Fast Point Cloud Registration for Urban Scenes via Pillar-Point Representation

Siyuan Gu and Ruqi Huang[⋆]

Tsinghua Shenzhen International Graduate School, Tsinghua-Berkeley Shenzhen Institute

Abstract. Efficient and robust point cloud registration is an essential task for real-time applications in urban scenes. Most methods introduce keypoint sampling or detection to achieve real-time registration of large-scale point clouds. Recent advances in keypoint-free methods have succeeded in alleviating the bias and error introduced by keypoint detection via coarse-to-fine dense matching strategies. Nevertheless, the running time performance of such a strategy turns out to be far inferior to keypoint methods. This paper proposes a novel framework that adopts a pillar-point representation based feature extraction pipeline and a three-stage semi-dense keypoint matching scheme. The scheme includes global coarse matching, anchor generation and local dense matching for efficient correspondence matching. Experiments on large-scale outdoor datasets, including KITTI and NuScenes, demonstrate that the proposed feature representation and matching framework achieve real-time inference and high registration recall.

Keywords: Point cloud registration · Pillar-point representation · Semi-dense keypoint matching.

1 Introduction

Point cloud registration plays a fundamental role in various applications including indoor scene reconstruction, drone mapping, autonomous driving, to name a few. In general, it aims to estimate the optimal rigid transformation between a pair of unaligned point clouds. Early axiomatic approaches cast point registration as minimizing certain types of geometric residuals (e.g., Euclidean distance [21], or normal distribution [18]). On the other hand, recent trends in learning-based methods[6, 2, 10, 16] follow a generalized correspondence-based workflow, which consists of feature extraction, keypoint sampling, correspondence matching, outlier rejection, and pose estimation.

When dealing with large-scale point clouds, such as those generated by scanning devices like LiDARs at a rate of hundreds of thousands of points per second, most axiomatic methods and learning-based methods alleviate computational burden by utilizing keypoint sampling. In particular, modern learning-based methods typically obtain keypoints through either plain Furthest Point Sampling (FPS) [16] or learned saliency [2]. However, keypoint-based methods are not only influenced by the sampling scheme but also often rely on the assumption that a sufficient number of repeatable keypoints across

[⋆] Ruqi Huang is the corresponding author (ruqihuang@sz.tsinghua.edu.cn).

input point clouds are available. Methods like [12, 2] attempt to enhance keypoint repeatability by employing deep learning-based keypoint detectors, but they still face challenges associated with detection errors.

By contrast, the recent keypoint-free paradigm[27, 19] proposes to perform dense point matching in a coarse-to-fine manner. This paradigm tackles the challenge of dense matching in large-scale point clouds by breaking point clouds down into local dense matching within multiple matched patches (*i.e.*, superpoints). The keypoint-free approach has demonstrated excellent performance in terms of accuracy and robustness. Despite the employment of hierarchical matching strategies and the avoidance of time-consuming RANSAC pose estimators, these keypoint-free methods have not yet achieved runtime efficiency comparable to methods that utilize keypoint sampling or detection [16, 7] in large-scale outdoor scenes.

Motivated by the aforementioned observations, we propose a pipeline that incorporates a novel matching scheme, which involves sampling anchor points on only one side of input point cloud pairs, referred to as semi-dense keypoint sampling. The corresponding candidates to anchor points are searched in feature space and the coarse matching stage can help to reduce the search space to ensure robustness. Virtual corresponding points as weighted sum of candidates are generated and used as centers of mini patches in local fine matching. Compared to keypoint-free methods, anchor correspondence generation narrows the distance between mini patches, leading to a higher inlier ratio and smaller neighborhood size.

Experimental results demonstrate that our pipeline achieves comparable registration performance with the state-of-the-art methods, including both keypoint-based and keypoint-free approaches. Additionally, our pipeline exhibits excellent generalization performance when applied to unseen scenes. Furthermore, as a pipeline that eliminates the need for keypoint detection, we significantly improve the runtime efficiency by a large margin, typically performing four times faster than the keypoint-free counterpart [19].

The main contributions of our work are listed as follows:

- An efficient and robust registration network for large-scale urban point clouds based on pillar-point representations.
- A three-stage correspondence matching scheme with semi-dense anchor correspondence generation.
- Extensive experiments on large-scale urban datasets have proved the high efficiency of our proposed network with comparable registration accuracy.

