Fully Geometric Panoramic Localization

Junho Kim[1], Jiwon Jeong[3], and Young Min Kim[1, 2]

[1] Dept. of Electrical and Computer Engineering, Seoul National University

[2] Interdisciplinary Program in Artificial Intelligence and INMC, Seoul National University

[3] Dept. of Electrical Engineering, Stanford University

Abstract

We introduce a lightweight and accurate localization method that only utilizes the geometry of 2D-3D lines. Given a pre-captured 3D map, our approach localizes a panorama image, taking advantage of the holistic $360°$ view. The system mitigates potential privacy breaches or domain discrepancies by avoiding trained or hand-crafted visual descriptors. However, as lines alone can be ambiguous, we express distinctive yet compact spatial contexts from relationships between lines, namely the dominant directions of parallel lines and the intersection between non-parallel lines. The resulting representations are efficient in processing time and memory compared to conventional visual descriptor-based methods. Given the groups of dominant line directions and their intersections, we accelerate the search process to test thousands of pose candidates in less than a millisecond without sacrificing accuracy. We empirically show that the proposed 2D-3D matching can localize panoramas for challenging scenes with similar structures, dramatic domain shifts or illumination changes. Our fully geometric approach does not involve extensive parameter tuning or neural network training, making it a practical algorithm that can be readily deployed in the real world. Project page including the code is available through this link: https://82magnolia.github.io/fgpl/.

1. Introduction

Visual localization considers the problem of estimating the camera pose with respect to a 3D map using an input image. When designing a visual localization system, the key desiderata is defining the *3D map* and the associated *image features* to match against the map. The most common choice in modern localization systems is to use 3D maps from Structure-from-Motion (SfM) [47, 48] or dense laser scans [52] and match an image against extracted visual descriptors [2, 11, 34]. While being performant, such a design choice has several limitations [61]: (i) building and storing the map of image features can be costly, (ii) privacy breaches may occur in client-server localization scenarios as visual descriptors cannot securely hide clients'

(a) Fully Geometric Localization Setup
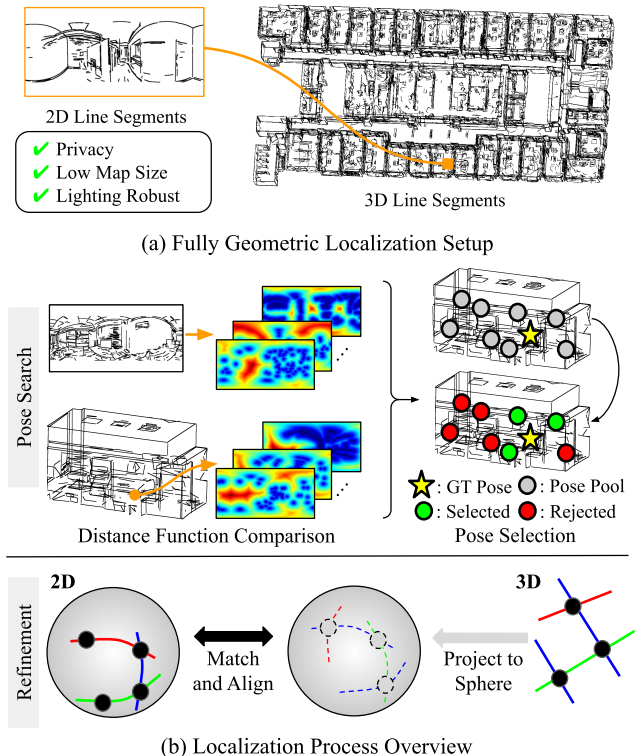

(b) Localization Process Overview

Figure 1. Overview of our method. (a) We target fully geometric localization using panoramas, where we only exploit lines in 2D and 3D. (b) Our method first searches for promising poses by comparing point and line distance functions that describe the holistic distribution of lines and their intersections. Then we refine each selected pose by aligning the line intersections on the sphere.

view [12, 43, 50], and (iii) localization may fail in challenging conditions such as lighting or scene changes.

In this paper, we argue that the holistic context of lines, when properly exploited, can be sufficient for performing accurate and scalable panoramic localization. To elaborate, as shown in Figure 1, we propose to *solely* use 3D lines as the map, and 2D lines extracted from the query panorama as features for localization. Thanks to the large field of view, localization methods using panoramas [8, 26–28, 58] are more robust against scene changes or repetitive struc-

tures compared to regular cameras. Further, the line maps can resolve the drawbacks of conventional descriptors: they are much cheaper to store than SfM or dense point maps, preserve user privacy as lines alone lack learnt/photometric features, and are robust to illumination changes.

From lines, we devise *fully geometric* and compact representations that summarize the geometric layout of lines as shown in Figure 1. Previous approaches for line-based localization extract rich information along the lines to establish correspondences between them [20, 33, 37, 44, 45], which often show inferior performance to prominent point features [32, 46] or require costly neural network inferences. We enhance the expressive power of our descriptor exploiting relative information between lines. Specifically, we first define principal directions to be the representative clusters of line directions. Then, we compute the intersections between pairs of lines having different principle directions. The intersection points, labeled with the two principle directions they are derived from, serve as the sparse yet distinctive representation for fine-grained localization.

We further propose a fast pipeline that matches the comprehensive distribution of lines while maintaining accuracy. We express the distribution of intersection points via point distance functions, which are the distance field measuring the closest distance to the intersection points on the spherical projection of the panorama. To efficiently compare distance functions in various of poses, we propose a formulation that decouples rotation and translation. Namely, we pre-compute and cache the translational variations of 3D distance functions, aligned on principal directions. Then, during localization we quickly compare them against the rotated versions of 2D distance functions. The matched pose can be further refined by aligning the intersection points and associated lines, as shown in Figure 1b. The resulting pipeline can quickly perform accurate localization against significantly large-scale line maps, without extensive hyperparameter tuning.

To summarize, our key contributions are as follows: (i) a fully geometric localization pipeline that overcomes the drawbacks of visual descriptors; (ii) novel spatial representations using line intersections for accurate pose search and refinement; and (iii) strategies for accelerating pose search in large-scale localization scenarios. Due to the efficiency and robustness previously difficult to achieve by geometric methods, we expect our pipeline to serve as the practical step towards fully geometric panoramic localization.

2. Related Work

Line-Based Localization Compared to points which are more frequently used for localization, lines can compactly describe the spatial layout of man-made structures, and thus can provide meaningful ques for localization [1, 14, 15, 18, 20, 28, 33, 36–38, 41, 42, 44, 45, 53, 56, 57]. Many exist-

ing approaches for line-based localization focus on establishing one-to-one matchings between line segments in the query and the 3D map. Here the matches are found from (i) learned features using CNNs [30, 41], Transformers [56] and graph neural networks [35, 36], or (ii) hand-crafted features describing the nearby texture [25, 59] and semantics [57] of lines. Another strand of approaches aim to jointly use points and lines during localization [15, 33, 42], which makes localization more robust in challenging scenes with repetitive structure or low texture. For panoramic localization, LDL [28] is a recently proposed work that promotes distributional matching of lines via line distance functions and avoids one-to-one line matching. Despite the effectiveness of the aforementioned approaches however, many works leverage the photometric information from the images used for line extraction, and thus are not *fully geometric*. We compare our method against exemplary line-based localization methods and demonstrate that our method can perform fast and accurate localization while only using the *geometry* of lines.

