PlückerNet: Learn to Register 3D Line Reconstructions

Liu Liu, Hongdong Li, Haodong Yao and Ruyi Zha
Australian National University, Canberra, Australia

Liu.Liu@anu.edu.au

Abstract

Aligning two partially-overlapped 3D line reconstructions in Euclidean space is challenging, as we need to simultaneously solve correspondences and relative pose between line reconstructions. This paper proposes a neural network based method and it has three modules connected in sequence: (i) a Multilayer Perceptron (MLP) based network takes Plücker representations of lines as inputs, to extract discriminative line-wise features and matchabilities (how likely each line is going to have a match), (ii) an Optimal Transport (OT) layer takes two-view line-wise features and matchabilities as inputs to estimate a 2D joint probability matrix, with each item describes the matchness of a line pair, and (iii) line pairs with Top-K matching probabilities are fed to a 2-line minimal solver in a RANSAC framework to estimate a six Degree-of-Freedom (6-DoF) rigid transformation. Experiments on both indoor and outdoor datasets show that registration (rotation and translation) precision of our method outperforms baselines significantly.

1. Introduction

Lines contain strong structural geometry information of environments (even for texture-less indoor scenes), and are widely used in many applications, *e.g.*, SLAM [47, 64], visual servoing [3, 30], event camera [50, 38, 39], image restoration [43, 36, 37], place recognition [48, 27] and camera pose estimation [28, 24]. 3D lines can be obtained from structure from motion [49], SLAM [61] or laser scanning [31]. Compared with 3D points, scene represented by lines is more complete and requires significantly less amount of storage [22, 18, 59]. Given 3D line reconstructions, a fundamental problem is how to register them (Figure 1). This technique can be used in building a complete 3D map, robot localization, SLAM, *etc*.

This paper studies the problem of aligning two partially-overlapped 3D line reconstructions in Euclidean space. This is not doable for traditional methods as it's very hard to find line matches by only checking 3D line coordinates, often
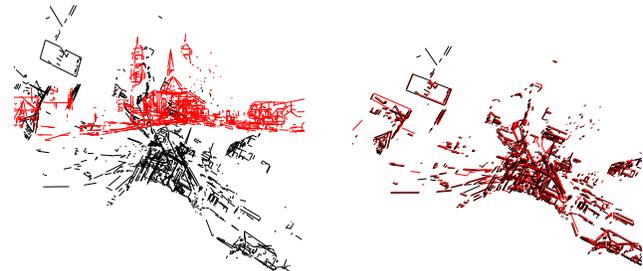

Figure 1: *Our problem is to align two partially-overlapped line reconstructions or, equivalently, to estimate the relative pose between two line reconstructions. **Left**: Red and Black lines (depicting street-view buildings and landmarks) are in two different coordinate systems from the Semantic3D dataset [14]. **Right**: our method is able to successfully align the two line reconstructions in a one-shot manner.*

one needs to manually set line matches [6, 5] or assumes lines are mostly located on planes [22] and windows [9]. With deep neural networks, we give a learning-based solution, dubbed as PlückerNet.

It is non-trivial to learn from lines, as we need to carefully handle line parameterization and geometry. For example, local structure defined by geometric nearest neighbor is a core-component in point-based networks (*e.g.*, PointNet++ [42]). However, for line-based network, defining geometric nearest neighbor is non-trivial as there is no universally agreed error metric for comparing lines [6].

We parameterize a 3D line using a deterministic 6-dim Plücker [41] representation with a 3-dim direction vector lying on a 3D unit hemisphere and a 3-dim moment vector. To capture local line structure, for each line, we first extract local features in the subspace of direction and moment, and then combine them to obtain a global high-dim line feature. To make line-wise features discriminative for matching, we use a graph neural network with attention mechanism [44, 45], as it can integrate contextual cues considering high-dim feature embedding relationships.

As we are addressing a partial-to-partial matching problem, lines do not necessarily have to match. We model the likelihood that a given line has a match

and is not an outlier by regressing line-wise prior matchability. Combined with line-wise features from two line reconstructions, we are able to estimate line correspondences in a global feature matching module. This module computes a weighting (joint probability) matrix using optimal transport, where each element describes the matchability of a particular source line with a particular target line. Sorting the line matches in decreasing order by weight produces a prioritized match list, which can be used to recover the 6-DoF relative pose between source and target line reconstructions.

With line matches are found, we develop a 2-line minimal pose solver ([5]) to solve for the relative pose in Euclidean space. We further show how to integrate the solver within a RANSAC framework using a score function to disambiguate inliers from outliers.

Our PlückerNet is trained end-to-end. The overall framework is illustrated in Figure 2. Our contributions are:

1. A simple, straightforward and effective learning-based method to estimate a rigid transformation aligning two line reconstructions in Euclidean space;

2. A deep neural network extracting features from lines, while respecting the line geometry;

3. An original global feature matching network based on the optimal transport theory to find line correspondences;

4. A 2-line minimal solver with RANSAC to register 3D line reconstructions in Euclidean space;

5. We propose two 3D line registration baselines (iterative closet lines and direct regression), three benchmark datasets build upon [63, 14, 1] and show the state-of-the-art performance of our method.

2. Related Work

For space reasons, we focus on geometric deep learning and aligning line reconstructions, hence omitting a large body of works on 2D line detection and description from images [53, 2, 62, 25, 65, 60], and 3D line fitting [18, 59, 55]. Interested readers can refer to these papers for details.

Learning from unordered sets. PointNet-based [42] networks can handle sparse and unordered 3D points. Though most works focus on classification and segmentation tasks [57, 42], geometry problems are ready to be addressed. For example, 3D–3D point cloud registration [4], 2D–2D [33] or 2D–3D [12] outlier correspondence rejection, and 2D–3D camera pose estimation [26, 7]. Instead of studying 3D points, this paper tackles the problem of learning from lines, specifically, aligning line reconstructions, which has not previously been researched.
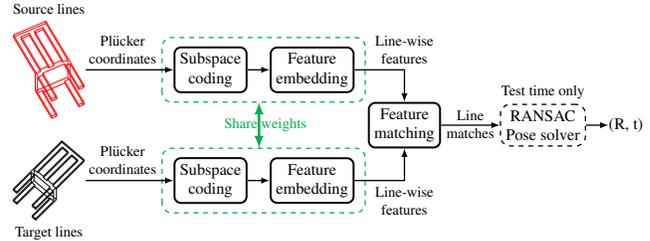

Figure 2: *Overall pipeline of our method. First, lines are represented as 6-dim Plücker coordinates, passed into a Siamese network to extract line-wise deep features via subspace coding and discriminative feature embedding (Section 3.2). Then a global feature matching module estimates line matches from these features using an optimal transport (OT) technique [52, 11, 10] (Section 3.3). Finally, at test time, apart from automatically recovering line correspondences, the 6-DoF relative pose ($\mathbf{R}$ and $\mathbf{t}$) between two coordinate systems is recovered via a 2-line minimal solver with RANSAC (Section 3.4).*

Sparse feature matching. Two sparse feature sets can be matched via nearest neighbor search with ratio test [29] or Hungarian algorithm [23]. Recently, researchers [26, 7, 44, 45] bring the Sinkhorn's algorithm [11] to this matching task for its end-to-end training ability. Though [7, 44, 45, 13] show they can match sparse feature sets, their usage only focuses on pairwise feature distance or matching cost. In contrast, we formulate this matching problem in line with traditional optimal transport [52, 11, 10] inside a probability framework, which aims to align two 1D probability distributions by estimating a 2D probabilistic coupling matrix. We directly regress these two 1D probability distributions using a multilayer perceptron (MLP) based network, and use a variant of Sinkhorn's algorithm [46, 32] to calculate a 2D probability matrix, describing the matchability of these two sparse line sets.