2 Related Works

Learned 3D Feature Descriptor Early works in 3D feature learning most belong to patch-based methods. [28] proposes the representative 3DMatch benchmark and a neural network utilizing voxel-based Truncated Distance Function as local patch descriptors. [1] proposes 3D cylindrical convolution layers to extract features from spherically voxelized point cloud. [6] first suggests to learn dense descriptors (FCGF) from full voxelized point cloud via sparse 3D convolutions and gains enormous speedup. [26] proposes KPConv which defines convolution on precomputed kernel points to adapt to irregularity of point clouds. KPConv is utilized by several registration methods including[2, 27, 19].

Works in object detection area also propose some generic represent learning concepts. [23, 14, 25] construct feature extract modules with keypoint-to-voxel set abstraction. This kind of modules first query keypoint features via trilinear interpolation from multi-scale voxel features or directly from independent point MLP and then fuse all the features. [11] and related [22] divide scenes into pillar structure with infinite length in vertical direction and achieve significant breakthrough in inference speed.

Correspondence-based Registration Methods Point cloud registration methods which utilize a common correspondence step can be summarized into two main categories, *i.e.*, with or without keypoints. The former, such as [16, 13], use uniform grid sampling or farthest point sampling to extract a few interest points and construct a keypoint set for training. Similar to our work, [7] suggests a pillar-based method to construct patches based on filtered salient keypoints and utilizes graph neural network to search correspondences. These methods reduce storage for interest points from the beginning but are easily influenced by keypoint sampling strategies.

On the other hand, keypoint-free methods including [6, 27], inspired by classical image matching method [20], perform hierarchical matching on point cloud. They utilize patch overlaps to supervise superpoint matching and solve patch-wise point matching with optimal transport techniques. Though they can achieve more stable results than the keypoint-based counterparts, they suffer from efficiency issue when the input point clouds become larger.

3 Methodology

Given a pair of input clouds denoted as $\mathcal{P} \in \mathbb{R}^{m\times 3}$ and $\mathcal{Q} \in \mathbb{R}^{n\times 3}$ respectively, we aim to estimate rigid rotation $\mathbf{R} \in SO(3)$ and translation $\mathbf{t} \in \mathbb{R}^3$ that align $\mathcal{P}$ and $\mathcal{Q}$. As shown in Fig. 1, our pipeline first learns a two-scale feature representation with a shared encoder-decoder, then solves for point correspondences and finally predicts rigid transformation.

3.1 Pillar-Point Based Feature Extractor

Pillar Feature Embedding Common scenes perceived from ground LiDAR have such uniqueness that the variances of the roll and pitch components of transformations are significantly smaller than that of the yaw component, or $Var(\theta_z) \gg Var(\theta_x), Var(\theta_y)$ under Euler angle form. Therefore the pillar-based feature embedding, previously utilized in [11], is introduced into our registration pipeline for efficiency.

A pillar structure is organized by uniformly dividing the whole scene into regular grids with full height in z-axis. For each pillar with center $\mathbf{x}_c^i = (x_c^i, y_c^i, z_c^i)$, point set $\{\mathbf{x}_j^i\}$ and its arithmetic mean $\overline{\mathbf{x}}^i\}$, we calculate a 6-dimensional vector $F_{pt} = \{x_j^i - x_c^i, y_j^i - y_c^i, z_j^i - z_c^i, x_j^i - \overline{x}^i, y_j^i - \overline{y}^i, z_j^i - \overline{z}^i\}$ as the initial input for every single point in the pillar. Then a two-layer shared MLP is applied to encode the statistics into pillar features $F_{pl} = \mathrm{MLP}(F_{pt})$. In the embedding process, dynamic voxelization strategy from [30] is adopted in our pillar partition step and hence no points are truncated.

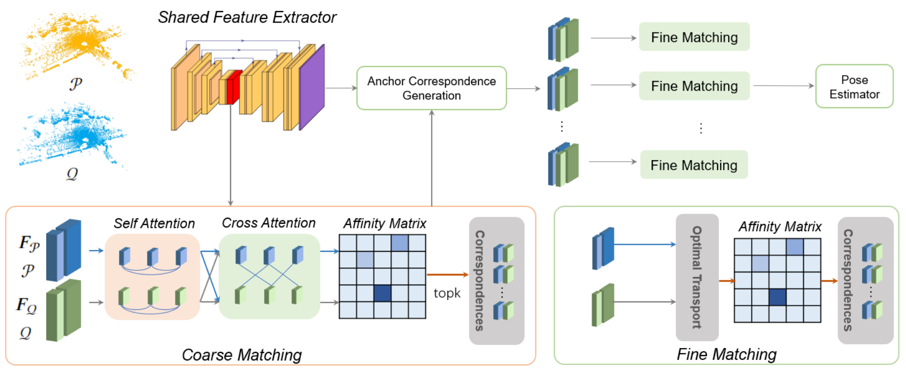

Fig. 1. The proposed point cloud registration pipeline is mainly composed a shared feature extractor, a hierarchical feature scheme and a pose estimator. The shared pillar-point based feature extractor extracts coarse-scale feature (marked in red) in downsampling and fine-scale feature (marked in purple) in upsampling. The feature matching scheme is composed of three stages, namely coarse matching, anchor generation and fine matching, which generate correspondences at different levels. The details of proposed feature matching scheme is stated in Section 3.2. Finally, a pose estimator based on an iterative proposal and verification scheme filters out outliers and outputs final transformation.