Localization without Visual Descriptors While visual descriptors have been the common choice in visual localization, there has been recent works exploiting geometric cues without the aid of visual descriptors. This is mainly motivated from (i) possible privacy breaches [6, 10, 12, 49, 50], and (ii) large map sizes required for storage while using visual descriptors. Existing methods can be classified into those that propose geometric methods for pose search and other that focus on pose refinement. For the former, LDL [28] and Micusik et al. [38] exploit lines as global geometric descriptors to perform pose search. For the latter, learning-based methods [9, 54, 61] propose to describe the geometric context of keypoint locations with neural networks, and optimization-based methods [7, 8] model keypoint locations as multi-modal probability distributions. While showing competitive results, these methods still require the use of visual descriptors in the remaining localization pipelines out of their interest. Our method can effectively localize while using geometric features for the entire pipeline, where we establish comparisons against existing methods in Section 5.

3. Preliminaries

Our method compares the holistic distribution of lines extracted from the 3D map and the panorama image, similar to LDL [28]. In this section, we describe shared components between LDL and our approach. Note that unlike LDL, our method only requires the line segments during localization, and thus being *fully geometric* without relying on costly keypoint descriptors.

Input Preparation LDL first extracts 2D and 3D line segments using off-the-shelf line extraction methods [17, 20, 55]. Throughout the remainder of the paper, we denote the 2D line segments as $L_{2D} = \{l = (s, e)\}$, where lines are represented as tuples of start and end points $s, e \in \mathbb{S}^2$ on the unit sphere. Similarly, we denote the 3D line segments as $L_{3D} = \{\tilde{l} = (\tilde{s}, \tilde{e})\}$, with the start and end points $\tilde{s}, \tilde{e} \in \mathbb{R}^3$. LDL further extracts three principal directions for 2D and 3D, as shown in Figure 2a. 2D principal directions $D_{2D} = \{d_i\}$ are extracted by estimating the three vanishing point directions with the largest number of incident lines. 3D principal directions $D_{3D} = \{\tilde{d}_j\}$ are similarly extracted by voting on the three most common line directions. After discarding lines that largely deviate from the principal directions, LDL clusters lines in 2D and 3D into three sets, namely $\mathcal{L}_{2D}^{cls} = \{L_{2D}^1, L_{2D}^2, L_{2D}^3\}$ and $\mathcal{L}_{3D}^{cls} = \{L_{3D}^1, L_{3D}^2, L_{3D}^3\}$. Each set contains the set of lines that are parallel to the designated principal direction.

Pose Search Using Line Distance Functions For each query, LDL generates a pool of $N_t \times N_r$ poses consisting of N_t translations and N_r rotations. The translation pool is set as grid partitions of the 3D line map, whereas the rotation pool is set by combinatorial association between the 2D and 3D principal directions, as shown in Figure 3a. Specifically, there can exist $2^3 \times 3!$ associations between principal directions in 2D and 3D considering sign and permutation ambiguity, where a single rotation is computed for each association using the Kabsch algorithm [23].

LDL compares line distance function of the 2D input against those generated from the pool of poses, and selects top-K poses to refine. 2D line distance functions are defined over the unit sphere $\mathbb{S}^2$ as the spherical distance to the closest line segment, namely

$$f_{2D}^{\mathrm{L}}(x; L_{2D}) = \min_{l \in L_{2D}} d_{\mathrm{L}}(x, l). \quad (1)$$

3D line distance functions are similarly defined for each pose $R \in SO(3), t \in \mathbb{R}^3$,

$$f_{3D}^{\mathrm{L}}(x; L_{3D}, R, t) = \min_{\tilde{l} \in L_{3D}} d_{\mathrm{L}}(x, \Pi_L(\tilde{l}; R, t)), \quad (2)$$

where $\Pi_L(\tilde{l}; R, t)$ denotes the projection of the transformed 3D line segment $(R[\tilde{s} - t], R[\tilde{e} - t])$ onto the unit sphere. Based on the line distance functions, LDL then scores pose samples R, t in the pool by comparing the function values at uniformly sampled points $Q \subset \mathbb{S}^2$ with the following cost function

$$C^{\mathrm{L}}(R, t) = \sum_i \sum_{q \in Q} \rho(f_{2D}^{\mathrm{L}}(q; L_{2D}^i) - f_{3D}^{\mathrm{L}}(q; L_{3D}^{\sigma(i)}, R, t)), \quad (3)$$

where $\rho(x) = -\mathbb{1}\{|x| < \tau\}$ is a robust cost function and $\sigma(i)$ denotes the permutation used for estimating R (i.e.,

$d_i \in D_{2D}$ is matched to $\tilde{d}_{\sigma(i)} \in D_{3D}$). Here the distance functions are separately computed for clusters of different principal directions as shown in Figure 2b, making the pose search process to better reason about the distribution of lines.

After selecting candidate poses, LDL refines the samples of poses using conventional local feature matching [46] with PnP-RANSAC [13, 31], whereas our approach is fully geometric. We refer the readers to the original paper [28] or the supplementary material for more details.

4. Method

As shown in Figure 1, our localization method operates by first extracting localization inputs (Section 4.1), performing efficient pose search using point and line distance functions (Section 5.2), and refining selected poses by matching line intersections (Section 4.3).

4.1. Input Preparation

In addition to lines and principal directions, we introduce *line intersection points* as input to enhance localization performance. As shown in Figure 2a, we obtain three clusters of line intersection points for both 2D and 3D, namely $\mathcal{P}_{2D}^{cls} = \{P_{2D}^{12}, P_{2D}^{23}, P_{2D}^{31}\}$ and $\mathcal{P}_{3D}^{cls} = \{P_{3D}^{12}, P_{3D}^{23}, P_{3D}^{31}\}$. Here P_{2D}^{ij} is the set of line intersections obtained from L_{2D}^i and L_{2D}^j, and P_{3D}^{ij} is similarly defined. Additional details about input preparation are deferred to the supplementary material.

4.2. Pose Search

4.2.1 Point Distance Functions

We use the additional input of line intersection points to devise point distance functions. As point distance functions contain more fine-grained information of scene keypoints, they can guide towards making more accurate pose search.