Relative pose estimation. Iff with ground-truth matches, traditional methods [6, 5] can be utilized to estimate a relative pose between two line reconstructions. Bartoli *et al*. [6] present a family of line motion matrix in projective, affine, similarity, and Euclidean space. Minimal 7 line matches are required to solve the line motion matrix linearly. They [5] further extend this method, and derives a minimal 5 line matches method. They also give a minimal 2 line matches solver for solving the line motion matrix in similarity space. This paper aligns line reconstructions in Euclidean space with unknown line-to-line matches.

3. PlückerNet

In this section we present PlückerNet – our method for aligning 3D line reconstructions. We first define the problem in Section 3.1, then describe our pipeline. Specifically, We present our method for extracting line-wise discriminative features in Section 3.2. We then describe our

global feature matching method for obtaining 3D–3D line match probabilities in Section 3.3. Finally, we provide the relative pose solver, a 2-line minimal solver in a RANSAC framework in Section 3.4.

3.1. Problem Definition

Plücker line. A line ℓ in a 3D space has 4 degrees of freedom. Usually, we have three ways to denote a 3D line: 1) a direction and a point the line passes through; 2) two points the line passes through [1], *i.e.* line segment with two start and end junction points; 3) the Plücker coordinates $(\mathbf{v}, \mathbf{m})$ [17], where $\mathbf{v}$ is the 3-dim direction vector and $\mathbf{m}$ is the 3-dim moment vector. We choose Plücker coordinates to parameterize a 3D line for its uniqueness once fixing its homogeneous freedom, and its mathematical completeness. We fix homogeneous freedom of a 6-dim Plücker line by first L_2 normalizing its direction vector $\mathbf{v}$ to a unit-sphere, and then set the value of the first dimension of $\mathbf{v}$ to be greater than 0. This ensures $\mathbf{v}$ lie on a hemisphere.

Partial-to-Partial registration. Let $L_{\mathcal{S}} = \{\ell_i\}, i = 1, ..., M$ denote a 3D line set with M Plücker lines in the source frame, $L_{\mathcal{T}} = \{\ell_j'\}, j = 1, ..., N$ denote a 3D line set with N Plücker lines in the target frame, and $\mathbf{C} \in \mathbb{R}^{M \times N}$ denote the correspondence matrix between $L_{\mathcal{S}}$ and $L_{\mathcal{T}}$.

We aim to estimate a rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{t}$ which transforms source line set $L_{\mathcal{S}}$ to align with the target line set $L_{\mathcal{T}}$. Specifically, $\ell_j' \approx \mathbf{T}\ell_i$ for $\mathbf{C}_{ij} = 1$, where the line motion matrix $\mathbf{T}$ [6] is given by:

$$\mathbf{T} = \begin{pmatrix} \mathbf{R} & [\mathbf{t}]_\times \mathbf{R} \\ 0 & \mathbf{R} \end{pmatrix}, \quad (1)$$

where $[\mathbf{t}]_\times$ is a skew-symmetric matrix.

The difficulty of this registration problem is to estimate the correspondence matrix $\mathbf{C}$. We propose to estimate $\mathbf{C}$ using a deep neural network. Specifically, for each tentative line match in $\mathbf{C}$, we calculate a weight $\mathbf{W}_{ij}$ describing the matchability of ℓ_i and ℓ_j'. We can obtain a set of line matches by taking the Top-K matches according to these weights. With line matches, we give a minimal 2-line method with RANSAC to solve the problem.

3.2. Feature extraction

Basic block. Our input is a set of unordered lines. Inpired by PointNet [42] which consumes a set of unordered points, we also use MLP blocks to extract line-wise features. An MLP block, *e.g.*, MLP(128, 128) denotes a two layers perceptron, with each layer size being 128. In our network, Groupnorm [58] with GeLU [16] is used for all layers (except the last layer) in an MLP block. We

found Groupnorm [58] and GeLU [16] is better than the commonly used Batchnorm [20] and ReLU, respectively.

Subspace coding. For a 6-dim Plücker line, its direction $\mathbf{v}$ and moment $\mathbf{m}$ lies in two domains. $\mathbf{v}$ lies on a hemisphere and $\mathbf{m}$ is unbounded with constraint $\mathbf{v} \cdot \mathbf{m} = 0$. $\|\mathbf{m}\|_2$ gives the distance from the origin to the line (iff $\mathbf{v}$ is L_2-normalized). To bridge this domain gap, we first use two parallel networks without sharing weights to process $\mathbf{v}$ and $\mathbf{m}$ independently, and then concatenate features from them to embed a Plücker line to a high-dim space [2]. The benefits of this subspace coding process lie in two parts: 1) the domain gap between $\mathbf{v}$ and $\mathbf{m}$ is explicitly considered; 2) we are able to define geometric nearest neighbors in each subspace of direction $\mathbf{v}$ and moment $\mathbf{m}$. Angular distance and L_2 distance is used to find geometric nearest neighbors in the subspace of $\mathbf{v}$ and $\mathbf{m}$, respectively. Note that we cannot define nearest neighbors in 6-dim space, as there is no universally agreed global distance metric between two Plücker lines, see discussions in Section 3.4.

For each $\mathbf{o}_i$ ($\mathbf{o}_i$ can be $\mathbf{v}_i$ or $\mathbf{m}_i$, and i is a line index) in a subspace, we first perform nearest neighbor search and build a line-wise Knn graph. For a Knn graph around the anchor $\mathbf{o}_i$, the edges from $\mathbf{o}_i$ to its neighbors capture the local geometric structure. Similar to EdgeConv [57], we concatenate the anchor and edge features, and then pass them through an MLP layer to extract local features around the i-th line. Specifically, the operation is defined by

$$E(\mathbf{o}_i) = \text{avg}_{\mathbf{o}_k, \mathbf{o}_k \in \mho(\mathbf{o}_i)} \left(\boldsymbol{\theta} \left(\mathbf{o}_k - \mathbf{o}_i \right) + \boldsymbol{\phi}\mathbf{o}_i \right), \quad (2)$$

where $\mathbf{o}_k \in \mho(\mathbf{o}_i)$ denotes that $\mathbf{o}_k$ is in the neighborhood $\mho(\mathbf{o}_i)$ of $\mathbf{o}_i$, $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are MLP weights performed on the edge $(\mathbf{o}_k - \mathbf{o}_i)$ and anchor $\mathbf{o}_i$ respectively, and $\text{avg}(\cdot)$ denotes that we perform average pooling in the neighborhood $\mho(\mathbf{o}_i)$ after the MLP to extract a single feature for the i-th line.

After extracting line-wise local features $E(\mathbf{v}_i)$ and $E(\mathbf{m}_i)$ in the subspace of $\mathbf{v}$ and $\mathbf{m}$ independently, we lift them to high-dim spaces by using an MLP block MLP(8, 16, 32, 64). Output features from the two subspaces are concatenated, and further processed by an MLP block MLP(128, 128, 128) to embed to a 128-dim space. This subspace coding process is given in Figure 3.

