUM and HPatches

Multi-view matching is evaluated on TUM RGB-D (indoor) and HPatches (outdoor/illumination) under viewpoint and appearance changes, including texture-sparse scenes. Four pipelines are compared: ORB+NN (Hamming nearest neighbors with ratio test, mutual check, RANSAC), SuperPoint+NN (ℓ_2 kNN + ratio + mutual + RANSAC), GCNv2+NN (binary NN + ratio + mutual + RANSAC), and the proposed method (graph-attention matcher with Sinkhorn assignment and dustbin, followed by geometric verification). All methods use the same number of detections ($\nu = 1000$) and identical NMS radii. For fairness, only RGB is used at matching time; depth or poses are used only to define ground truth for evaluation on TUM.

The metric is mean Average Precision (mAP) computed from precision–recall curves obtained by sweeping a confidence threshold. For the baselines the score is the normalized descriptor similarity after the ratio test; for the proposed matcher the score is the Sinkhorn assignment probability P_{ij}. A match is considered correct if its reprojection error under ground truth is within $\varepsilon = 3\,\mathrm{px}$:

- HPatches: correctness is checked with the provided homography.
- TUM: ground-truth camera poses (and, when available, depth to back-project) are used to compute reprojection error; otherwise evaluation is performed against the ground-truth essential matrix via Sampson distance.

Viewpoint analysis uses $30°/60°/90°$ bins. For HPatches, we approximate viewpoint change using a local in-plane rotation proxy derived from the ground-truth homography: we sample a grid, compute the local 2×2 Jacobian $A(x)$, take its polar decomposition $A = RS$, and use the median $|\angle(R)|$; pairs are then assigned to the nearest of $\{30°, 60°, 90°\}$ (thresholds $<45°$, $45°$–$75°$, $\geq 75°$). This is a proxy for viewpoint change and may differ from the true 3D rotation on non-planar scenes. For TUM, the true relative rotation $\theta = \arccos((\mathrm{tr}(R) - 1)/2)$ is used.

Figure 7, visualizes matches (red: high confidence, purple: low). In indoor scenes with repetitive structure (TUM) and outdoor illumination changes (HPatches), nearest-neighbor pipelines retain many ambiguous pairs that are later rejected by RANSAC, reducing recall. The graph-attention matcher suppresses ambiguous regions through self-attention and concentrates probability mass on geometrically consistent candidates via cross-attention and Sinkhorn, improving both precision and recall.

Figure 8, summarizes mAP under illumination changes; the proposed approach improves descriptor-level matching by +9.27% over GCNv2+NN and +12.93% over SuperPoint+NN on the evaluated sequences.

TABLE 2, reports mAP versus viewpoint bins for representative sequences ("v_*"). Across methods, accuracy decreases with larger rotations, but the matcher degrades more gracefully. Averaged over the five sequences shown, improvements over GCNv2+NN are +22.4% at 30°, +55.1% at 60°, and +84.9% at 90°; over SuperPoint+NN the gains are +28.5%, +58.9%, and +92.4%, respectively.

Graph-attention with Sinkhorn-based assignment leverages both appearance and spatial context to produce cleaner and denser correspondences. The gains are most pronounced at extreme viewpoints