Point distance functions are defined in a similar way as line distance functions. Given a set of 2D points P_{2D}, 2D point distance functions are defined over the unit sphere $\mathbb{S}^2$ as the distance to the closest point in P_{2D}, namely

$$f_{2D}^{\mathrm{P}}(x; P_{2D}) = \min_{p \in P_{2D}} d_{\mathrm{P}}^{\gamma}(x, p), \quad (4)$$

where $d_{\mathrm{P}}(x, p) = \cos^{-1}\langle x, p \rangle$ and $\gamma = 0.2$ is a sharpening parameter to incur more dramatic change near the point locations. 3D point distance functions are defined similarly for rotations and translations within the 3D map,

$$f_{3D}^{\mathrm{P}}(x; P_{3D}, R, t) = \min_{\tilde{p} \in P_{3D}} d_{\mathrm{P}}^{\gamma}(x, \Pi(\tilde{p}; R, t)), \quad (5)$$

where $\Pi(\tilde{p}; R, t)$ denotes the spherical projection of the transformed 3D point $R[\tilde{p} - t]$. As in Equation 3, point distance functions can be separately compared for intersection

(a) Line Intersection Extraction

2D Lines

3D Lines   Point Distance Functions   Line Distance Functions
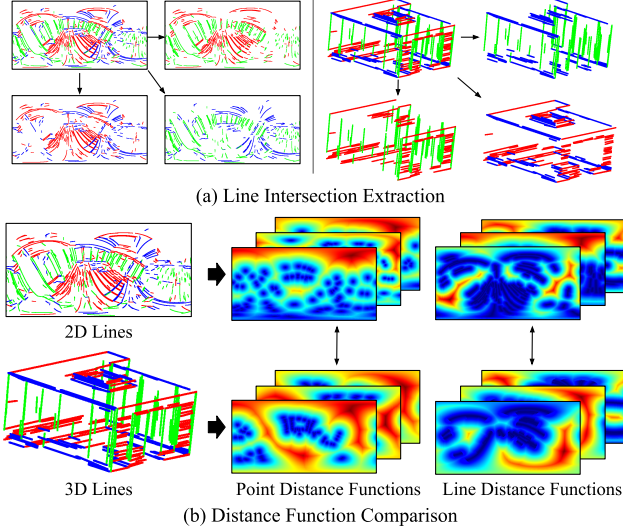
(b) Distance Function Comparison

Figure 2. Line intersection extraction and distance function comparison. (a) We pairwise intersect lines clustered along the principal directions and obtain three groups of intersection points. (b) The intersection point clusters are used to define point distance functions. Together with line distance functions that describe the coarse scene layout, point distance functions describe the fine-grained geometry of lines which enable accurate pose search.

point clusters as follows,

$$
C^{\mathrm{P}}(R,t) = \sum_{i \neq j} \sum_{q \in Q} \rho(f_{2D}^{\mathrm{P}}(q; P_{2D}^{ij}) - f_{3D}^{\mathrm{P}}(q; P_{3D}^{\sigma(ij)}, R, t)),
$$

(6)

where with an abuse of notation $\sigma(ij)$ is the juxtaposition of $\sigma(i)$ and $\sigma(j)$, namely permutations of principal directions for estimating R. As explained in Section 3 and Figure 3a, each rotation in the rotation pool is obtained by combinatorially associating principal directions. Our method then uses point distance functions *along with* line distance functions for pose search,

$$
C(R,t) = C^{\mathrm{L}}(R,t) + C^{\mathrm{P}}(R,t).
$$

(7)

4.2.2 Efficient Distance Function Comparison

Given a pool of $N_t \times N_r$ poses, our method selects top-K poses to refine by comparing Equation 7. However, exhaustive comparison cannot scale to large scenes containing multiple line maps (or rooms) as shown in Figure 3a. The 3D distance function term depends on the rotation pool, which changes for different query images as their 2D principal directions change. Therefore the 3D distance functions have to be evaluated *on-the-fly* for each query image for all line maps in 3D. Below we introduce two strategies for efficient pose search that is scalable to large scale localization.

Pre-computing 3D Distance Functions We observe that rotation from 3D distance functions can be decoupled so

(a) Large Scale Localization Setup & Challenges

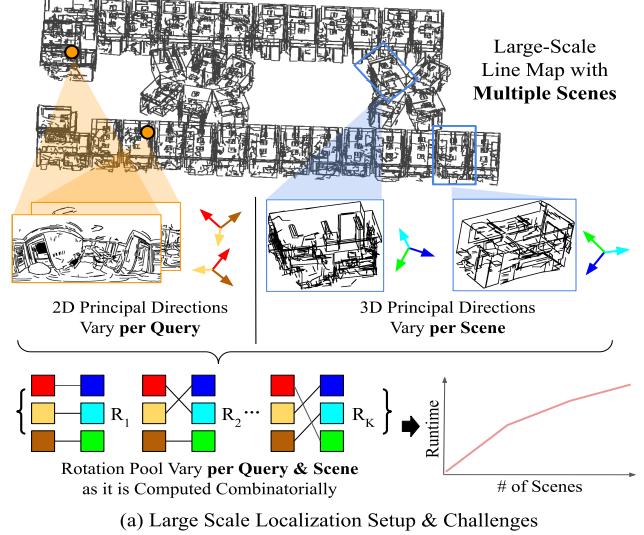(b) Efficient Distance Function Computation

Figure 3. Motivation and overview of efficient distance function comparison. (a) In large-scale localization scenarios, the rotation pool constantly changes due to the variability of principal directions in 2D and 3D. Thus exhaustively computing 3D distance functions for all possible poses on the fly leads to large runtime. (b) We instead propose to (i) decouple translation and rotation, (ii) precompute and cache 3D distance functions aligned in the canonical direction, and (iii) during localization interpolate 2D distance function values at various rotations greatly reducing computation.

that the distance functions only for different translations can be pre-computed and *cached.* Specifically, we propose the following modified cost functions,

$$
C^{\mathrm{L}}(R,t) = \sum_i \sum_{q \in Q} \rho(f_{2D}^{\mathrm{L}}(q; R^T L_{2D}^i) - \underbrace{f_{3D}^{\mathrm{L}}(q; L_{3D}^{\sigma(i)}, I, t)}_{\text{pre-compute \& cached}}),
$$

(8)

$$
C^{\mathrm{P}}(R,t) = \sum_{i \neq j} \sum_{q \in Q} \rho(f_{2D}^{\mathrm{P}}(q; R^T P_{2D}^{ij}) - \underbrace{f_{3D}^{\mathrm{P}}(q; P_{3D}^{\sigma(ij)}, I, t)}_{\text{pre-compute \& cached}}),
$$

(9)

where $R^T L_{2D}^i, R^T P_{2D}^{ij}$ denotes 2D lines and points rotated by R^T respectively.

The proposed formulation indicates applying N_r rotations on 2D distance functions and comparing against the cached 3D distance functions. When pre-computing 3D distance functions, we propose to align the 3D principal directions to a common coordinate frame (i.e., *canonicalize* 3D

lines), as shown in Figure 3b. This alleviates calculating all possible rotations associating 2D-3D principal directions, but focus only on 2D rotations relative to the canonicalized 3D principal directions. Therefore the 2D line distance functions can be computed in a constant runtime, independent to the number of line maps.