Discriminative feature embedding. After coding 6-dim Plücker lines to 128-dim features, we aim to make these features discriminative for matching. Inspired by the success of attention mechanism and graph neural network in 3D–3D points cloud registration [56, 13, 45] and two-view image matching [44], we adopt both self-attention and cross-attention to extract line-wise discriminative features.

---

[1] Extracting endpoints of lines accurately and reliably is difficult due to viewpoint changes and occlusions [61, 25].

[2] For features from the two subspace, we find no additional benefit of imposing the orthogonal constraint via the Gram-Schmidt process, though the original direction $\mathbf{v}$ is orthogonal to the moment $\mathbf{m}$.
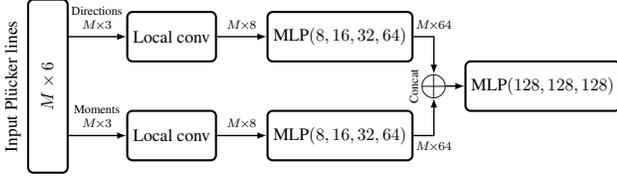
Figure 3: *Pipeline of subspace coding. The network takes M 6-dim Plücker lines as input, split lines into M 3-dim moments and directions, extract features in the subspace of moments and directions independently, and finally concatenate features from subspaces to obtain global high-dim Plücker line features. Local conv. stands for aggregating line-wise local geometry features and is given in Eq. (2). MLP(·) stands for multi-layer perceptron block, numbers in bracket are layer sizes.*
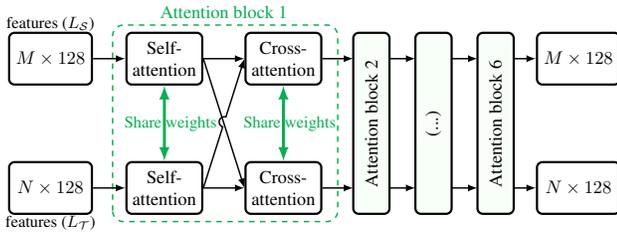

Figure 4: *Pipeline of discriminative feature embedding. The network takes M and N 128-dim line features from subspace coding as input, improve feature discriminativeness via both self and cross attention. We use 6 attention blocks in total.*

This process is given in Figure 4.

We first define two complete graphs for source and target line reconstructions, denoted by $\mathcal{G}_{L_{\mathcal{S}}}$ and $\mathcal{G}_{L_{\mathcal{T}}}$, respectively. Nodes in $\mathcal{G}_{L_{\mathcal{S}}}$ and $\mathcal{G}_{L_{\mathcal{T}}}$ corresponds to lines ℓ_i and ℓ_j', respectively. Node values corresponds to line features, and are denoted by ${}^{(t)}\mathbf{f}_{\ell_i}$ and ${}^{(t)}\mathbf{f}_{\ell_j'}$, where t is the layer index. Initial node values ($t = 1$) are outputs of subspace coding.

The two complete graphs ($\mathcal{G}_{L_{\mathcal{S}}}$, $\mathcal{G}_{L_{\mathcal{T}}}$) form a multiplex graph [34, 35]. It contains both intra-frame (or self) edges $\mathcal{E}_{\text{self}}$ and inter-frame (or cross) edges $\mathcal{E}_{\text{cross}}$. Intra-frame edges connecting all lines within the same line reconstructions. Inter-frame edges connecting one line from $\mathcal{G}_{L_{\mathcal{S}}}$ to all lines in $\mathcal{G}_{L_{\mathcal{T}}}$, and vice versa.

Node values are updated using multi-head self and cross attention in a message passing framework [44, 45]. The merit of using this framework is integrating both intra-frame and inter-frame contextual cues to increase distinctiveness of line features. For self-contain purposes, we briefly summarize it.

Node values in $\mathcal{G}_{L_{\mathcal{S}}}$ are updated via (same for $\mathcal{G}_{L_{\mathcal{T}}}$):

$$
{}^{(t+1)}\mathbf{f}_{\ell_i} = {}^{(t)}\mathbf{f}_{\ell_i} + \mathcal{U}\left({}^{(t)}\mathbf{f}_{\ell_i} || \mathfrak{m}_{\mathcal{E}\to\ell_i}\right), \tag{3}
$$

where $\cdot||\cdot$ denotes concatenation, $\mathfrak{m}_{\mathcal{E}\to\ell_i}$ denotes message from $\mathcal{E}$ to node ℓ_i, and $\mathcal{U}(\cdot)$ denotes feature propagation and is implemented as an MLP block MLP(256, 256, 128). The

message $\mathfrak{m}_{\mathcal{E}\to\ell_i}$ encodes propagation of all nodes which are connected to node ℓ_i via edges $\mathcal{E}$ ($\mathcal{E} = \mathcal{E}_{\text{self}}$ or $\mathcal{E}_{\text{cross}}$). In our implementation, the total depth of network is set to 12 (*i.e.*, $t \in [1, T], T = 12$) and we alternate to perform self and cross message passing with increasing network depth, *i.e.*, $\mathcal{E} = \mathcal{E}_{\text{self}}$ if t is odd and $\mathcal{E} = \mathcal{E}_{\text{cross}}$ if t is even.

Message $\mathfrak{m}_{\mathcal{E}\to\ell_i}$ is calculated through attention [51]:

$$
\mathfrak{m}_{\mathcal{E}\to\ell_i} = \Sigma_{j:(i,j)\in\mathcal{E}} \quad \alpha_{i,j}\mathtt{v}_j, \tag{4}
$$

where attention weight $\alpha_{i,j}$ is the Softmax over the key $\mathtt{k}_j$ to query $\mathtt{q}_i$ similarites, $\alpha_{i,j} = \text{Softmax}_j\left(\mathtt{q}_i^{\mathsf{T}}\mathtt{k}_j/\sqrt{D}\right)$, $D = 128$ is the feature dimension. In our implementation, the key $\mathtt{k}_j$, query $\mathtt{q}_i$ and value $\mathtt{v}_j$ are obtained via linear projection of line features, using different projection matrix. Following common practice, multi-head (4-heads in this paper) attention is used to improve the performance of Eq. (4). We do not use recent works [8, 54] concerning speed-up the computations of Eq. (4) as they all decrease the performance of our PlückerNet.

3.3. Feature matching

Given a learned feature descriptor per line in $L_{\mathcal{S}}$ and $L_{\mathcal{T}}$, we perform global feature matching to estimate the likelihood that a given Plücker line pair matches.

Matching cost. We first compute the pairwise distance matrix $\mathbf{H} \in \mathbb{R}_+^{M\times N}$, which measures the cost of assigning lines in $L_{\mathcal{S}}$ to $L_{\mathcal{T}}$. To calculate $\mathbf{H}$, we linearly project outputs ${}^{(T)}\mathbf{f}_{\ell_i}$ and ${}^{(T)}\mathbf{f}_{\ell_j'}$ from discriminative feature embedding process to obtain $\mathbf{f}_{\mathbf{x}_i}$ and $\mathbf{f}_{\mathbf{y}_j}$ for lines in $L_{\mathcal{S}}$ and $L_{\mathcal{T}}$, respectively. $\mathbf{f}_{\mathbf{x}_i}$ and $\mathbf{f}_{\mathbf{y}_j}$ are post L_2 normalized to embed them to a metric space. Each element of $\mathbf{H}$ is the L_2 distance between the features at line ℓ_i and ℓ_j', *i.e.*, $\mathbf{H}_{ij} = \|\mathbf{f}_{\mathbf{x}_i} - \mathbf{f}_{\mathbf{y}_j}\|_2$.