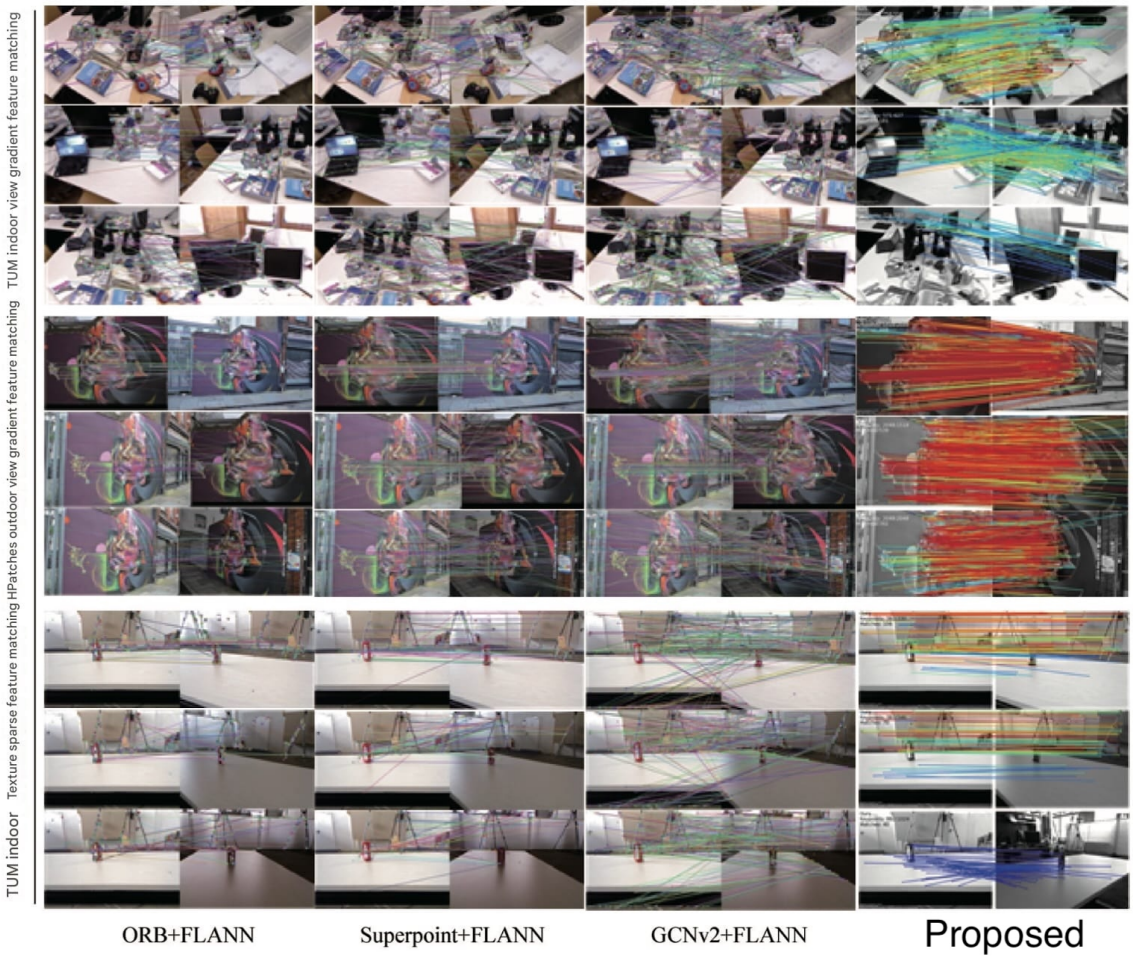

Figure 7: Matching under viewpoint variation on TUM (indoor) and HPatches (outdoor). Colors encode match confidence. The method yields denser, cleaner correspondences in challenging views.

Table 2: mAP under viewpoint variation (representative sequences). Higher is better.

Dataset	ORB+NN			SuperPoint+NN			GCNv2+NN			Proposed		
	30°	60°	90°	30°	60°	90°	30°	60°	90°	30°	60°	90°
v_abstract	0.64	0.46	0.30	0.62	0.47	0.24	0.73	0.53	0.29	**0.80**	**0.74**	**0.51**
v_bees	0.65	0.48	0.26	0.70	0.54	0.39	0.65	0.50	0.37	**0.86**	**0.69**	**0.57**
v_beyus	0.68	0.50	0.31	0.61	0.45	0.27	0.61	0.43	0.32	**0.83**	**0.74**	**0.57**
v_home	0.61	0.40	0.27	0.63	0.41	0.29	0.73	0.47	0.25	**0.80**	**0.71**	**0.54**
v_woman	0.58	0.39	0.24	0.65	0.43	0.26	0.67	0.42	0.24	**0.84**	**0.74**	**0.52**

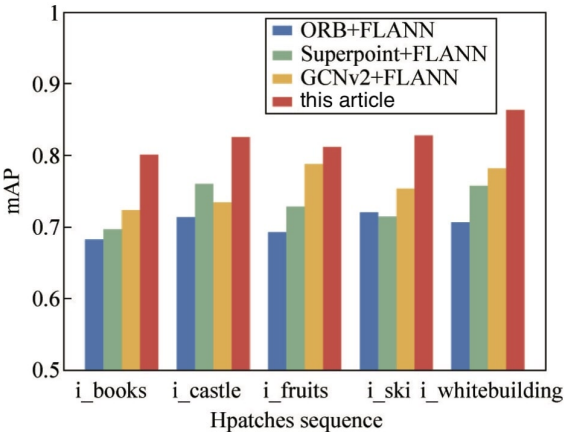

Figure 8: HPatches (illumination): mAP of descriptor matching.

and under illumination change, which are regimes where nearest-neighbor pipelines struggle the most.

6.3 Trajectory Accuracy

6.3.1 TUM RGB-D

Performance under large-viewpoint motion is assessed on seven indoor sequences from the TUM RGB-D dataset, comparing against three representative SLAM systems: ORB-SLAM2, DX-SLAM, and GCNv2-SLAM. For all methods, RGB-only tracking and mapping is used: depth frames are never used to create or track map points (PnP relies only on monocularly triangulated 3D points). Depth is used strictly for evaluation. To account for unknown monocular scale, ATE is computed after 7-DoF $\mathrm{Sim}(3)$ alignment with `evaluate_ate.py`, and RPE is reported with `evaluate_rpe.py` [31].