Fast 2D Distance Functions via Interpolation We further introduce a method to accelerate 2D distance function extraction. Specifically, we evaluate distance function only once at the identity rotation, and use them to estimate values at N_r rotations as shown in Figure 3b. Since distance functions are defined on a sphere, rotation of the function is equivalent to rotating the query points $q \in Q$ as

$$f_{2D}(q; \hat{R}L_{2D}) = f_{2D}(\hat{R}^T q; L_{2D}) \approx f_{2D}(q^*; L_{2D}), \quad (10)$$

where $q^* = \arg\min_{\hat{q} \in Q} d_P(q, \hat{R}^T \hat{q})$, or the nearest neighbor interpolation.

While the new cost functions and the interpolation scheme can effectively reduce runtime for pose search, they are *approximations* of the original cost functions. Nevertheless, we find that the approximations are sufficiently accurate, due to the following theorem.

Theorem 1. *Given a metric $d(\cdot, \cdot)$ defined over the unit sphere $\mathbb{S}^2$, let $f(x; S) := \min_{s \in S} d(x, s)$ denote a distance function to a set of spherical points $S \subset \mathbb{S}^2$. Consider a countable, finite set of spherical points $Q \subset \mathbb{S}^2$ that satisfy $\max_{q \in Q} \min_{\hat{q} \in Q} d(q, R\hat{q}) \leq \max_{q \in Q} \min_{\hat{q} \neq q} d(q, \hat{q}) = \delta$ for all $R \in SO(3)$. For an arbitrary rotation $\tilde{R} \in SO(3)$, the following inequality holds:*

$$\frac{1}{|Q|} \sum_{q \in Q} |f(q; S) - f(\arg\min_{\hat{q} \in Q} d(\hat{q}, \tilde{R}q); \tilde{R}S)| \leq \delta. \quad (11)$$

We provide detailed proofs to the theorem in the supplementary material. The theorem suggests that for *sufficiently dense* query points, the sum of distance function values remain almost constant under rotation, namely $\sum_{q \in Q} f(q; S) \approx \sum_{q \in Q} f(q; RS)$. As a result, we can substitute the lines and points in the original cost functions with their rotated counterparts: for example for lines,

$$\sum_i \sum_{q \in Q} \rho(f_{2D}^L(q; L_{2D}^i) - f_{3D}^L(q; L_{3D}^{\sigma(i)}, R, t)) \quad (12)$$

$$\approx \sum_i \sum_{q \in Q} \rho(f_{2D}^L(q; R^T L_{2D}^i) - f_{3D}^L(q; L_{3D}^{\sigma(i)}, I, t)), \quad (13)$$

which leads to the modified cost function in Equation 8, 9.

4.3. Pose Refinement

After finding the top-K poses $\{(R_k, t_k)\}$, we can refine them only using geometric representations. Given a good

(a) Translation Refinement from Line Intersection Matching

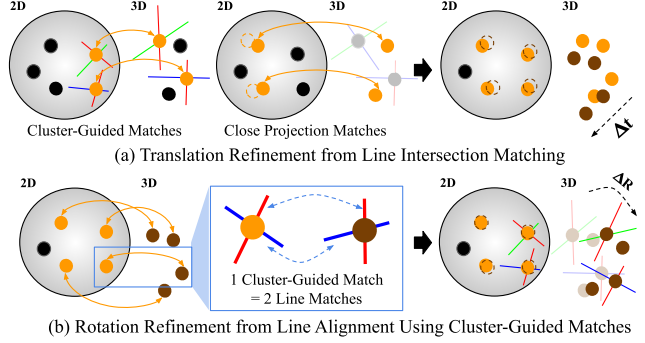(b) Rotation Refinement from Line Alignment Using Cluster-Guided Matches

Figure 4. Pose refinement using line intersections. (a) We first match line intersections that belong to the same cluster type using mutual nearest neighbors (Cluster-Guided Matches), along with a small pair of 2D, 3D intersections that are sufficiently close together on the sphere (Close Projection Matches). The matches are then used to optimize translation, where matches are also iteratively updated similar to ICP [4]. (b) Rotation is then optimized based on the final set of matches by aligning the incident line directions of cluster-guided matches. Here we exploit the fact that each cluster-guided match yields a pair of line matches.

initial pose, we establish correspondences in 2D and 3D from nearest neighbor matching, and refine translation then rotation to find a highly accurate pose.

Line Intersection Matching Our refinement, in a big picture, is similar to the iterative closest point (ICP) [4], where fast and accurate correspondences are crucial. As shown in Figure 4a, we avoid outliers by considering two types of matches, namely intersection points with (i) coherent principal direction clusters, or (ii) sufficiently close projected distances. For the former *cluster-guided matches*, we use the permutation $\sigma(ij)$ for estimating R_k of the candidate pose. We find the mutual nearest neighbor [19] between the corresponding intersection cluster pairs: $P_{2D}^{ij} \subset \mathcal{P}_{2D}^{cls}$ and $P_{3D}^{\sigma(ij)} \subset \mathcal{P}_{3D}^{cls}$. Specifically, we first project 3D intersections onto the sphere $\Pi(R_k[P_{3D}^{\sigma(ij)} - t_k])$ and find matches with P_{2D}^{ij}. For the latter *close projection matches*, we ignore the cluster types and project all 3D intersections onto the sphere. We then retrieve 2D-3D pairs with distances below a threshold $\delta = 0.1$, resulting in an initial set of point matches $\mathcal{M}^P = \{(m_{2D}, m_{3D})\}$.

Translation Refinement We fix rotation estimated from principal directions in Section 3 and only optimize for translation given the set of point matches. Specifically, we minimize the following cost function with respect to translation t_k using gradient descent [24, 29]:

$$C^{trans}(t_k^{(n)}) = \sum_{m \in \mathcal{M}^P} \|m_{2D} - \Pi(R_k[m_{3D} - t_k^{(n)}])\|_1, \quad (14)$$

where $t_k^{(n)}$ is the translation at step n. After each step we also update the set of matches $\mathcal{M}^{\mathrm{P}}$ via nearest neighbor search between P_{2D}^{ij} and $\Pi(R_k[P_{3D}^{\sigma(ij)} - t_k^{(n)}])$. The refined translation $\hat{t}_{k^*}$ with the smallest cost value and its associated matches $\widehat{\mathcal{M}}^{\mathrm{P}}$ are then passed to rotation refinement.