Prior Matchability. We are solving a partial-to-partial registration problem. Lines in $L_{\mathcal{S}}$ and $L_{\mathcal{T}}$ do not necessarily have to match. To model the likelihood that a given line has a match and is not an outlier, we define unary matchability vectors, denoted by $\mathbf{r}$ and $\mathbf{s}$ for lines in $L_{\mathcal{S}}$ and $L_{\mathcal{T}}$, respectively. To estimate $\mathbf{r}$ (same for $\mathbf{s}$), we add a lightweight matchability regression network, and the operation is defined by:

$$
\mathbf{r}_{\ell_i} = \text{Softmax}_i\left(\mathcal{P}\left({}^{(T)}\mathbf{f}_{\ell_i}||\text{avg}_{\ell_j'\in L_{\mathcal{T}}}{}^{(T)}\mathbf{f}_{\ell_j'}||\max_{\ell_j'\in L_{\mathcal{T}}}{}^{(T)}\mathbf{f}_{\ell_j'}\right)\right), \tag{5}
$$

where $\text{avg}_{\ell_j'\in L_{\mathcal{T}}}(\cdot)$ and $\max_{\ell_j'\in L_{\mathcal{T}}}(\cdot)$ denotes performing average and max pooling at each feature dimension, aiming to capture global context of line features for $L_{\mathcal{T}}$. $\mathcal{P}(\cdot)$ denotes feature propogation and is implemented as an MLP block MLP(384, 256, 256, 128, 1). Softmax is used to convert lines' logits to probabilities. Eq. (5) is based on
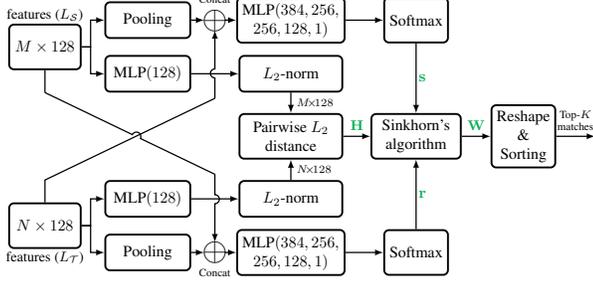
Figure 5: *Our feature matching pipeline. Given an $M \times 128$ feature set from discriminative feature embedding of source line set L_S and an $N \times 128$ feature set from the target line set L_T, we compute the pairwise L_2 distance matrix $\mathbf{H}$. Along with a unary matchability M-vector $\mathbf{r}$ from L_S and N-vector $\mathbf{s}$ from L_T, the distance matrix $\mathbf{H}$ is transformed to a joint probability matrix $\mathbf{W}$ using Sinkhorn's algorithm. Reshaping $\mathbf{W}$ and sorting the line matches by their corresponding matching probabilities generates a prioritized line match list. We take the Top-K matches as our set of putative correspondences.*

the idea of looking at global context of cross frame L_T to regress line-wise matching prior for current frame L_S.

Collecting the matchabilities for all lines in L_S or L_T yields a matchability histogram, a 1D probability distribution, given by $\mathbf{r} \in \Sigma_M$ and $\mathbf{s} \in \Sigma_N$, where a simplex in $\mathbb{R}^M$ is defined as $\Sigma_M = \left\{ \mathbf{r} \in \mathbb{R}_+^M, \sum_i \mathbf{r}_i = 1 \right\}$.

Global matching. From $\mathbf{H}$, $\mathbf{r}$, and $\mathbf{s}$, we estimate a weighting matrix $\mathbf{W} \in \mathbb{R}_+^{M \times N}$ where each element $\mathbf{W}_{ij}$ represents the matchability of the Plücker line pair $\{\boldsymbol{\ell}_i, \boldsymbol{\ell}'_j\}$. Note that each element $\mathbf{W}_{ij}$ is estimated from the cost matrix $\mathbf{H}$ and the unary matchability vectors $\mathbf{r}$ and $\mathbf{s}$, rather than locally from $\mathbf{H}_{ij}$. In other words, the weighting matrix $\mathbf{W}$ globally handles pairwise descriptor distance ambiguities in $\mathbf{H}$, while respecting the unary priors. The overall pipeline is given in Figure 5.

Sinkhorn solver. From optimal transport theory [52, 11, 10], the joint probability matrix $\mathbf{W}$ can be solved by

$$\underset{\mathbf{W} \in \Pi(\mathbf{r}, \mathbf{s})}{\arg\min} \langle \mathbf{H}, \mathbf{W} \rangle - \lambda E(\mathbf{W}), \qquad (6)$$

where $\langle \cdot, \cdot \rangle$ is the Frobenius dot product and $\Pi(\mathbf{r}, \mathbf{s})$ is the transport polytope that couples two unary matchability vectors $\mathbf{r}$ and $\mathbf{s}$, given by

$$\Pi(\mathbf{r}, \mathbf{s}) = \left\{ \mathbf{W} \in \mathbb{R}_+^{M \times N}, \mathbf{W}\mathbf{1}^N = \mathbf{r}, \mathbf{W}^\mathsf{T}\mathbf{1}^M = \mathbf{s} \right\}, \qquad (7)$$

where $\mathbf{1}^N = [1, 1, ..., 1]^\mathsf{T} \in \mathbb{R}^N$. The constraint on $\mathbf{W}$ ensures that we assign the matchabilities of each line in L_S (or L_T) to all lines in L_T (or L_S) without altering its matchability. The entropy regularization term $E(\mathbf{W})$

facilitates efficient computation [11] and is defined by

$$E(\mathbf{W}) = -\sum_{i,j} \mathbf{W}_{ij} \left(\log \mathbf{W}_{ij} - 1 \right). \qquad (8)$$

To solve (6), we use a variant of Sinkhorn's Algorithm [46, 32], given in Algorithm 1. Unlike the standard algorithm that generates a square, doubly-stochastic matrix, our usage generates a rectangular joint probability matrix, whose existence is guaranteed (Theorem 4 [32]).

Algorithm 1: Sinkhorn's Algorithm to solve (6). Hadamard (elementwise) division is denoted by $\oslash$.