Convolutional Backbone We adopt a Residual U-Net backbone like that in [6], which is built with sparse 2D convolution block to aggregate pillar features. The backbone has two output features, coarse-level features F_c at the bottleneck of backbone and fine-level features F_f. It should be noticed that a proposed point feature decoder is appended to last convolutional layer to revert tensors located on 2D grids to point-wise features by interpolation.

Point Feature Decoder In our case, a pillar contains multiple points, inconsistent to the output feature map F_{bev} of last convolutional layer. To convert the feature map into point-wise features F_f suitable for fine matching among points, we incorporate the concept of set abstraction in [14] to further increase the separability of points. The final point feature F_f is calculated by $F_f = \text{MLP}(\text{Concat}[\mathbf{x}, F_{pl}, F_{bev}])$. The grid set abstraction procedure is illustrated in Fig. 2. It simultaneously reserves the scalability of voxel/pillar-based methods and the compatibility with point-based methods.

3.2 Hierarchical Matching Scheme

Coarse Matching Following [20], we adopt a transformer to learn patch features at a coarse-level. The transformer structure consists of a sequence of self- and cross- attention modules.

We first aggregate coarse features F_c for each point cloud with a geometric-aware self-attention block. For attention mechanism, the query $\mathbf{Q}$, key $\mathbf{K}$ and value $\mathbf{V}$ by projecting input $\mathbf{x}_i$,$\mathbf{x}_j$ with learnable weight matrixs $\mathbf{W}^{\mathbf{Q}}, \mathbf{W}^{\mathbf{K}}, \mathbf{W}^{\mathbf{V}} \in \mathbb{R}^{b \times b}$. The

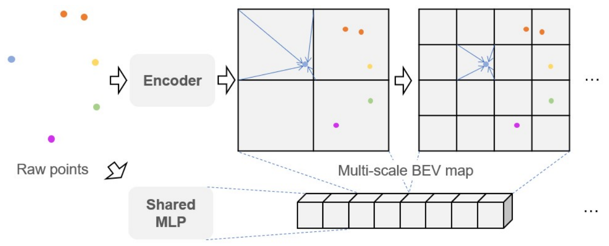

Fig. 2. Grid set abstraction based decoder. The middle results from convolution backbone are in the form of sparse 2D vectors. We convert the vectors into dense BEV maps and retract point features via bilinear interpolation. The decoder also includes a standalone point branch for compensation. In our pipeline, the point branch is integrated with MLP in pillar embedding layer.

self attention matrix $\mathbf{A}_{SA}$ can be expressed as $\mathbf{A}_{SA} = \mathbf{Q}(\mathbf{K}+\mathbf{G})^T/\sqrt{b}$, where $\mathbf{G} = \mathbf{g}\cdot\mathbf{W}^{\mathbf{G}}$ is projected geometric embedding. The output feature $F_{SA}^{\mathcal{P}}$ is calculated by $F_{SA}^{\mathcal{P}} = \mathbf{A}_{SA}\cdot\mathbf{V}$, where $\mathbf{A}_{SA}$ is first normalized by softmax function as in [19]. The updated feature $F_{SA}^{\mathcal{Q}}$ is calculated by the same formula. The proposed geometric embedding $\mathbf{g}$ serves as positional hints to enhance the self attention. For each point $\mathbf{x}$, its neighborhood embedding $\mathbf{g}$ is constructed by its nearest k neighbours $\{\mathbf{x_1}, ..., \mathbf{x_k}\}$.

$$\mathbf{g} = \mathrm{MLP}(\mathrm{Concat}[\|\mathbf{x}_i - \mathbf{x}\|, \angle(\mathbf{x}_i - \overline{\mathbf{x}}, \mathbf{x} - \overline{\mathbf{x}})/(2\pi)]). \quad (1)$$

The geometric embedding vector $\mathbf{g}$ is concatenated by Euclidean distance between $\mathbf{x}$ and its neighbours and angles with neighbourhood center $\overline{\mathbf{x}}$ as reference point.