Rotation Refinement Finally, we deviate from intersection points, and refine the rotation using the original line directions. As shown in Figure 4b, we can deduce two pairs of *line matches* from each cluster-guided match in $\widehat{\mathcal{M}}^{\mathrm{P}}$. Let $\widehat{M^{\mathrm{L}}} = \{([s_{2D}, e_{2D}], [\tilde{s}_{3D}, \tilde{e}_{3D}])\}$ denote the set of line matches obtained from $\widehat{\mathcal{M}}^{\mathrm{P}}$. We iteratively minimize the following cost function with respect to rotation R_{k^*},

$$C^{\mathrm{rot}}(R_{k^*}^{(n)}) = \sum_{m \in \widehat{M^{\mathrm{L}}}} |\cos(\frac{s_{2D} \times e_{2D}}{\|s_{2D} \times e_{2D}\|}, \frac{R_{k^*}^{(n)}[\tilde{s}_{3D} - \tilde{e}_{3D}]}{\|\tilde{s}_{3D} - \tilde{e}_{3D}\|})|.$$
$$(15)$$

The cost function aligns the line directions in 2D to the rotated line directions in 3D. Combined with efficient pose search, our sequential pose refinement scheme enables accurate pose estimation without resorting to feature descriptors, where we perform detailed comparisons against feature-based methods in Section 5.

5. Experiments

In this section, we evaluate our method on a wide range of localization scenarios. We use two commonly used datasets [8, 26–28, 58]: Stanford 2D-3D-S [3] and OmniScenes [26]. Stanford 2D-3D-S consists of 1413 panoramas captured in 272 rooms, while OmniScenes is a recently proposed dataset that consists of 4121 panoramas from 7 rooms. Unless specified otherwise, we use the entire Stanford 2D-3D-S dataset and the Extreme split from OmniScenes, which is the most challenging split that contains both scene changes and fast camera motion. For all of scenarios, we use the fixed set of hyperparameters with the inlier threshold $\tau = 0.1$, number of query points $|Q| = 642$, and number of translations per line map $N_t = 500$. Our method is implemented in PyTorch [40], and we use a single RTX2080 GPU with an Intel Core i7-7500U CPU.

5.1. Large-scale Localization

We first provide evaluation in large-scale localization scenarios, where the geometric methods as ours can benefit from the lightweight representations. Here we join multiple room-level line maps in OmniScenes [26] and Stanford 2D-3D-S [3] to create four large-scale maps: *split 1* containing all rooms in OmniScenes, *split 2* containing all rooms in Area 1 from Stanford 2D-3D-S, *split 3* and *split 4* containing 40 and 60 office rooms in Stanford 2D-3D-S respectively. Note we design the latter two splits to examine localization in large number of similar-looking structures. An exemplary large scale map is shown in Figure 1.

Scene No.	1	2	3	4	Search Time (s)	Refine Time (s)	Map Size (GB)
# of Rooms	7	44	40	60			
LDL	0.54	0.26	0.21	0.19	0.24	0.07	129.75
CP + LG	0.66	0.58	0.67	**0.69**	0.23	**0.07**	131.11
CP + GS	0.61	0.58	**0.73**	0.61	0.23	0.21	132.76
CP + GM	0.25	0.27	0.35	0.33	0.23	0.53	2.86
SFRS + LG	0.60	**0.61**	0.66	0.68	0.10	0.07	132.47
SFRS + GS	0.47	0.58	0.60	0.58	0.10	0.21	134.12
SFRS + GM	0.07	0.23	0.31	0.23	0.10	0.53	4.22
Ours	**0.76**	**0.61**	0.68	0.64	**0.02**	0.38	**2.56**

Table 1. Localization evaluation results in large scale scenes. We compare our method against LDL [28] along with combinations of various pose search (Cosplace [5] (CP), SFRS [16]) and refinement (LightGlue [32] (LG), Gluestick (GS), GoMatch (GM)) methods. For each scene, we report the localization accuracy at 0.1m and 5°. We additionally report the average pose search time, refinement time, and map size during localization.
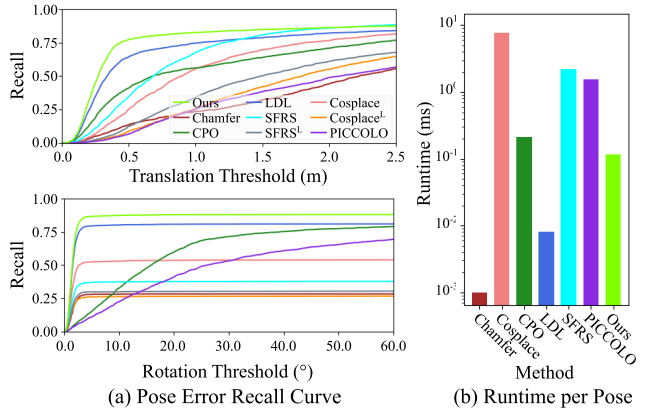

(a) Pose Error Recall Curve (b) Runtime per Pose

Figure 5. Pose error recall and runtime comparison in OmniScenes [26] and Stanford 2D-3D-S [3] for the top-1 retrieval results. Note the runtime is plotted in log scale, and the superscript X^{L} denotes that the baseline takes line images as input.

In Table 1 we compare against LDL [28] and competitive combinations of neural network-based pose search and refinement methods. Note that for baselines other than LDL, we synthesize images by projecting the colored point cloud to extract descriptors, similar to [27, 28]. Here all methods extract top-5 retrieval from pose search, and refine them. Our method largely outperforms LDL and demonstrates competitive performance against neural network-based methods. Further, due to the efficient pose search scheme proposed in Section 5.2, our method exhibits an order-of-magnitude shorter search time compared to the baselines. In addition, as our method only stores the $|Q|$ 3D distance function values, the map size is much smaller than the baselines that cache high-dimensional global/local feature descriptors. We make a detailed analysis of the map size in the supplementary material.

5.2. Pose Search

We make further analysis on the first stage, i.e., pose search, where the task is to quickly find the closest pose from

Method	Accuracy (0.1m, 5°)	Accuracy (0.2m, 10°)	Accuracy (0.3m, 15°)
LDF	0.57	0.58	0.58
LDF + PDF	**0.61**	**0.62**	**0.63**

(a) Pose search using line / point distance functions (LDF / PDFs)

Method	Accuracy (0.1m, 5°)	Runtime (s) @ GPU	Runtime (s) @ CPU
Ours w/o Decoupling	**0.62**	45.049	1908.522
Ours	0.61	**2.890**	**109.561**

(b) Decoupling rotation and translation for 3D distance functions

Interp.	Canonical.	Accuracy (0.1m, 5°)	Runtime (s) @ GPU	Runtime (s) @ CPU
✗	✗	**0.61**	0.396	7.243
✗	✓	**0.61**	0.007	0.198
✓	✓	**0.61**	**0.004**	**0.080**

(c) Interpolation and line canonicalization for 2D distance functions

Table 2. Ablation study of our pose search method, using the entire Area 1 of Stanford 2D-3D-S as the 3D map. Note the runtime for distance functions indicates the time needed to generate *all* 2D or 3D distance functions for rooms in the map.

pools of rotations and translations in a 3D map. The compared baselines are categorized into neural network-based (SFRS [16], Cosplace [5]); color distribution-based (PIC-COLO [26], CPO [27]); and line-based (LDL [28], Chamfer [38]) methods. We use the colored point cloud from the tested datasets and render synthetic views at various translations and rotations, from which we prepare neural network descriptors for the baselines. For fair comparison, we use the identical pool of translations and rotations. We additionally implement neural network-based methods that take line images as input instead of the original RGB images (SFRS[L], Cosplace[L]), to examine how these methods can handle geometric localization scenarios.