- ORB-SLAM2 uses ORB keypoints with Hamming NN matching; camera motion is estimated by RANSAC (essential/fundamental matrix for 2D–2D or PnP when 3D map points exist), followed by local/global optimization.
- DX-SLAM integrates learned (CNN-based) local features and global descriptors within the ORB-SLAM2 framework to improve robustness to appearance change.
- GCNv2-SLAM adopts a learned detector and binary descriptor (implemented with CNNs despite the name) to increase efficiency relative to handcrafted ORB features.

FIGURE 9 shows qualitative trajectory overlays. The proposed method achieves the lowest errors on six of the seven sequences, with stable tracking in texture-sparse or highly oblique views. ORB-SLAM2 exhibits larger drift when viewpoint or appearance changes reduce match support. DX-SLAM experiences occasional tracking loss on more challenging sequences (e.g., fr2/room,

fr2/desk), while GCNv2-SLAM is generally robust but can fail under extreme viewpoint or low-texture conditions, increasing ATE.

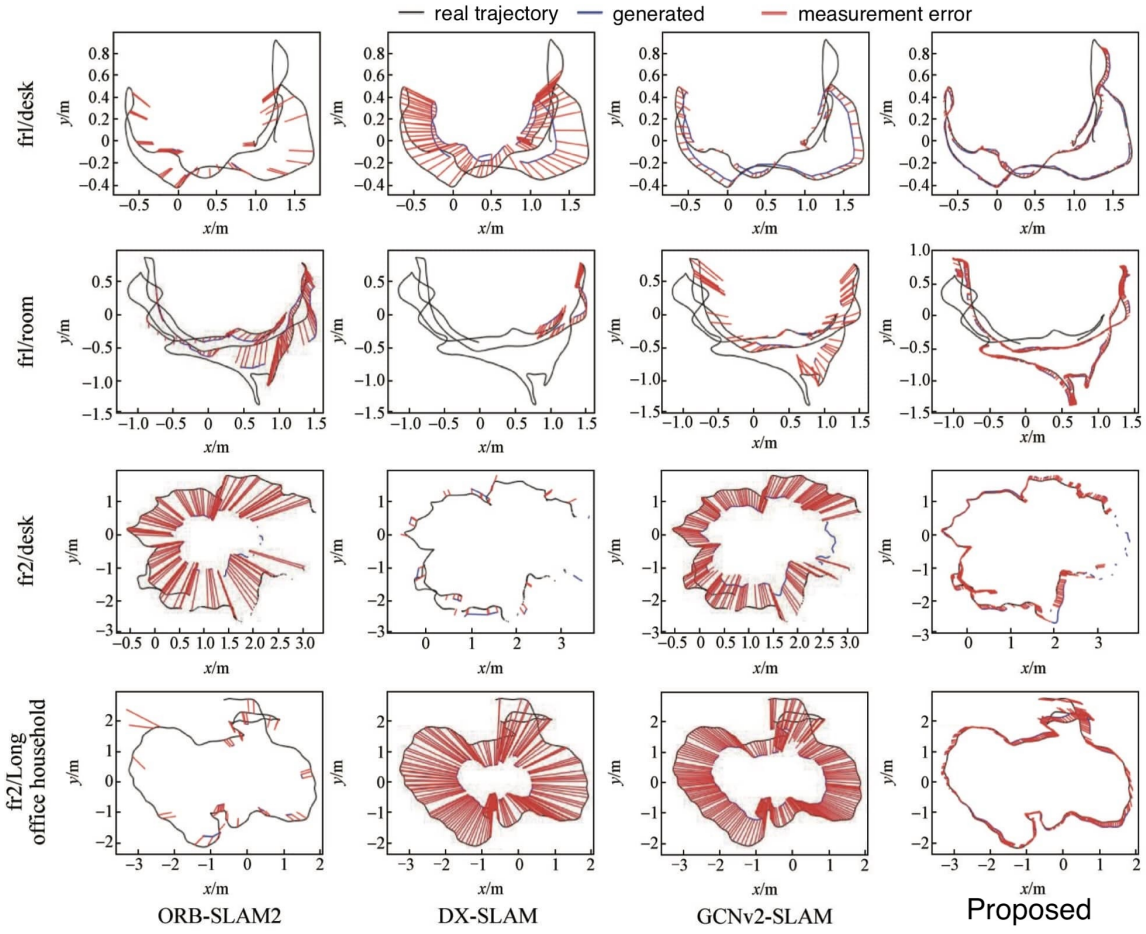

Figure 9: Trajectory comparison on TUM RGB-D (RGB-only tracking; depth used for evaluation).