Inputs: $\mathbf{H}, \mathbf{r}, \mathbf{s}, \mathbf{b} = \mathbf{1}^N$, λ, and iterations *Iter*
Output: Weighting matrix $\mathbf{W}$
1   $\boldsymbol{\Upsilon} = \exp(-\mathbf{H}/\lambda)$
2   $\boldsymbol{\Upsilon} = \boldsymbol{\Upsilon}/\sum \boldsymbol{\Upsilon}$
3   **while** $it < Iter$ **do**
4 | $\mathbf{a} = \mathbf{r} \oslash (\boldsymbol{\Upsilon}\mathbf{b})$; $\mathbf{b} = \mathbf{s} \oslash (\boldsymbol{\Upsilon}^\mathsf{T}\mathbf{a})$
5   **end**
6   $\mathbf{W} = \mathrm{diag}(\mathbf{a})\boldsymbol{\Upsilon}\,\mathrm{diag}(\mathbf{b})$

Loss. To train our feature extraction and matching network, we apply a loss function to the weighting matrix $\mathbf{W}$. Since $\mathbf{W}$ models the joint probability distribution of $\mathbf{r}$ and $\mathbf{s}$, we can maximize the joint probability of inlier correspondences and minimize the joint probability of outlier correspondences using

$$\mathcal{L} = \sum_i^M \sum_j^N \gamma_{i,j} \mathcal{F}\left(\mathbf{C}_{ij}^{\mathrm{gt}}, \mathbf{W}_{ij}\right), \qquad (9)$$

where $\mathbf{C}_{ij}^{\mathrm{gt}}$ equals to 1 if $\{\boldsymbol{\ell}_i, \boldsymbol{\ell}'_j\}$ is a true correspondence and 0 otherwise. $\mathcal{F}\left(\mathbf{C}_{ij}^{\mathrm{gt}}, \mathbf{W}_{ij}\right)$ and $\gamma_{i,j}$ are given by,

$$\begin{cases} \mathcal{F}\left(\mathbf{C}_{ij}^{\mathrm{gt}}, \mathbf{W}_{ij}\right) = -\log \mathbf{W}_{ij}; \gamma_{i,j} = 1/\mathcal{A}_{\mathrm{true}} & \text{if } \mathbf{C}_{ij}^{\mathrm{gt}} = 1 \\ \mathcal{F}\left(\mathbf{C}_{ij}^{\mathrm{gt}}, \mathbf{W}_{ij}\right) = -\log(1 - \mathbf{W}_{ij}); \gamma_{i,j} = 1/\mathcal{A}_{\mathrm{false}} & \text{if } \mathbf{C}_{ij}^{\mathrm{gt}} = 0, \end{cases} \qquad (10)$$

where $\gamma_{i,j}$ is a weight to balance true and false correspondences. $\mathcal{A}_{\mathrm{true}}$ and $\mathcal{A}_{\mathrm{false}}$ are the total number of true and false correspondences, respectively.

3.4. Pose estimation

We now have a set of putative line correspondences, some of which are outliers, and want to estimate the relative pose between source and target line reconstructions. According to [5], we need minimal 2 line correspondences to solve the relative rotation $\mathbf{R}$ and translation $\mathbf{t}$ in similarity space. In Euclidean space, by substituting Eq. (1) to $\boldsymbol{\ell}'_j = \mathbf{T}\boldsymbol{\ell}_i$, we obtain,

$$\mathbf{m}'_j = \mathbf{R}\mathbf{m}_i + [\mathbf{t}]_\times \mathbf{R}\mathbf{v}_i \qquad (11)$$

$$\mathbf{v}'_j = \mathbf{R}\mathbf{v}_i. \qquad (12)$$

Note $\mathbf{t}$ is not contained in Eq. (12). We can first solve $\mathbf{R}$, then substitute $\mathbf{R}$ to Eq. (11) to solve $\mathbf{t}$.

According to Eq. (12), $\mathbf{R}$ aligns line direction $\mathbf{v}_i$ to $\mathbf{v}'_j$. In [19], authors show that $\mathbf{R}$ is the closest orthonormal matrix to $\mathbf{M} = \sum \mathbf{v}'_j \mathbf{v}_i^\top$. Let $\mathbf{M} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ be the singular value decomposition of $\mathbf{M}$, then $\mathbf{R} = \mathbf{U}\mathbf{V}^\top$. The sign ambiguity of $\mathbf{R}$ is fixed by $\mathbf{R} = \mathbf{R}/\det(\mathbf{R})$.

Substitute estimated $\mathbf{R}$ to Eq. (11), and reshape it as:

$$[\mathbf{R}\mathbf{v}_i]_\times^\top \mathbf{t} = \mathbf{m}'_j - \mathbf{R}\mathbf{m}_i. \tag{13}$$

Stacking along rows for $i = 1, 2$ yields a linear equation $\mathbf{A}\mathbf{t} = \mathbf{b}$, where $\mathbf{A}$ is a 6×3 matrix and $\mathbf{b}$ is a 6×1 vector. The least square solution is given by $\mathbf{t} = \mathbf{A}^+ \mathbf{b}$, where $\mathbf{A}^+$ is the pseudo-inverse of $\mathbf{A}$.

2-**line minimal solver in RANSAC framework.** Given the above 2-line minimal solver, we are ready to integrate it into the RANSAC framework. We need to define the score function, which is used to evaluate the number of inlier line correspondences given an estimated pose from the minimal solver. Our score function is defined by:

$$\mathcal{S}(\boldsymbol{\ell}', \boldsymbol{\ell}) = \left\| \boldsymbol{\ell}' - \mathbf{T}\boldsymbol{\ell} \right\|_2, \tag{14}$$

where we use estimated line motion matrix $\mathbf{T}$ (Eq. (1)) to transform 6-dim Plücker lines from $L_\mathcal{S}$ to $L_\mathcal{T}$, and measure the L_2 distance between 6-dim Plücker lines.

We choose to use L_2 distance for its simplicity, and quadratic for easy-minimization. Though L_2 distance only works in a neighborhood of $\boldsymbol{\ell}'$ ($\mathbf{v}$ and $\mathbf{m}$ lie in two different subspaces), in practice, transformed lines from $L_\mathcal{S}$ using our estimated pose lie within the local neighborhood of their matching line in $L_\mathcal{T}$ (if have), leading to the success of using L_2 distance. Despite Plücker lines lie in a 4-dim Klein manifold [41], we do not use distance defined via Klein quadric as it needs to specify two additional planes for each line pair, which introduces many hyper-parameters [40].

For a Plücker line matching pair, if the score function defined in Eq. (14) is smaller than a pre-defined threshold ϵ, it is deemed as an inlier pair. After RANSAC, all inlier matching pairs are used to jointly optimize Eq. (14).

4. Experiments

4.1. Datasets and evaluation methodology

We first conduct experiments on both indoor (Structured3D [63]) and outdoor (Semantic3D [14]) datasets, and then show our results of addressing real-world line-based visual odometry on the Apollo dataset [1].

Structured3D [63] contains 3D annotations of junctions and lines for indoor houses. It has $3,500$ scenes/houses in total, with average/median number of lines at 306/312, respectively. The average size of a house is around 11m $\times$

10m $\times$ 3m. Since the dataset captures structures of indoor houses, most lines are parallel or perpendicular to each other. We randomly split this dataset to form a training and testing dataset, with numbers at 2975 and 525, respectively.

Semantic3D [14] contains large-scale and densely-scanned 3D points cloud of diverse urban scenes. We use the semantic-8 dataset, and it contains 30 scans, with over a billion points in total. For each scan, we use a fast 3D line detection method [59] to extract 3D line segments. Large-scale 3D line segments are further partitioned into different geographical scenes/cells, with cell size at 10m $\times$ 10m for the X–Y dimensions. Scenes with less than 20 lines are removed. We obtain $1,981$ scenes in total, with average/median number of lines at 676/118, respectively. We randomly split this dataset to form a training and testing dataset, with numbers at 1683 and 298, respectively.