Figure 5 shows the top-1 recall curves and runtimes for generating a global descriptor per pose. Our method attains a high recall even at strict translation and rotation thresholds, and shows a much shorter runtime than neural network-based or color-based methods. While several line-based methods (LDL, Chamfer) exhibit a shorter runtime, the retrieved poses from there methods are relatively inaccurate compared to our method.

Ablation Study We ablate the key constituents of the fast and accurate pose search pipeline, namely the point distance functions and the efficient distance function computation. Table 2 reports the localization error and runtime for top-5 retrieval using the entire Area 1 of Stanford 2D-3D-S [3]. Table 2a suggests that using point distance functions along with line distance functions enhance localization performance. While the LDF-only method is similar to LDL, we can still observe a performance gap compared to Table 1. Due to the efficient distance function comparison, our method can seamlessly handle large number ($|Q| = 642$) of query points during pose search, whereas LDL can only handle a limited number of query points ($|Q| = 42$). We

Refinement Type	Method	Visual Desc.	Accuracy (0.1m, 5°)	Accuracy (0.2m, 10°)	Accuracy (0.3m, 15°)
Line-Based Refinement	Line Transformer [56]	○	0.76	0.79	0.80
	Gluestick [42]	○	0.75	0.79	0.80
Point-Based Refinement	SuperGlue [46]	○	0.78	**0.80**	**0.81**
	LightGlue [32]	○	**0.79**	**0.80**	**0.81**
	LoFTR [51]	○	0.74	0.78	0.78
	PICCOLO [26]	○	0.57	0.59	0.60
	CPO [27]	○	0.62	0.64	0.65
Geometric Refinement	GoMatch [61]	✗	0.61	0.72	0.74
	BPnPNet [9]	✗	0.04	0.22	0.41
	Gao et. al [14]	✗	0.21	0.56	0.66
	PDF Minimization	✗	0.22	0.38	0.47
	LoFTR[L] [51]	✗	0.09	0.31	0.50
	SuperGlue[L] [46]	✗	0.06	0.34	0.57
	LightGlue[L] [32]	✗	0.37	0.57	0.67
	GlueStick[L] [42]	✗	0.32	0.51	0.62
	Line Transformer[L] [56]	✗	0.06	0.29	0.49
	Ours	✗	0.67	0.74	0.76

Table 3. Pose refinement evaluation in OmniScenes [26] and Stanford 2D-3D-S [3]. Note the superscript X[L] denotes that the baseline takes line images as input.

Method	t-error	R-error	Accuracy (0.1m, 5°)
Only trans. refine	**0.06**	1.56	**0.78**
No intersection clusters	0.08	1.11	0.63
Ours	**0.06**	**1.05**	0.77

Table 4. Ablation study of key components of our pose refinement method, using the Room 2 subset from OmniScenes [26].

provide additional experiments regarding this aspect in the supplementary material. Table 2b and 2c further support this claim: decoupling rotation/translation for 3D distance functions and interpolating and canonicalizing 2D distance functions lead to significant runtime drops, while showing almost no loss in accuracy.

5.3. Pose Refinement

To evaluate the second stage of pose refinement, we retrieve the same top-1 pose using our pose search scheme, and perform refinement using various methods. We compare the accuracy against three classes of refinement methods: line-based [42, 56], point-based [26, 27, 32, 46, 51], and geometric [9, 14, 61]. While line and point-based methods find matches using the photometric information near the lines and points to increase accuracy, geometric methods only rely on the locations of keypoints (GoMatch [61], BP-nPNet [9]) or lines, as we propose. We also implement the line alignment method proposed from Gao et al. [14], along with a conceived baseline that directly minimizes Equation 6 with gradient descent (PDF minimization).

Table 3 shows that our method consistently outperforms all the tested geometric methods, while showing competitive performance against methods that use visual descriptors. Considering the minimal map size (Table 1), it is a noticeable performance improvement of purely geometric approaches.

Method	Visual Desc.	Orig.	Intensity	Gamma	White Balance	Range	Std
Line Transformer [ ]	○	0.88	0.70	0.86	0.89	0.22	0.09
Gluestick	○	**0.89**	**0.80**	**0.89**	**0.90**	0.12	0.05
GoMatch	✗	0.67	0.66	0.63	0.67	0.08	0.03
Gluestick^L	✗	0.36	0.49	0.37	0.31	0.22	0.09
Ours	✗	0.77	0.74	0.74	0.77	**0.05**	**0.02**

Table 5. Pose refinement evaluation under varying lighting conditions in OmniScenes [26]. We report the localization accuracy at 0.1m and 5°, along with their range and standard deviations.
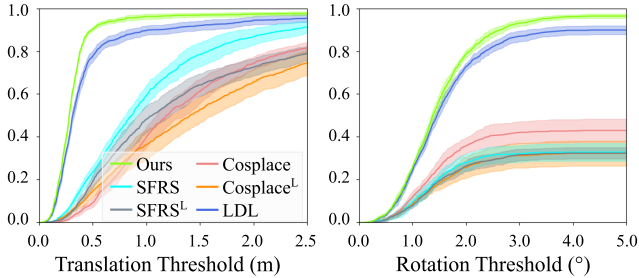
Figure 6. Recall curves of top-1 pose search amidst lighting condition variations. We evaluate localization performance on six lighting variations and shade the maximum and minimum recall. The solid lines are the averaged recall values for each tested baseline.

Ablation Study Our pose refinement scheme deploys line intersections in the place of conventional visual descriptors and effectively disambiguates matches from point clusters. Table 4 displays the effect of those components in terms of localization accuracy and the median translation/rotation errors using Room 2 from OmniScenes [26]. Optimizing only translation leads to larger rotation errors, which suggests the importance of rotation refinement. In addition, directly performing nearest neighbor matching without using intersection clusters lead to a large drop in localization performance. As intersection clusters prevent outlier matches, they are crucial for accurate pose refinement.

5.4. Robustness Evaluation

Illumination Changes Our method can stably perform localization in various environment conditions as long as we can extract lines. We demonstrate the robustness of our method amidst lighting changes using the OmniScenes [26] dataset. Specifically, we separately measure the performance of pose search and refinement after applying synthetic color variations to the input panorama, similar to the experiments conducted in LDL [28]. We consider three types of color variations (intensity, gamma, and white balance), where we apply two levels of variations for each type.