The quantitative comparison is summarized in Table 3. The proposed approach attains the best ATE and RPE in six sequences; GCNv2-SLAM leads on fr1/360. Averaged over all sequences, ATE and RPE are reduced substantially relative to the learned and handcrafted baselines (see table footnote for aggregate percentages). These results indicate that augmenting the frontend with the position–prior detector and graph-attention matcher benefits pose estimation under large viewpoint changes while preserving robustness indoors.

6.3.2 KITTI Odometry

Outdoor performance is evaluated on KITTI odometry sequences 00–10, which contain urban, suburban, and highway scenes with sharp turns and long, large-curvature trajectories. Because

Table 3: TUM dataset: average trajectory errors (ATE/RPE). Best per row in bold.

Dataset	ORB-SLAM2		DX-SLAM		GCNv2-SLAM		Proposed	
	ATE	RPE	ATE	RPE	ATE	RPE	ATE	RPE
fr1/360	—	—	—	—	**0.06**	**0.08**	0.15	0.19
fr1/floor	0.39	0.47	—	—	0.14	0.17	**0.02**	**0.04**
fr1/desk	0.08	0.10	0.30	0.36	0.19	0.23	**0.04**	**0.05**
fr1/desk2	0.57	0.99	0.34	0.42	0.16	0.19	**0.05**	**0.07**
fr1/room	0.29	0.35	0.18	0.21	0.25	0.31	**0.07**	**0.11**
fr2/desk	0.93	1.15	0.58	0.71	0.16	0.20	**0.11**	**0.13**
fr3/office	1.22	1.55	0.79	1.01	0.36	0.45	**0.17**	**0.17**

per-pixel RGB-D depth is not provided (KITTI offers stereo image pairs, from which depth can be estimated but we deliberately do not use stereo), we run all methods on the left camera only (pure monocular tracking) for fairness.

Trajectories are evaluated with `evo` using standard KITTI tools: ATE (RMSE) via `evo_ape kitti` and RPE via `evo_rpe`. For monocular pipelines a global $\mathrm{Sim}(3)$ alignment (Umeyama) is performed before ATE to account for unknown scale; RPE is computed on translational drift over fixed distance segments. All methods use identical image resolution and camera intrinsics.

The comparison includes ORB-SLAM2 (monocular) and DX-SLAM (monocular learned features within the ORB-SLAM2 framework). The proposed system replaces the frontend (detector+descriptor+matcher) while retaining an ORB-SLAM2-compatible backend (local BA, loop closing, pose-graph optimization).

FIGURE 10, shows trajectory overlays. On sequences with large curvature (e.g., 00, 05, 09) ORB-SLAM2 exhibits larger deviations when viewpoint changes and appearance transitions reduce match support; on 09, sharp turns and repetitive facades occasionally prevent loop closure. DX-SLAM improves stability but may still suffer tracking loss in extreme viewpoint segments. The proposed method maintains denser, cleaner correspondences, reducing drift and aiding loop closures.

Across sequences 00–10, Figure 11, reports ATE (RMSE) and RPE. For illustration, the ATE (RMSE, meters) on {00, 05, 09} is {63.17, 12.21, 111.68} for ORB-SLAM2 and {21.55, 15.20, 139.52} for DX-SLAM, while the proposed method achieves {11.83, 11.64, 14.07}. Overall, the proposed approach attains the best ATE in all 11 sequences and the best RPE in 8/11. On average, ATE is reduced by 38.5% and RPE by 11.24% relative to DX-SLAM; versus ORB-SLAM2 the reductions are 37.59% (ATE) and 19.67% (RPE).

The graph-attention matcher, combined with the position-prior detector, yields more reliable correspondences under large viewpoint change and appearance transitions. This translates into lower drift and stronger loop-closure corrections in outdoor driving, improving both ATE and RPE over handcrafted and CNN-only baselines.