Partial-to-Partial registration. For 3D lines of each scene ($L_\mathcal{S}$), we sample a random rigid transformation along each axis, with rotation in $[0°, 45°]$ and translation in $[-2.0\text{m}, 2.0\text{m}]$, and apply it to the source line set $L_\mathcal{S}$ to obtain the target line set $L_\mathcal{T}$. We then add noise to lines in $L_\mathcal{S}$ and $L_\mathcal{T}$ independently. Specifically, we first transform Plücker representation of a line to point-direction representation, with the point at the footprint of line's perpendicular through the origin. Gaussian noise sampled from $\mathcal{N}(0\text{m}, 0.05\text{m})$ and clipped to $[-0.25\text{m}, 0.25\text{m}]$ is added to the footprint point, and the direction is perturbed by a random rotation, with angles sampled form $\mathcal{N}(0°, 2°)$ and clipped to $[-5°, 5°]$. After adding noises, point-direction representations are transformed back to Plücker representations. To simulate partial scans of $L_\mathcal{S}$ and $L_\mathcal{T}$, we randomly select 70% lines from $L_\mathcal{S}$ and $L_\mathcal{T}$ independently, yielding an overlapping ratio at ~ 0.7.

Implementation details. Our network is implemented in Pytorch and is trained from scratch using the Adam optimizer [21] with a learning rate of 10^{-3} and a batch size of 12. The number of nearest neighbors for each line is set to 10 in Eq. (2). The number of Sinkhorn iterations is set to 30, λ is set to 0.1, Top-200 line matches are used for pose estimation, the inlier threshold ϵ is set to 0.5m, and the number of RANSAC iterations is set to 1000. Our model is trained on a single NVIDIA Titan XP GPU in one day.

Evaluation metrics. The rotation error is given by the angle difference $\zeta = \arccos((\text{trace}(\mathbf{R}_{\text{gt}}^\top \mathbf{R}) - 1)/2)$, where $\mathbf{R}_{\text{gt}}$ and $\mathbf{R}$ is the ground-truth and estimated rotation, respectively. The translation error is given by the L_2 distance between the ground-truth and estimated translation vector. We also calculate the recalls (percentage of poses) by varying pre-defined thresholds on rotation and translation error. For each threshold, we count the number of poses with error less than that threshold, and then normalize the number by the total number of poses.
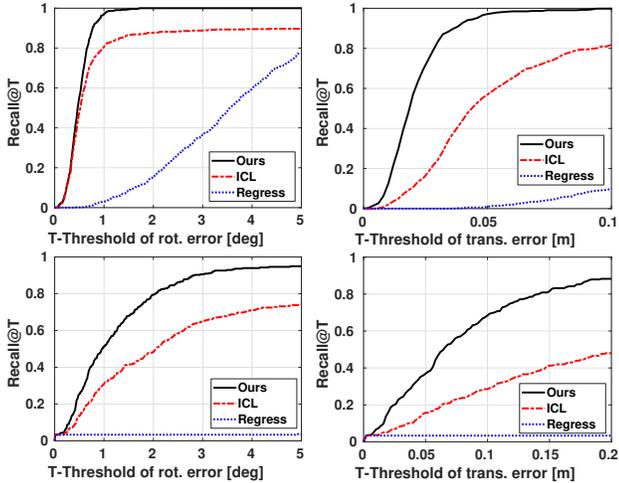
Figure 6: *Recall of rotation (Left) and translation (Right) on the Structured3D (Top-row) and Semantic3D (Bottom-row) datasets, with respect to an error threshold. The regression baseline diverges on the Semantic3D dataset, leading to the poor performance.*
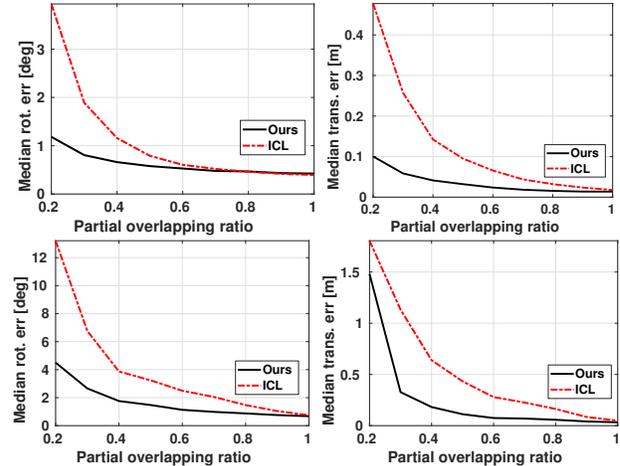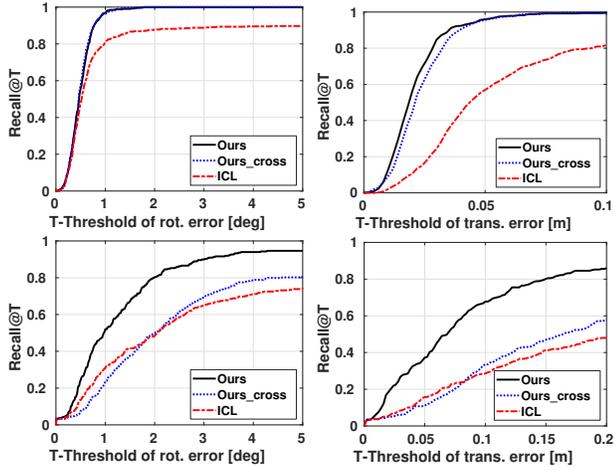

Figure 7: *Median rotation (Left) and translation error (Right) on the Structured3D (Top-row) and Semantic3D (Bottom-row), with respect to increasing level of partial overlapping ratio.*

Baselines. There is no off-the-shelf baseline available. We propose and implement: **1) ICL.** an Iterative-Closest-Line (ICL) method, mimicking the pipeline of traditional Iterative-Closest-Point (ICP) method. **2) Regression.** This baseline does not estimate line-to-line matches. After extracting line-wise features, we append a global max-pooling layer to obtain a global feature for each source and target set. A concatenated global feature is fed to an MLP block to directly regress the rotation and translation.

4.2. Results and discussions

Comparison with baselines. In this experiment, we show the effectiveness of our method to estimate a 6-DoF pose. We compare our method against baselines. The pose estimation performance is given in Table 1 and Figure 6. It shows that our method outperforms baselines with much higher recalls at each level of pre-defined error thresholds. For example, the median rotation and translation error on Semantic3D dataset for our method, ICL and regression is $0.961°/2.050°/20.934°$ and $0.064m/0.226m/1.871m$, respectively. We found that the regression baseline can be trained to converge on the Structured3D dataset, while diverges on the Semantic3D dataset. This shows that directly regressing the relative pose between line reconstructions falls short of applying to different datasets. Since both our method and ICL outperforms the regression baseline significantly, we only compare our method with ICL for following experiments.

Robustness to overlapping ratios. We test the robustness of our network to overlapping ratios for partial-to-partial registration. Overlapping ratios are set within $[0.2, 1]$,

where overlapping ratio at 1 means source and target line reconstructions have one-to-one line match. For overlapping ratios smaller than 0.2, both our method and ICL fail to estimate meaningful poses. Note we do not re-train networks. The median rotation and translation errors with respect to increasing levels of overlapping ratio are given in Figure 7. Our method outperforms ICL on the Structured3D and Semantic3D datasets.