Table 5 displays the pose refinement performance at varying lighting conditions, where all the tested methods take the top-1 pose search results using our method as input. Our refinement scheme performs competitively against visual descriptor-based methods, while showing a much smaller range and standard deviations in localization accuracy. A similar trend is observable in Figure 6 that shows

Method	Additional Input	Estimation Target	Accuracy (0.1m, 5°)	Accuracy (1m, 30°)
LaLaLoc [22]	Layout Depth	2D Trans.	0.91	0.95
LaLaLoc [22]	N/A	2D Trans.	0.58	0.88
LaLaLoc++ [21]	N/A	2D Trans.	0.72	0.92
Laser2D [39]	N/A	2D Trans. + Rot.	0.79	0.95
Ours	Layout Lines	3D Trans. + Rot.	**0.95**	**0.96**

Table 6. Localization evaluation using lines from indoor floorplans in Structured3D [60]. Note for baselines that only estimate translation, we report the accuracy using only the translation threshold.

the recall curves of top-1 retrievals from various pose search methods. The width of the shaded regions, which indicates the range of recall values under illumination changes, is much smaller for our method compared to the baselines. By using lines and their intersections as the only cues for localization, our method remains robust against lighting changes.

Applicability to Floorplan Localization While our method is originally designed for localizing against 3D line maps obtained from point cloud scans or structure from motion, we find that our fully geometric setup is versatile to handle 3D floorplan maps without any hyperparameter changes. We test our method on localizing panoramas against 3D lines extracted from a floorplan map [21, 22, 39] using the Structured3D [60] dataset. Instead of simply extracting lines from panoramas, we use the lines from 2D layout annotations [60] to be compatible with the 3D floorplan maps. Table 6 shows that our method can outperform existing methods. Further, unlike prior works that only estimate the 2D translation and rotation, our method can estimate the full 6DoF pose while attaining competitive accuracy. Therefore, if combined with sufficiently accurate 2D layout extraction from panoramas (removing the demand for 2D layout annotations), we expect our method to perform practical floorplan localization.

6. Conclusion

In this paper, we introduced an accurate and lightweight pipeline for fully geometric panoramic localization. Our method is solely based on the geometry of lines, and thus can offer privacy protection while using a much smaller map size than methods using visual descriptors. To effectively utilize the otherwise ambiguous geometric entities, we propose point distance functions along with an efficient comparison scheme for pose search and use principal directions of lines to match their intersections during refinement. Due to the lightweight formulation, our method can perform scalable localization in large scenes and attain robustness in lighting changes. We thus expect our method to serve as a practical pipeline for fully geometric localization.

References

[1] Hichem Abdellali, Robert Frohlich, Viktor Vilagos, and Zoltan Kato. L2d2: Learnable line detector and descriptor. In *2021 International Conference on 3D Vision (3DV)*, pages 442–452, 2021. 2

[2] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 1

[3] Iro Armeni, Sasha Sax, Amir R Zamir, and Silvio Savarese. Joint 2d-3d-semantic data for indoor scene understanding. *arXiv preprint arXiv:1702.01105*, 2017. 6, 7

[4] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(5):698–700, 1987. 5

[5] Gabriele Berton, Carlo Masone, and Barbara Caputo. Rethinking visual geo-localization for large-scale applications. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4878–4888, 2022. 6, 7

[6] Christian Cachin, Idit Keidar, and Alexander Shraer. Trusting the cloud. *SIGACT News*, 40(2):81–86, 2009. 2

[7] Dylan Campbell, Lars Petersson, Laurent Kneip, and Hongdong Li. Globally-optimal inlier set maximisation for camera pose and correspondence estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page preprint, 2018. 2

[8] Dylan Campbell, Lars Petersson, Laurent Kneip, Hongdong Li, and Stephen Gould. The alignment of the spheres: Globally-optimal spherical mixture alignment for camera pose estimation. In *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page to appear, Long Beach, USA, 2019. IEEE. 1, 2, 6

[9] Dylan Campbell, Liu Liu, and Stephen Gould. Solving the blind perspective-n-point problem end-to-end with robust differentiable geometric optimization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, page preprint. Springer, 2020. 2, 7

[10] Deeksha Dangwal, Vincent T. Lee, Hyo Jin Kim, Tianwei Shen, Meghan Cowan, Rajvi Shah, Caroline Trippel, Brandon Reagen, Timothy Sherwood, Vasileios Balntas, Armin Alaghi, and Eddy Ilg. Analysis and mitigations of reverse engineering attacks on local feature descriptors, 2021. 2

[11] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPR Deep Learning for Visual SLAM Workshop*, 2018. 1

[12] Mihai Dusmanu, Johannes Schönberger, Sudipta Sinha, and Marc Pollefeys. Privacy-preserving image features via adversarial affine subspace embeddings. In *Computer Vision and Pattern Recognition (CVPR 2021)*. CVF/IEEE, 2021. 1, 2

[13] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981. 3

[14] Shuang Gao, Jixiang Wan, Yishan Ping, Xudong Zhang, Shuzhou Dong, Jijunnan Li, and Yandong Guo. Pose refinement with joint optimization of visual points and lines. *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2888–2894, 2021. 2, 7

[15] Shuang Gao, Jixiang Wan, Yishan Ping, Xudong Zhang, Shuzhou Dong, Jijunnan Li, and Yandong Guo. Pose refinement with joint optimization of visual points and lines. *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2888–2894, 2021. 2

[16] Yixiao Ge, Haibo Wang, Feng Zhu, Rui Zhao, and Hongsheng Li. Self-supervising fine-grained region similarities for large-scale image localization. In *European Conference on Computer Vision*, 2020. 6, 7

[17] Rafael Grompone von Gioi, Jeremie Jakubowicz, Jean-Michel Morel, and Gregory Randall. Lsd: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(4): 722–732, 2010. 3

[18] Zirui Guo, Huimin Lu, Qinghua Yu, Ruibin Guo, Junhao Xiao, and Hongshan Yu. Hdpl: a hybrid descriptor for points and lines based on graph neural networks. *Ind. Robot*, 48: 737–744, 2021. 2

[19] Laleh Haghverdi, Aaron Lun, Michael Morgan, and John Marioni. Batch effects in single-cell rna-sequencing data are corrected by matching mutual nearest neighbors. *Nature biotechnology*, 36, 2018. 5

[20] Manuel Hofer, Michael Maurer, and Horst Bischof. Efficient 3d scene abstraction using line segments. *Computer Vision and Image Understanding*, 157:167–178, 2017. Large-Scale 3D Modeling of Urban Indoor or Outdoor Scenes from Images and Range Scans. 2, 3

[21] Henry Howard-Jenkins and Victor Adrian Prisacariu. Lalaloc++: Global floor plan comprehension for layout localisation in unvisited environments. 2022. 8