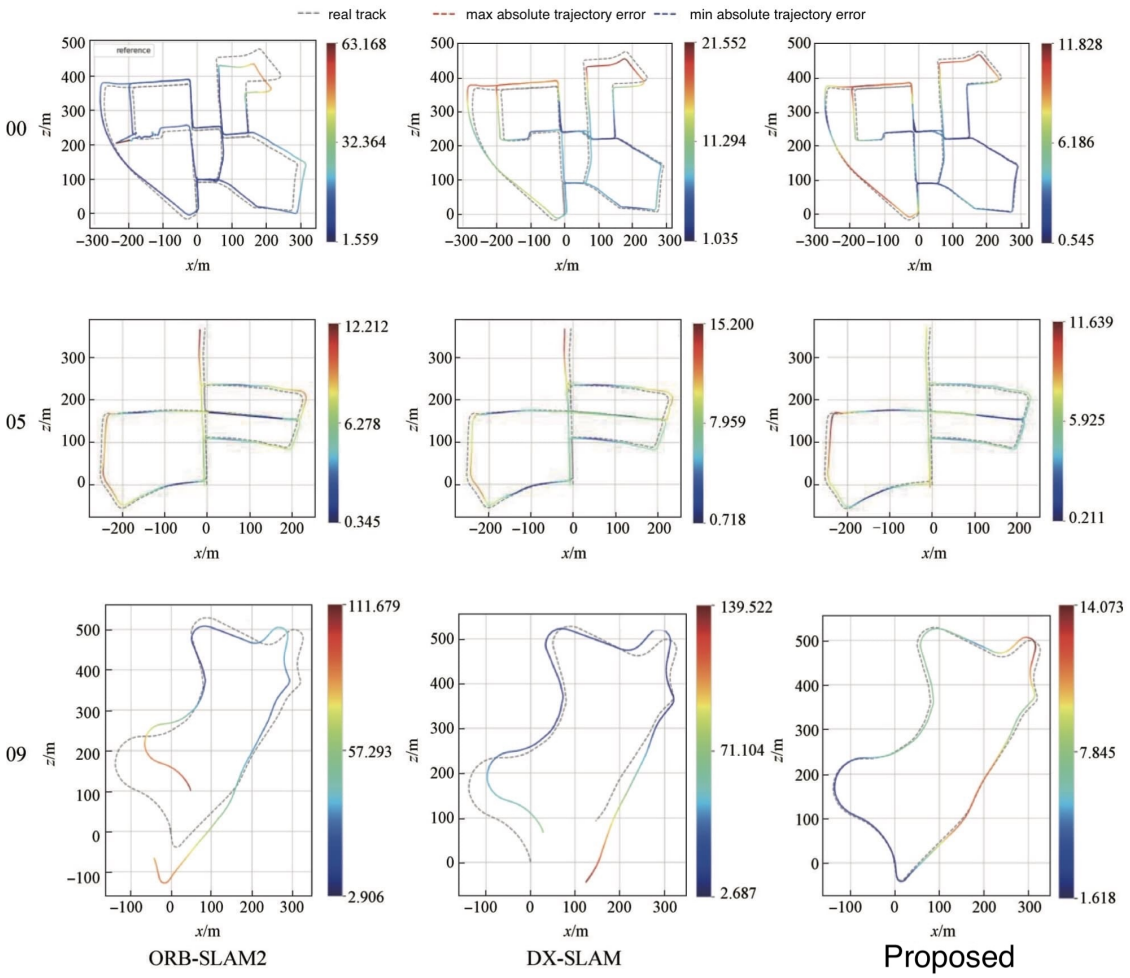

Figure 10: Trajectory comparison on KITTI (sequences 00, 05, 09 shown). All methods are monocular; $\mathrm{Sim}(3)$ alignment is applied for ATE.

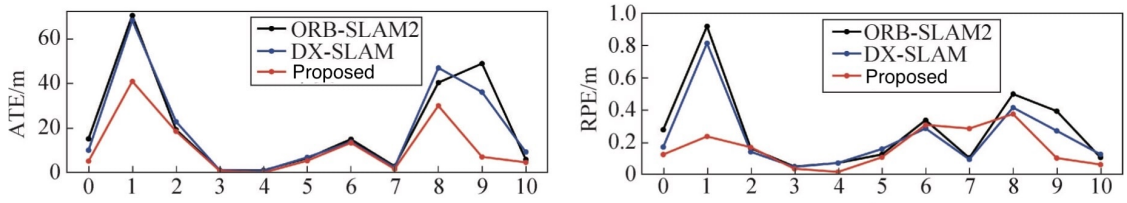

Figure 11: KITTI 00–10: ATE (RMSE, $\mathrm{Sim}(3)$-aligned) and RPE. The learned frontend reduces drift and improves consistency, especially on high-curvature trajectories.

6.4 Real-World Mobile Robot Experiments

The approach was further validated in a real indoor environment exhibiting strong illumination changes and low-texture areas, conditions that induce large apparent viewpoint shifts during motion. An Intel RealSense D435i was mounted on a TurtleBot3 at $\approx 0.3\,\mathrm{m}$ height. The robot was teleoperated along a figure-eight path around two obstacles (a, b), forming a loop in a $5.95\,\mathrm{m} \times 4.8\,\mathrm{m}$ area. The layout and qualitative trajectory overlays are shown in FIGURE 12. All methods used RGB-only tracking for fairness (depth was not used for pose estimation).