The attention matrix of cross attention block is similarly computed as $\mathbf{A}_{CA}^{\mathcal{P}} = \mathbf{Q}^{\mathcal{P}}(\mathbf{K}^{\mathcal{Q}})^T/\sqrt{b}$, ,where we use subscripts $:^{\mathcal{P}}$,$:^{\mathcal{Q}}$ to distinguish variables from two point clouds. The output cross attention feature $F_{CA}^{\mathcal{P}}$ is calculated as $F_{CA}^{\mathcal{P}} = \mathbf{A}_{CA}^{\mathcal{P}}\cdot\mathbf{V}^{\mathcal{Q}}$, where attention matrix is also first normalized by softmax function. The attention matrix $\mathbf{A}_{CA}^{\mathcal{Q}}$ and cross attention feature $F_{CA}^{\mathcal{Q}}$ is calculated in the symmetric formula.

After performing inter- and intra- point cloud attention aggregation, we got aggregated coarse-level feature $F_c^{\prime\mathcal{P}}$ and $F_c^{\prime\mathcal{Q}}$. The entries of similarity matrix $S' \in \mathbb{R}^{|\mathcal{P}|\times|\mathcal{Q}|}$ is computed as dual Softmax function of patch features. We refer readers to [19] for details. We select top N_c entries of similarity matrix S' with minimum values as confident sparse correspondences.

Anchor Correspondence Generation Points in overlapping pillars tend to have larger spatial distribution compared to points in the knn-grouped node stated in previous works[27, 19]. Therefore, we propose to sample anchor points inside sparse correspondences and adopt virtual correspondence generation inspired by HRegNet[16]. Fig. 3 shows the scheme for generating a pair of anchors. First for each selected coarse pair $(P^{\mathcal{P}}, P^{\mathcal{Q}})$ in the coarse matching stage, we sample max to M_0 anchor points $\mathbf{X_q} \subset P^{\mathcal{Q}}$ per pair. Then for each anchor point $x_q \in \mathbf{X_q}$, we search for M_1 nearest points in feature space as candidates.

Next, we incorporate original point feature F_p, feature similarity F_d and structure similarity F_s to learn the attentive weights of candidate points. For calculation of feature similarity F_d, we use common cosine similarity $F_d =< F_p, F_p^i > /(|F_p| \cdot |F_p^i|)$. For calculation of structure similarity, we search for K nearest neighbours of anchor points $\mathbf{x}_i \in \mathbf{X_q}$ in sampled anchor points, denoted as $\{\mathbf{x}_k\}_{k=1}^{K} \subset \mathbf{X_q}$. We calculate the distances between x_i and x_i^j and also distances between their candidate points x_i^m, x_k^n. The entry d_{imk} of final structure similarity F_d is calculated by

$$d_{imk} = \min_{n} |\|\mathbf{x}_i - \mathbf{x}_k\|_2 - \|\mathbf{x}_i^m - \mathbf{x}_k^n\|_2|. \tag{2}$$

Finally, we concatenate F_p, F_d, F_s and input them into a three-layer shared-MLP as HRegNet and SDMNet[16, 15]. A maxpool and softmax layer is then applied to get final weights w_{ik} of candidate points. The final virtual correspondence $\mathbf{x}_i' \in \mathbf{X_v}$ is calculated as weighted sum of candidate points $\mathbf{x}_i' = \sum_k w_{ik}\mathbf{x}_k$. In addition, MLP and sigmoid function is applied to learned weights w_{ik} to get confidence score c_i for each pair of $(\mathbf{x}_i, \mathbf{x}_i')$. In testing stage, we utilize learned confidence score to filter anchor correspondences with confidence score below a predefined threshold c_0.

The output coordinates and confidence score of anchor correspondence generation module is respectively regularized by classification loss and probabilistic distance.

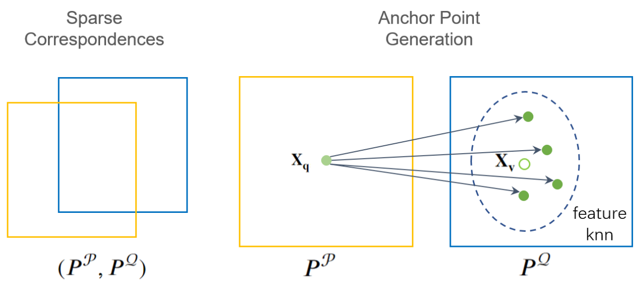

Fig. 3. Scheme for anchor correspondence generation.

Fine Matching At the fine-scale level, we seek for learning point-wise correspondence using optimal transport algorithm [20].