[22] Henry Howard-Jenkins, Jose-Raul Ruiz-Sarmiento, and Victor Adrian Prisacariu. Lalaloc: Latent layout localisation in dynamic, unvisited environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10107–10116, 2021. 8

[23] W. Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 32(5):922–923, 1976. 3

[24] J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *Ann. Math. Statist.*, 23(3): 462–466, 1952. 5

[25] Hyunwoo Kim and Sukhan Lee. A novel line matching method based on intersection context. In *2010 IEEE International Conference on Robotics and Automation*, pages 1014–1021, 2010. 2

[26] Junho Kim, Changwoon Choi, Hojun Jang, and Young Min Kim. Piccolo: Point cloud-centric omnidirectional localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3313–3323, 2021. 1, 6, 7, 8

[27] Junho Kim, Hojun Jang, Changwoon Choi, and Young Min Kim. Cpo: Change robust panorama to point cloud localization. ECCV, 2022. 6, 7

[28] Junho Kim, Changwoon Choi, Hojun Jang, and Young Min Kim. Ldl: Line distance functions for panoramic localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 17882–17892, 2023. 1, 2, 3, 6, 7, 8

[29] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 5

[30] Manuel Lange, Fabian Schweinfurth, and Andreas Schilling. DLD: A Deep Learning Based Line Descriptor for Line Feature Matching. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5910–5915. IEEE, 2019. 2

[31] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o(n) solution to the pnp problem. *International Journal Of Computer Vision*, 81:155–166, 2009. 3

[32] Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. LightGlue: Local Feature Matching at Light Speed. In *ICCV*, 2023. 2, 6, 7

[33] Shaohui Liu, Yifan Yu, Rémi Pautrat, Marc Pollefeys, and Viktor Larsson. 3d line mapping revisited. In *Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[34] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004. 1

[35] QuanMeng Ma, Guang Jiang, and DianZhi Lai. Robust line segments matching via graph convolution networks. *ArXiv*, abs/2004.04993, 2020. 2

[36] Quanmeng Ma, Guang Jiang, Jiajie Wu, Changshuai Cai, Dianzhi Lai, Zixuan Bai, and Hao Chen. Wglsm: An end-to-end line matching network based on graph convolution. *Neurocomput.*, 453(C):195–208, 2021. 2

[37] Branislav Micusik and Horst Wildenauer. Structure from motion with line segments under relaxed endpoint constraints. In *2014 2nd International Conference on 3D Vision*, pages 13–19, 2014. 2

[38] Branislav Micusik and Horst Wildenauer. Descriptor free visual indoor localization with line segments. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3165–3173, 2015. 2, 7

[39] Zhixiang Min, Naji Khosravan, Zachary Bessinger, Manjunath Narayana, Sing Bing Kang, Enrique Dunn, and Ivaylo Boyadzhiev. Laser: Latent space rendering for 2d visual localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11122–11131, 2022. 8

[40] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library.

In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 6

[41] Rémi Pautrat*, Juan-Ting Lin*, Viktor Larsson, Martin R. Oswald, and Marc Pollefeys. Sold2: Self-supervised occlusion-aware line description and detection. In *Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[42] Rémi Pautrat, Iago Suárez, Yifan Yu, Marc Pollefeys, and Viktor Larsson. GlueStick: Robust image matching by sticking points and lines together. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. 2, 7

[43] Francesco Pittaluga, Sanjeev J Koppal, Sing Bing Kang, and Sudipta N Sinha. Revealing scenes by inverting structure from motion reconstructions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 145–154, 2019. 1

[44] Yohann Salaün, Renaud Marlet, and Pascal Monasse. Robust and accurate line- and/or point-based pose estimation without manhattan assumptions. In *European Conference on Computer Vision*, 2016. 2

[45] Y. Salaun, R. Marlet, and P. Monasse. Line-based robust sfm with little image overlap. In *2017 International Conference on 3D Vision (3DV)*, pages 195–204, Los Alamitos, CA, USA, 2017. IEEE Computer Society. 2

[46] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 2, 3, 7

[47] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[48] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 1

[49] Pablo Speciale, Johannes L. Schönberger, Sing Bing Kang, Sudipta N. Sinha, and Marc Pollefeys. Privacy preserving image-based localization. 2019. 2

[50] Pablo Speciale, Johannes Schönberger, Sudipta Sinha, and Marc Pollefeys. Privacy preserving image queries for camera localization. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1486–1496, 2019. 1, 2

[51] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-free local feature matching with transformers. *CVPR*, 2021. 7

[52] Hajime Taira, Masatoshi Okutomi, Torsten Sattler, Mircea Cimpoi, Marc Pollefeys, Josef Sivic, Tomas Pajdla, and Akihiko Torii. InLoc: Indoor Visual Localization with Dense Matching and View Synthesis. In *CVPR 2018 - IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, United States, 2018. 1

[53] Felix Taubner, Florian Tschopp, Tonci Novkovic, Roland Siegwart, and Fadri Furrer. Lcd – line clustering and description for place recognition, 2020. 2

[54] Shuzhe Wang, Juho Kannala, and Daniel Barath. Dgcgnn: Descriptor-free geometric-color graph neural network for 2d-3d matching, 2023. 2

[55] Lu Xiaohu, Liu Yahui, and Li Kai. Fast 3d line segment detection from unorganized point cloud. *arXiv preprint arXiv:1901.02532*, 2019. 3

[56] Sungho Yoon and Ayoung Kim. Line as a visual sentence: Context-aware line descriptor for visual localization. *IEEE Robotics and Automation Letters*, 6(4):8726–8733, 2021. 2, 7

[57] Xin Yu, Sagar Chaturvedi, Chen Feng, Yuichi Taguchi, Teng-Yok Lee, Clinton Fernandes, and Srikumar Ramalingam. Vlase: Vehicle localization by aggregating semantic edges. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3196–3203, 2018. 2

[58] Chao Zhang, Ignas Budvytis, Stephan Liwicki, and Roberto Cipolla. Rotation equivariant orientation estimation for omnidirectional localization. In *ACCV*, 2020. 1, 6

[59] Lilian Zhang and Reinhard Koch. An efficient and robust line segment matching approach based on lbd descriptor and pairwise geometric consistency. *Journal of Visual Communication and Image Representation*, 24(7):794–805, 2013. 2

[60] Jia Zheng, Junfei Zhang, Jing Li, Rui Tang, Shenghua Gao, and Zihan Zhou. Structured3d: A large photo-realistic dataset for structured 3d modeling. In *Proceedings of The European Conference on Computer Vision (ECCV)*, 2020. 8

[61] Qunjie Zhou, Sérgio Agostinho, Aljoša Ošep, and Laura Leal-Taixé. Is geometry enough for matching in visual localization? In *Computer Vision – ECCV 2022*, pages 407–425, Cham, 2022. Springer Nature Switzerland. 1, 2, 7