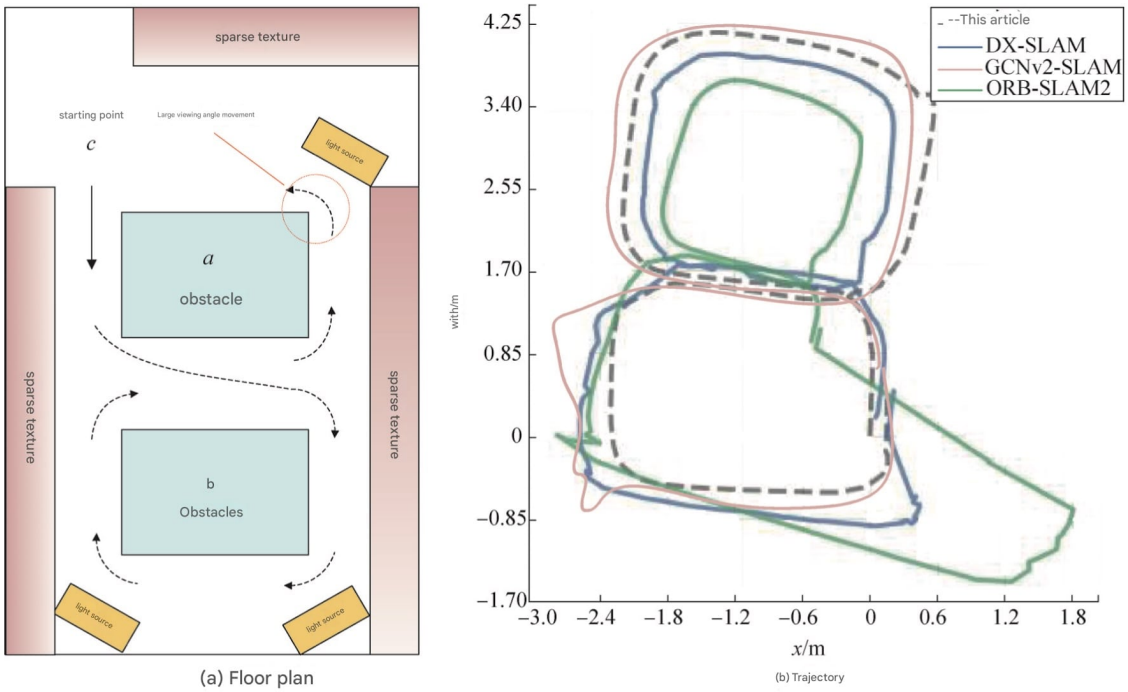

Figure 12: Real-world layout and trajectory overlays. The figure-eight induces viewpoint and appearance changes; the learned frontend maintains stable correspondences and closes the loop cleanly.

Qualitatively, ORB-SLAM2 produced a spurious loop closure near point c, which led to an incorrect global correction and map misalignment when the appearance changed sharply. DX-SLAM showed reduced match support during the tight turns, and its loop closure occasionally pulled the map toward a smaller scale. In contrast, the position–prior detector and graph-attention matcher preserved sufficient inliers through the challenging segments, enabling accurate loop closure and a well-aligned map.

The sequence comprised 714 frames with a target of 200 keypoints per frame. Per-frame times for feature extraction and matching are reported together with the measured end-to-end throughput. ORB-SLAM2 ran on CPU; learned methods (DX-SLAM, GCNv2-SLAM, Proposed) used the

GPU for their feature modules. Because of this device difference, absolute FPS are not strictly comparable; still, per-stage timings are informative.

Table 4: Average runtime on the figure-eight sequence. FPS is computed as $1000/(\text{total ms})$ and rounded.

Algorithm	extraction (ms)	matching (ms)	total time (ms)	throughput (frames/s)
ORB-SLAM2	12.9	15.1	28.0	**36**
DX-SLAM	32.4	39.2	71.6	14
GCNv2-SLAM	19.5	31.6	51.1	20
Proposed	16.3	30.8	47.1	21

Overall, the learned frontend sustained real-time operation and delivered cleaner loop closures and lower drift under strong appearance and viewpoint changes, supporting accurate mapping on a modest mobile platform.

7. LIMITATIONS AND FUTURE WORK

7.1 Limitations

Although the proposed frontend increases robustness to large viewpoint and illumination changes, several constraints remain. The self-supervised pretraining based on synthetic homographies introduces a bias toward locally planar warps and grayscale 240×320 imagery; strong non-planar parallax, rolling-shutter distortions, severe motion blur, or extreme low-light can still reduce calibration of the Sinkhorn probabilities and hurt recall under distribution shift. The system is designed and evaluated in a monocular setting, so absolute scale is unobservable and long, low-parallax segments may remain challenging despite improved matching. Loop detection relies on DBoW2 with ORB descriptors while tracking uses learned local features, creating a descriptor mismatch that may hinder place recognition under drastic appearance changes or repetitive patterns and requires storing heterogeneous descriptors per keyframe. Cross-attention is restricted to ANN-retrieved top-k_c candidates; in extreme viewpoints with very small overlap or heavy aliasing, true correspondences may be pruned before attention can recover them. Compute and memory budgets are also relevant: the learned extractor and matcher benefit from GPU acceleration and, even with dynamic gating, may introduce latency or throughput limitations on resource-constrained platforms; memory usage grows with per-point learned descriptors. The pipeline further assumes a largely static scene and relies on RANSAC to reject outliers; in highly dynamic environments the inlier set can shrink substantially without explicit motion handling. Finally, performance depends on thresholds for gating, non-maximum suppression, and assignment; suboptimal settings can delay or over-trigger switches between ORB and the learned frontend. With an output stride of s=8, very small structures may also be under-resolved, and gridwise one-candidate-per-cell can miss fine details near cell boundaries.