For each anchor correspondence $(\mathbf{x}, \mathbf{x}')$, we group knn neighbours of both into mini patches and carry out local fine matching. The initial batched similarity matrix is constructed by calculation of unnormalized cosine similarity $s_{ij}' =< F_{\mathbf{x},i}, F_{\mathbf{x}',j} > /\sqrt{b}$ where $F_{\mathbf{x},i}$ denotes feature of ith neighbour of $\mathbf{x}$ and likewise for $F_{\mathbf{x}',j}$, b is the feature dimension length of $F_{\mathbf{x}}$. The feature similarity matrix is enhanced with an additional row and column serving as dustbins as in [20] and denoted as $\tilde{S}'$. Then Sinkhorn algorithm [24] is applied to solve the extended assignment matrix $\tilde{Z}$ on $\tilde{S}'$. The final assignment matrix Z is obtained by discarding the last row and column of $\tilde{Z}$. Finally, same as coarse-scale matcher, entries of S_{ij}' with top k maximum similarity are chosen as final point correspondences. We refer readers to [20] for more details.

3.3 Pose Estimator

Finally, we adopt parametric-free weighted SVD algorithm [3] as our pose estimator in a iterative proposal and verification scheme. We calculate a set of $\mathbf{R}_i, \mathbf{t}_i$ for each mini-patch generated from confident anchor correspondences as N_c' proposals. For each iteration in pose verification, we select $\mathbf{R}_i, \mathbf{t}_i$ with most inliers whose residuals are less than a predefined distance threshold d_0 and use the inlier points as input of next iteration. Experiments show that the pose estimator can guarantee accuracy while outperforming RANSAC on efficiency.

3.4 Loss Function

For the supervision of coarse matching stage, an overlap-aware circle loss function from [19] is utilized. Each pair of patch correspondence is seen as positive sample only if point correspondences account for at least 10%. We denote patch from point cloud $\mathcal{P}$ as $P^{\mathcal{P}}$ and its positive and negative correspondence from point cloud $\mathcal{Q}$ as $P^{\mathcal{Q}+}$ and $P^{\mathcal{Q}-}$. The overlap-aware circle loss is defined as following:

$$\mathcal{L}_c^{\mathcal{P}} = \frac{1}{|\mathcal{N}|} \sum_{P^{\mathcal{P}} \in \mathcal{N}} \log[1 + \sum_{P^{\mathcal{Q}+}} e^{\sqrt{\lambda}\beta^+(d-\Delta^+))} \cdot \sum_{P^{\mathcal{Q}-}} e^{\beta^-(\Delta^- - d))}], \tag{3}$$

where λ represents the overlap ratio, d is feature space distance. $\beta^+ = \gamma(d - \Delta^+)$ and $\beta^- = \gamma(\Delta^+ - d)$ are weights for positive samples and negative samples respectively. Loss $\mathcal{L}_c^{\mathcal{Q}}$ is calculated in the same way and final loss is $\mathcal{L}_c = (\mathcal{L}_c^{\mathcal{P}} + \mathcal{L}_c^{\mathcal{Q}})/2$.

For anchor generation stage, the coordinates $(\mathbf{x}_q, \mathbf{x}_v)$ and confidence c of generated anchor correspondences is supervised by distance loss and classification loss borrowed from [15].

$$\mathcal{L}_{a1} = \sum_i (\ln(\alpha - c_i) + \frac{d}{\alpha - c_i}), d = \|\mathbf{R}_{gt}\mathbf{x}_q + \mathbf{t}_{gt} - \mathbf{x}_v\|_2, \tag{4}$$

$$\mathcal{L}_{a2} = BCE(c, \exp(-d/d_0)), d = \|\mathbf{R}_{gt}\mathbf{x}_q + \mathbf{t}_{gt} - \mathbf{x}_v\|_2. \tag{5}$$

Fine-level loss is a negative log-likelihood loss defined on ground truth correspondence set $\mathcal{M}$ with matching radius τ and unmatched point sets $\mathcal{I}$,$\mathcal{J}$ from two point clouds as in [20, 19].

$$\mathcal{L}_f = -\sum_{(i,j)\in\mathcal{M}} \log \tilde{Z}_{ij} - \sum_{i\in\mathcal{I}} \log \tilde{Z}_{i,N+1} - \sum_{j\in\mathcal{J}} \log \tilde{Z}_{M+1,j}, \tag{6}$$

where $\tilde{Z} \in \mathbb{R}^{(M+1)\times(N+1)}$.

The overall loss is $\mathcal{L} = \lambda_c\mathcal{L}_c + \lambda_{a1}\mathcal{L}_{a1} + \lambda_{a2}\mathcal{L}_{a2} + \lambda_f\mathcal{L}_f$.