Generalization ability. We test the generalization ability of our network via cross-dataset validation, *i.e.*, using a network trained on the Structured3D to test its performance on the Semantic3D, and vice versa. The comparison of recall performance is given in Figure 8. For the Structured3D dataset, our network trained on the Semantic3D dataset get a similar recall performance as the network trained on the Structured3D dataset, both outperforming the ICL method. For the Semantic3D dataset, though there is a recall performance drop for our cross-dataset trained network, its performance is comparable to the ICL method.

Real-world line-based visual odometry. In this experiment, we demonstrate the effectiveness of our method for a real-world application of registering 3D line reconstructions, *i.e.* visual odometry. We randomly select one sequence (road02_ins) from the apollo repository [1], and it contains 5123 frames. For each frame, we first use a fast line segment detector [2] to obtain 2D line segments, and then use the provided depth map (from Lidar) to fit 3D lines [15]. The median number of 3D lines is 1975. Sample 2D and 3D line segments are given in Figure 9.

Given 3D line segments for each frame, we use Plücker representations of line segments to compute relative poses for consecutive frames, using our model trained on the

Table 1: *Comparison of rotation and translation errors on the Structured3D and Semantic3D dataset. Q1 denotes the first quartile, Med. denotes the median, and Q3 denotes the third quartile.*

Error Method	Structured3D						Semantic3D					
	Rotation (°)			Translation (m)			Rotation (°)			Translation (m)		
	Q1	Med.	Q3	Q1	Med.	Q3	Q1	Med.	Q3	Q1	Med.	Q3
ICL	0.353	0.520	0.795	0.030	0.044	0.078	0.803	2.050	5.239	0.084	0.226	0.881
Regression	2.436	3.610	4.935	0.151	0.240	0.367	15.902	20.934	24.947	1.347	1.871	2.281
Ours	**0.342**	**0.468**	**0.621**	**0.013**	**0.019**	**0.026**	**0.482**	**0.961**	**1.791**	**0.032**	**0.064**	**0.119**

Figure 8: *Recall of rotation (Left) and translation (Right) on the Structured3D (Top-row) and Semantic3D (Bottom-row), with respect to an error threshold. Ours_cross denotes using network trained on one dataset to test on the other.*

Figure 9: *Sample 2D (Left) and 3D (Right) line segments from the Apollo dataset.*

Figure 10: *Recall of rotation (Left) and translation (Right) on the Apollo dataset, with respect to an error threshold. We directly use model trained on Semantic3D dataset without fine-tuning.*

speed-up the time-consuming RANSAC iterations in C++.

5. Conclusion

In this paper, we have proposed the first end-to-end trainable network for solving the problem of aligning two partially-overlapped 3D line reconstructions in Euclidean space. The key innovation is to directly learn line-wise features by parameterizing lines as 6-dim Plücker coordinates and respecting line geometry during feature extraction. We use these line-wise features to establish line matches via a global matching module under the optimal transport framework. The high-quality line correspondences found by our method enable to apply a 2-line minimal-case RANSAC solver to estimate the 6-DOF pose between two line reconstructions. Experiments on three benchmarks show that the new method significantly outperforms baselines (iterative closet lines and direct regression) in terms of registration accuracy.

Acknowledgement

Semantic3D dataset without fine-tuning. We compare our method with ICL, and the results are given in Figure 10. Our method outperforms ICL with much higher rotation and translation recalls, with median rotation and translation error for our method and ICL at $1.13°/0.44$m and $2.83°/1.12$m, respectively.

Computational efficiency. We compare the computation time of our approach to ICL. The averaged time breakdowns for our method are 0.037s/0.008s/0.327s (Structured3D), 0.036s/0.024s/0.335s (Semantic3D) for feature extraction, matching and RANSAC pose estimation, respectively. The averaged time for ICL is 0.055s/0.144s for Structured3D and Semantic3D, respectively. The most time-consuming part of our method is the RANSAC pose estimation step, which is programmed in Python. Interested readers can
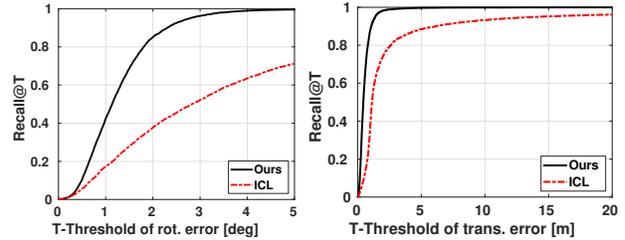
References

[1] Apollo Scene Parsing, http://apolloscape.auto/scene.html, Accessed: 2020-10-30. 2, 6, 7

[2] Cuneyt Akinlar and Cihan Topal. Edlines: Real-time line segment detection by edge drawing (ed). In *2011 18th IEEE International Conference on Image Processing*, pages 2837–2840. IEEE, 2011. 2, 7

[3] Nicolas Andreff, Bernard Espiau, and Radu Horaud. Visual servoing from lines. *The International Journal of Robotics Research*, 21(8):679–699, 2002. 1

[4] Yasuhiro Aoki, Hunter Goforth, Rangaprasad Arun Srivatsan, and Simon Lucey. Pointnetlk: Robust & efficient point cloud registration using pointnet. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7163–7172, 2019. 2

[5] Adrien Bartoli, Richard I Hartley, and Fredrik Kahl. Motion from 3d line correspondences: linear and nonlinear solutions. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 1, pages I–477. IEEE, 2003. 1, 2, 5

[6] Adrien Bartoli and Peter Sturm. The 3d line motion matrix and alignment of line reconstructions. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I. IEEE, 2001. 1, 2, 3

[7] Dylan Campbell*, Liu Liu*, and Stephen Gould. Solving the blind perspective-n-point problem end-to-end with robust differentiable geometric optimization. In *ECCV*, 2020. * equal contribution. 2

[8] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020. 4

[9] Andrea Cohen, Johannes L Schönberger, Pablo Speciale, Torsten Sattler, Jan-Michael Frahm, and Marc Pollefeys. Indoor-outdoor 3d reconstruction alignment. In *European Conference on Computer Vision*, pages 285–300. Springer, 2016. 1

[10] Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. Optimal transport for domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 39(9):1853–1865, 2016. 2, 5

[11] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems*, pages 2292–2300, 2013. 2, 5

[12] Zheng Dang, Kwang Moo Yi, Yinlin Hu, Fei Wang, Pascal Fua, and Mathieu Salzmann. Eigendecomposition-free training of deep networks with zero eigenvalue-based losses. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 768–783, 2018. 2

[13] Zheng Dang, Fei Wang, and Mathieu Salzmann. Learning 3d-3d correspondences for one-shot partial-to-partial registration. *arXiv preprint arXiv:2006.04523*, 2020. 2, 3

[14] Timo Hackel, N. Savinov, L. Ladicky, Jan D. Wegner, K. Schindler, and M. Pollefeys. SEMANTIC3D.NET: A new large-scale point cloud classification benchmark. In *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume IV-1-W1, pages 91–98, 2017. 1, 2, 6

[15] Shida He, Xuebin Qin, Zichen Zhang, and Martin Jagersand. Incremental 3d line segment extraction from semi-dense slam. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1658–1663. IEEE, 2018. 7

[16] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 3

[17] William Vallance Douglas Hodge, WVD Hodge, and Daniel Pedoe. *Methods of Algebraic Geometry: Volume 2*, volume 2. Cambridge University Press, 1994. 3