7.2 Future Work

There are several promising directions to address these limitations and broaden applicability. Incorporating additional sensing, such as stereo or RGB-D for overlap and scale, inertial data for short-term stabilization, and event cameras for high-speed or low-light motion, could improve tracking in challenging conditions, together with geometry that accounts for rolling shutter effects. Integrating place recognition with the learned local pipeline through global descriptors or vocabularies derived from the same features would remove the ORB/DBoW2 mismatch and reduce perceptual aliasing. A hierarchical, geometry-aware matching strategy that combines coarse-to-fine pyramids with epipolar or pose-conditioned attention could narrow candidate sets without sacrificing recall, while calibrated sparse optimal transport could maintain assignment quality at low computational cost. Explicit treatment of dynamic elements, through motion segmentation, semantic priors, or multi-model fitting, would help preserve inliers around moving objects. Online adaptation based on self-supervised updates that use photometric and geometric consistency, along with verification signals, could refine detector density, descriptor statistics, and gating thresholds when encountering new environments. To improve practical deployment, further efficiency work on quantization, distillation, fused kernels, and a strong CPU fallback is needed, along with compact storage of per-keyframe descriptors. Finally, calibrating assignment probabilities and propagating uncertainty into PnP and bundle adjustment, as well as examining descriptor aging and appearance-based map maintenance for long-term, large-scale scenarios, remain promising directions.

8. CONCLUSION

This work addresses the challenge of robust visual SLAM under large viewpoint changes and severe appearance variations (illumination, low texture), conditions that degrade handcrafted features and conventional nearest-neighbor matching. A graph neural network-based frontend for ORB-SLAM2 is proposed that improves both detection and matching while preserving real-time performance.

Main contributions are:

- A monocular framework that augments the frontend of ORB-SLAM2 with learned components while retaining its standard backend (local mapping, loop closing, and optimization), maintaining compatibility and scalability.
- A position–prior detector/descriptor that predicts one candidate per grid cell with sub-cell refinement and confidence, yielding repeatable and uniformly distributed keypoints that remain stable under illumination and texture scarcity.
- A graph-attention matcher with self-/cross-attention, Sinkhorn assignment, and a dustbin for unmatched points, providing reliable correspondences under large inter-frame viewpoint changes.
- A dynamic frontend policy that falls back to fast ORB tracking when conditions are favorable, activating the learned components only when match support degrades, thus preserving real-time throughput.

The approach was validated on TUM RGB-D, KITTI odometry, and real-world mobile robot experiments. Using RGB-only tracking for fair comparisons, the system consistently reduced trajectory error and improved loop-closure robustness versus ORB-SLAM2, DX-SLAM, and GCNv2-SLAM, particularly in high-curvature motion and strong appearance change. These results indicate that incorporating attention-based matching and position-aware detection into classical SLAM frontends offers a practical and effective path toward greater robustness in challenging real-world environments.

References

[1] Smith RC, Cheeseman P. On the Representation and Estimation of Spatial Uncertainty. Int J Robot Res. 1986;5:56-68.

[2] Dissanayake MW, Newman P, Clark S, Durrant-Whyte HF, Csorba M. A Solution to the Simultaneous Localization and Map Building (SLAM) Problem. IEEE Trans Robot Autom. 2001;17:229-241.

[3] Kümmerle R, Grisetti G, Strasdat H, Konolige K, Burgard W. G^2o: A General Framework for Graph Optimization. 2011 IEEE International Conference on Robotics and Automation. IEEE. 2011:3607-3613.

[4] Mur-Artal R, Tardós JD. ORB-SLAM2: An Open-Source Slam System for Monocular, Stereo, and RGB-D Cameras. IEEE Trans Robot. 2017;33:1255-1262.

[5] Rublee E, Rabaud V, Konolige K, Bradski G. ORB: An Efficient Alternative to SIFT or SURF. 2011 International Conference on Computer Vision. IEEE. 2011:2564-2571.

[6] Gálvez-López D, Tardos JD. Bags of Binary Words for Fast Place Recognition in Image Sequences. IEEE Trans Robot. 2012;28:1188-1197.

[7] DeTone D, Malisiewicz T, Rabinovich A. Superpoint: Self-Supervised Interest Point Detection and Description. 2018 Proceedings of the IEEE conference on computer vision and pattern recognition workshops. IEEE. 2018:224-236.