4 Experiments

4.1 Implementation Details

We implement our pipeline with a GTX 2080Ti GPU and a Intel Xeon Platinum 8255C @ 2.50GHz CPU. We train our model for 15 epochs using an Adam optimizer with an initial learning rate $1e-4$ and an exponential decaying rate 0.9. The batch size is set to 1 for all experiments. The width for pillar embedding is set to 0.3m. In both training and testing phases, we select up to 128 coarse correspondences but accept those only with overlap ratios > 0.1 for training phase. We randomly sample 12 anchor points per patch and select 4 candidates for each anchor points in generation of virtual correspondence. For data augmentation, we apply random rotation within $4°$ in pitch and roll and within $360°$ in yaw, and random translation within 0.5m and random scale within [0.95, 1.05].

We evaluate our pipeline and the baselines on two large-scale urban datasets: KITTI Odometry dataset [9] and NuScenes dataset [4]. We follow the split settings of [16] to organize point cloud pairs and refine the point cloud pair poses from original KITTI Odometry dataset with ICP registration.

4.2 Results on KITTI and NuScenes

We evaluate registration result with three metrics following[6, 27]. The first is Relative Rotation Error (RRE), *i.e.*, the geodesic distance between estimated and ground truth rotation matrices $\hat{\mathbf{R}}$ and $\mathbf{R}_{gt}$, which is computed as $RRE = \arccos(Tr(\mathbf{R}_{gt}^{-1}\hat{\mathbf{R}} - I)/2)$. The second is Relative Translation Error (RTE), the Euclidean distance between estimated and ground truth translation $\hat{\mathbf{t}}$ and $\mathbf{t}_{gt}$, which is $RTE = \|\mathbf{t}_{gt} - \hat{\mathbf{t}}\|_2$. And the last is Registration Recall (RR), defined as the fraction of point cloud pairs with RRE and RTE below certain thresholds. Following prior works [6, 2, 27, 19, 10], we set the threshold as RRE$< 5°$ and RTE$< 2m$ throughout.

Regarding baselines, we compare our methods with both axiomatic methods **FGR**[29], **RANSAC**[8] and learning-based methods **DGR**[5], **HRegNet**[16], and **GeoTrans**[19]. Part of results are borrowed from [16]. Apart from the three metrics, we also report the mean inference time (in seconds) on the test point cloud pairs. Note that DGR and HRegNet distinct from the rest as they explicitly take ground truth transformations into loss function and regress transformation.

Table 1. Registration results on KITTI and Nuscenes dataset.

Model	KITTI dataset				NuScenes dataset			
	RRE(°)	RTE(m)	RR(%)	T(s)	RRE(°)	RTE(m)	RR(%)	T(s)
FGR	0.96± 0.81	0.93± 0.59	39.43	0.506	1.01± 0.92	0.71± 0.62	32.24	0.285
RANSAC	0.54± 0.40	0.13± 0.70	91.90	0.550	0.74± 0.70	0.21± 0.19	60.86	0.268
DGR	0.37± 0.30	0.32± 0.32	98.71	1.497	0.48± 0.43	0.21± 0.18	98.4	0.523
HRegNet	**0.18± 0.08**	0.056± 0.075	99.77	0.106	**0.27± 0.20**	**0.12± 0.11**	**100.0**	0.087
GeoTrans	0.18± 0.13	0.063± 0.045	**100.0**	0.368	OOM[1]			
Ours	0.26± 0.20	**0.046± 0.034**	**100.0**	**0.092**	0.31± 0.25	0.14± 0.13	**100.0**	**0.082**

Table 1 shows that though relative rotation error of our work is in midstream, our method achieves comparable performance to state-of-art methods in registration

[1] Out of memory during training phase.

Table 2. Generalized registration results on Apollo SouthBay Dataset.

Sequence	Model	RRE(°)	RTE(cm)	RR(%)	T(s)
SanJoseDowntown	HRegNet	0.15± 0.34	9.9± 22.2	90.0	<u>0.108</u>
	GeoTrans	<u>0.13± 0.20</u>	<u>5.5± 8.9</u>	<u>99.9</u>	0.321
	Ours	**0.13± 0.15**	**3.2± 4.8**	**100.0**	**0.092**
BaylandsToSeafood	HRegNet	0.37± 0.67	29.6± 42.9	81.4	<u>0.108</u>
	GeoTrans	**0.11± 0.14**	**8.8± 10.2**	**93.6**	0.293
	Ours	<u>0.17± 0.25</u>	<u>9.9± 13.9</u>	<u>93.3</u>	**0.092**

recall and relative translation error. Compared with the state-of-art keypoint-free approach [19], our runtime is significantly smaller. Moreover, in the case of NuScenes dataset, we observe an out-of-memory breakdown in implementing **GeoTrans**, while **Ours** deliver the best or second best results in comparison. Compared with the state-of-art keypoint-free approach **HRegNet**, the recall rate of the proposed method is higher on KITTI dataset and it is proved in Section 4.3 that our performance downgrades less than **HRegNet** on unseen datasets. The runtime of our method achieves best on both datasets.