[18] Manuel Hofer, Michael Maurer, and Horst Bischof. Line3d: Efficient 3d scene abstraction for the built environment. In *German Conference on Pattern Recognition*, pages 237–248. Springer, 2015. 1, 2

[19] Berthold KP Horn, Hugh M Hilden, and Shahriar Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *JOSA A*, 5(7):1127–1135, 1988. 6

[20] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 3

[21] Diederick P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 6

[22] Tobias Koch, Marco Körner, and Friedrich Fraundorfer. Automatic alignment of indoor and outdoor building models using 3d line segments. *Proceedings of Computer Vision and Pattern Recognition 2016*, pages 10–18, 2016. 1

[23] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955. 2

[24] Sang Jun Lee and Sung Soo Hwang. Elaborate monocular point and line slam with robust initialization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1121–1129, 2019. 1

[25] Kai Li, Jian Yao, Mengsheng Lu, Yuan Heng, Teng Wu, and Yinxuan Li. Line segment matching: a benchmark. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9. IEEE, 2016. 2, 3

[26] Liu Liu, Dylan Campbell, Hongdong Li, Dingfu Zhou, Xibin Song, and Ruigang Yang. Learning 2d-3d correspondences to solve the blind perspective-n-point problem. *arXiv preprint arXiv:2003.06752*, 2020. 2

[27] Liu Liu, Hongdong Li, and Yuchao Dai. Stochastic attraction-repulsion embedding for large scale image localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2570–2579, 2019. 1

[28] Yuncai Liu, Thomas S Huang, and Olivier D Faugeras. Determination of camera location from 2-d to 3-d line and point correspondences. *IEEE Transactions on pattern analysis and machine intelligence*, 12(1):28–37, 1990. 1

[29] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 2

[30] Yang Lyu, Quan Pan, Yizhai Zhang, Chunhui Zhao, Haifeng Zhu, Tongguo Tang, and Liu Liu. Simultaneously multi-uav mapping and control with visual servoing. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 125–131. IEEE, 2015. 1

[31] Lingfei Ma, Ying Li, Jonathan Li, Zilong Zhong, and Michael A Chapman. Generation of horizontally curved driving lines in hd maps using mobile laser scanning point clouds. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(5):1572–1586, 2019. 1

[32] Albert W Marshall and Ingram Olkin. Scaling of matrices to achieve specified row and column sums. *Numerische Mathematik*, 12(1):83–90, 1968. 2, 5

[33] Kwang Moo Yi, Eduard Trulls, Yuki Ono, Vincent Lepetit, Mathieu Salzmann, and Pascal Fua. Learning to find good correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2666–2674, 2018. 2

[34] Peter J Mucha, Thomas Richardson, Kevin Macon, Mason A Porter, and Jukka-Pekka Onnela. Community structure in time-dependent, multiscale, and multiplex networks. *science*, 328(5980):876–878, 2010. 4

[35] Vincenzo Nicosia, Ginestra Bianconi, Vito Latora, and Marc Barthelemy. Growing multiplex networks. *Physical review letters*, 111(5):058701, 2013. 4

[36] Liyuan Pan, Shah Chowdhury, Richard Hartley, Miaomiao Liu, Hongguang Zhang, and Hongdong Li. Dual pixel exploration: Simultaneous depth estimation and image restoration. *arXiv preprint arXiv:2012.00301*, 2020. 1

[37] Liyuan Pan, Yuchao Dai, and Miaomiao Liu. Single image deblurring and camera motion estimation with depth map. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2116–2125. IEEE, 2019. 1

[38] Liyuan Pan, Richard Hartley, Cedric Scheerlinck, Miaomiao Liu, Xin Yu, and Yuchao Dai. High frame rate video reconstruction based on an event camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 1

[39] Liyuan Pan, Miaomiao Liu, and Richard Hartley. Single image optical flow estimation with an event camera. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1669–1678. IEEE, 2020. 1

[40] Helmut Pottmann, Michael Hofer, Boris Odehnal, and Johannes Wallner. Line geometry for 3d shape understanding and reconstruction. In *European Conference on Computer Vision*, pages 297–309. Springer, 2004. 6

[41] Helmut Pottmann and Johannes Wallner. *Computational line geometry*. Springer Science & Business Media, 2009. 1, 6

[42] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 1, 2, 3

[43] Andrei Rares, Marcel JT Reinders, and Jan Biemond. Edge-based image restoration. *IEEE Transactions on Image Processing*, 14(10):1454–1468, 2005. 1

[44] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4938–4947, 2020. 1, 2, 3, 4

[45] Martin Simon, Kai Fischer, Stefan Milz, Christian Tobias Witt, and Horst-Michael Gross. Stickypillars: Robust feature matching on point clouds using graph neural networks. *arXiv preprint arXiv:2002.03983*, 2020. 1, 2, 3, 4

[46] Richard Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The annals of mathematical statistics*, 35(2):876–879, 1964. 2, 5

[47] Paul Smith, ID Reid, and Andrew J Davison. Real-time monocular slam with straight lines. 2006. 1

[48] Felix Taubner, Florian Tschopp, Tonci Novkovic, Roland Siegwart, and Fadri Furrer. Lcd–line clustering and description for place recognition. *arXiv preprint arXiv:2010.10867*, 2020. 1

[49] Camillo J Taylor and David J Kriegman. Structure and motion from line segments in multiple images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(11):1021–1032, 1995. 1

[50] David Reverter Valeiras, Xavier Clady, Sio-Hoi Ieng, and Ryad Benosman. Event-based line fitting and segment detection using a neuromorphic visual sensor. *IEEE transactions on neural networks and learning systems*, 30(4):1218–1230, 2018. 1

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 4

[52] Cédric Villani. Optimal transport–old and new, volume 338 of a series of comprehensive studies in mathematics, 2009. 2, 5

[53] 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, 2008. 2

[54] Sinong Wang, Belinda Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020. 4

[55] Xiaogang Wang, Yuelang Xu, Kai Xu, Andrea Tagliasacchi, Bin Zhou, Ali Mahdavi-Amiri, and Hao Zhang. Pie-net: Parametric inference of point cloud edges. *arXiv preprint arXiv:2007.04883*, 2020. 2

[56] Yue Wang and Justin M. Solomon. Deep closest point: Learning representations for point cloud registration. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 3

[57] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018. 2, 3

[58] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 3

[59] Lu Xiaohu, Liu Yahui, and Li Kai. Fast 3d line segment detection from unorganized point cloud. *arXiv preprint arXiv:1901.02532*, 2019. 1, 2, 6

[60] Nan Xue, Tianfu Wu, Song Bai, Fudong Wang, Gui-Song Xia, Liangpei Zhang, and Philip HS Torr. Holistically-attracted wireframe parsing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2788–2797, 2020. 2

[61] Guoxuan Zhang, Jin Han Lee, Jongwoo Lim, and Il Hong Suh. Building a 3-d line-based map using stereo slam. *IEEE Transactions on Robotics*, 31(6):1364–1377, 2015. 1, 3

[62] 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

[63] 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. 2, 6

[64] Huizhong Zhou, Danping Zou, Ling Pei, Rendong Ying, Peilin Liu, and Wenxian Yu. Structslam: Visual slam with building structure lines. *IEEE Transactions on Vehicular Technology*, 64(4):1364–1375, 2015. 1

[65] Yichao Zhou, Haozhi Qi, and Yi Ma. End-to-end wireframe parsing. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 962–971, 2019. 2