[8] Revaud J, De Souza C, Humenberger M, Weinzaepfel P. R2D2: Repeatable and Reliable Detector and Descriptor. In: NeurIPS. Curran Associates Inc. 2019:12414–12424.

[9] Tyszkiewicz M, Fua P, Trulls E. DISL: Learning Local Features With Policy Gradient. Adv Neural Inf Process Syst. 2020;33:14254-14265.

[10] Ono Y, Trulls E, Fua P, Yi KM. LF-net: Learning Local Features From Images. In: NeurIPS. 2018:6237-6247.

[11] Barroso-Laguna A, Riba E, Ponsa D, Mikolajczyk K. Key.Net: Keypoint Detection by Handcrafted and Learned CNN Filters. Proceedings of the IEEE/CVF international conference on computer vision. IEEE. 2019:5836-5844.

[12] Zhao X, Wu X, Miao J, Chen W, Chen PC, et al. ALIKE: Accurate and Lightweight Keypoint Detection and Descriptor Extraction. IEEE Trans Multimedia. 2023;25:3101-3112.

[13] Tateno K, Tombari F, Laina I, Navab N. CNN-SLAM: Real-Time Dense Monocular Slam With Learned Depth Prediction. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2017:6243-6252.

[14] Li D, Shi X, Long Q, Liu S, Yang W, Wang F, et al. DXSLAM: A Robust and Efficient Visual Slam System With Deep Features. IEEE/RSJ International conference on intelligent robots and systems (IROS). IEEE. 2020:4958-4965.

[15] Tang J, Ericson L, Folkesson J, Jensfelt P. GCNv2: Efficient Correspondence Prediction for Real-Time SLAM. IEEE Robot Autom Lett. 2019;4: 3505-3512.

[16] Sarlin PE, DeTone D, Malisiewicz T, Rabinovich A. Superglue: Learning Feature Matching With Graph Neural Networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. IEEE. 2020:4938-4947.

[17] Lindenberger P, Sarlin PE, Pollefeys M. Lightglue: Local Feature Matching at Light Speed. In: Proceedings of the IEEE/CVF international conference on computer vision. IEEE. 2023:17627-17638.

[18] Sun J, Shen Z, Wang Y, Bao H, Zhou X. LoFTR: Detector-Free Local Feature Matching With Transformers. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. IEEE. 2021:8922-8931.

[19] Wang Q, Zhang J, Yang K, Peng K, Stiefelhagen R. Matchformer: Interleaving Attention in Transformers for Feature Matching. In: Proceedings of the Asian conference on computer vision. ACVV. 2022:2746-2762.

[20] Campos C, Elvira R, Rodriguez JJ, M. Montiel JM, D. Tardos J. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multi-Map SLAM. IEEE Trans Robot. 2021;37:1874-1890.

[21] Nistér D. An Efficient Solution to the Five-Point Relative Pose Problem. IEEE Trans Pattern Anal Mach Intell. 2004;26:756-777.

[22] Lepetit V, Moreno-Noguer F, Fua P. EP*n*P: An Accurate *O(n)* Solution to the P*n*P Problem. Int J Comput Vis. 2009;81:155-166.

[23] Chum O, Matas J. Matching With PROSAC — Progressive Sample Consensus. In: CVPR. IEEE. 2005:220-226.

[24] Brachmann E, Rother C. Neural-Guided RANSAC: Learning Where to Sample Model Hypotheses. In: Proceedings of the IEEE/CVF international conference on computer vision. IEEE. 2019:4322-4331.

[25] Cuturi M. Sinkhorn Distances: Lightspeed Computation of Optimal Transport. Adv Neural Inf Process Syst. 2013;26.

[26] Johnson J, Douze M, Jégou H. Billion-Scale Similarity Search With GPUs. IEEE Trans Big Data. 2021;7:535-547.

[27] Lin TY, Maire M, Belongie S, Hays J, Perona P, et al. Microsoft COCO: Common Objects in Context. In: European Conference on Computer Vision. Springer. 2014:740-755.

[28] Sturm J, Engelhard N, Endres F, Burgard W, Cremers D. A Benchmark for the Evaluation of RGB-D SLAM Systems. IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE. 2012:573-580.

[29] Geiger A, Lenz P, Stiller C, Urtasun R. Vision Meets Robotics: The KITTI Dataset. Int J Robot Res. 2013;32:1231-1237.

[30] Balntas V, Lenc K, Vedaldi A, Mikolajczyk K. HPatches: A Benchmark and Evaluation of Handcrafted and Learned Local Descriptors. In: Proceedings of the IEEE conference on computer vision and pattern recognition. IEEE. 2017:5173-5182.

[31] https://github.com/weichnn/Evaluation_Tools