4.3 Generalization to Apollo Southbay dataset

We further test the generalization ability of our method and the baselines on the 'SanJoseDowntown' and 'BaylandsToSeafood' sequence from Apollo SouthBay Dataset [17]. The test point cloud pairs are constructed in the same way as NuScenes. For each method, we directly apply pretrained weight trained on KITTI Odometry dataset.

Table 2 reports the generalization performance. When encountering unseen dataset, **HRegNet** suffers a significant drop in registration recall while detection-free works like **GeoTrans** and ours still perform well.

4.4 Ablation Study

We conduct ablation experiments to study the effectiveness of the key component within our pipeline. Table 3 shows ablation result on different modules. We directly initialize the candidates points for anchors in the whole point cloud range to ablate coarse matching. We test the setting similar to [19] where no anchor points are required and dense points are randomly sampled inside patches to ablate anchor generation. We ablate fine matching by comparing the estimated pose using filtered anchor correspondences with original estimated poses. Results show that our individual modules keep the our performance the best.

The selection of hyperparameters, particularly selected sparse correspondence numbers N_c and max anchor points per pair N_a has influence on final result. We change N_c and N_a to search for a balanced setting. Table 4 shows the impact of sampling density. The accuracy increases with density while runtime decreases.

Table 3. Comparison of our modules on KITTI dataset.

Model	IR(%)	RRE(°)	RTE(cm)	RR(%)	T(s)
ours	**96.1**	**0.26**	**4.6**	**100**	0.092
w/o coarse	86.7	1.18	23.6	73	0.101
w/o anchor	87.8	0.36	5.9	**100**	0.109
w/o fine	92.4	0.31	8.2	**100**	**0.081**

Table 4. Comparison of different sampling density on KITTI dataset.

Hyperparameters	PIR(%)	IR(%)	RRE(°)	RTE(cm)	RR(%)	T(s)
$N_c = 256, N_a = 12$	97.7	95.5	0.246	**4.2**	**100**	0.103
$N_c = 128, N_a = 12$	98.9	96.1	0.258	4.6	**100**	0.092
$N_c = 64, N_a = 12$	**99.2**	**96.5**	**0.284**	5.1	**100**	0.086
$N_c = 128, N_a = 16$	98.9	95.9	0.260	4.6	**100**	0.098
$N_c = 128, N_a = 8$	98.9	96.2	0.259	4.6	**100**	**0.084**

We also ablate the choice of superpoint representation in feature extractor. Table 5 shows that superpoint in pillar form achieves a little inferior performance in exchange for 30% improvement in inference time.

Table 5. Comparison of different representations on KITTI dataset.

Setting	Length	RRE(°)	RTE(cm)	RR(%)	T(s)	Model Size
Voxel	0.3m	**0.244**	**4.3**	**100**	0.131	11.2M
Pillar	0.3m	0.259	4.6	**100**	0.092	**5.4M**
Pillar	0.5m	0.291	5.1	**100**	**0.086**	**5.4M**

5 Conclusion

We have proposed an efficient network for point cloud registration in the urban scenes. By utilizing efficient feature extraction modules and hierarchical matching strategy, we make a step towards real-time robust deep learning based point cloud registration. Experiments show that our pipeline, being compact and light, achieves comparable inference speed and accuracy with state-of-art methods in urban scenes.

In the future work, we will further look into works like [13] and study the potential of our pipeline on multi-task learning. Our pipeline is suitable for more universal scene understanding by integration with other downstream task such as object detection and semantic segmentation.

References

1. Ao, S., Hu, Q., Yang, B., Markham, A., Guo, Y.: SpinNet: Learning a General Surface Descriptor for 3D Point Cloud Registration. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 11748–11757 (2021)
2. Bai, X., Luo, Z., Zhou, L., Fu, H., Quan, L., Tai, C.L.: D3Feat: Joint Learning of Dense Detection and Description of 3D Local Features. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 6358–6366 (2020)
3. Besl, P.J., McKay, N.D.: A Method for registration of 3-D shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence **14**, 239–256 (1992)
4. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: NuScenes: A multimodal dataset for autonomous driving. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 11618–11628 (2020)
5. Choy, C.B., Dong, W., Koltun, V.: Deep Global Registration. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 2511–2520 (2020)
6. Choy, C.B., Park, J., Koltun, V.: Fully Convolutional Geometric Features. In: IEEE International Conference on Computer Vision. pp. 8957–8965 (2019)
7. Fischer, K., Simon, M., Olsner, F., Milz, S., Gross, H.M., Mader, P.: Stickypillars: Robust and efficient feature matching on point clouds using graph neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 313–323 (2021)
8. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM **24**(6), 381–395 (1981)
9. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The KITTI vision benchmark suite. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 3354–3361 (2012)
10. Huang, S., Gojcic, Z., Usvyatsov, M.M., Wieser, A., Schindler, K.: PREDATOR: Registration of 3D Point Clouds with Low Overlap. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 4265–4274 (2021)
11. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: PointPillars: Fast Encoders for Object Detection From Point Clouds. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 12689–12697 (2019)
12. Li, J., Lee, G.H.: USIP: Unsupervised Stable Interest Point Detection From 3D Point Clouds. In: IEEE International Conference on Computer Vision. pp. 361–370 (2019)
13. Liu, C.J., Guo, J., Yan, D., Liang, Z., Zhang, X., Cheng, Z.L.: SARNet: Semantic Augmented Registration of Large-Scale Urban Point Clouds (2022), arXiv preprint arXiv:2206.13117
14. Liu, Z., Tang, H., Lin, Y., Han, S.: Point-Voxel CNN for Efficient 3D Deep Learning. In: Advances in Neural Information Processing Systems. pp. 963–973 (2019)
15. Lu, F., Chen, G., Liu, Y., Zhan, Y., Li, Z., Tao, D., Jiang, C.: Sparse-to-Dense Matching Network for Large-scale LiDAR Point Cloud Registration. IEEE Transactions on Pattern Analysis and Machine Intelligence pp. 1–13 (2023)
16. Lu, F., Chen, G., Liu, Y., Zhang, L., Qu, S., Liu, S., Gu, R.: HRegNet: A Hierarchical Network for Large-scale Outdoor LiDAR Point Cloud Registration. In: IEEE International Conference on Computer Vision. pp. 15994–16003 (2021)
17. Lu, W., Zhou, Y., Wan, G., Hou, S., Song, S.: L3-net: Towards learning based lidar localization for autonomous driving. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 6382–6391 (2019)
18. Magnusson, M., Lilienthal, A.J., Duckett, T.: Scan registration for autonomous mining vehicles using 3D-NDT. Journal of Field Robotics **24**(10), 803–827 (2007)

19. Qin, Z., Yu, H., Wang, C., Guo, Y., Peng, Y., Xu, K.: Geometric Transformer for Fast and Robust Point Cloud Registration. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 11133–11142 (2022)
20. Sarlin, P.E., DeTone, D., Malisiewicz, T., Rabinovich, A.: SuperGlue: Learning Feature Matching With Graph Neural Networks. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 4937–4946 (2020)
21. Segal, A.V., Hähnel, D., Thrun, S.: Generalized-ICP. In: Robotics: Science and Systems (2009)
22. Shi, G., Li, R., Ma, C.: PillarNet: Real-Time and High-Performance Pillar-based 3D Object Detection. In: European Conference on Computer Vision (2022)
23. Shi, S., Guo, C., Jiang, L., Wang, Z., Shi, J., Wang, X., Li, H.: PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 10526–10535 (2020)
24. Sinkhorn, R., Knopp, P.: Concerning nonnegative matrices and doubly stochastic matrices. Pacific Journal of Mathematics **21**, 343–348 (1967)
25. Tang, H., Liu, Z., Zhao, S., Lin, Y., Lin, J., Wang, H., Han, S.: Searching Efficient 3D Architectures with Sparse Point-Voxel Convolution. In: European Conference on Computer Vision. vol. 12373, pp. 685–702 (2020)
26. Thomas, H., Qi, C., Deschaud, J.E., Marcotegui, B., Goulette, F., Guibas, L.J.: KPConv: Flexible and Deformable Convolution for Point Clouds. In: IEEE International Conference on Computer Vision. pp. 6410–6419 (2019)
27. Yu, H., Li, F., Saleh, M., Busam, B., Ilic, S.: CoFiNet: Reliable Coarse-to-fine Correspondences for Robust Point Cloud Registration. In: Advances in Neural Information Processing Systems. vol. 29, pp. 23872–23884 (2021)
28. Zeng, A., Song, S., Nießner, M., Fisher, M., Xiao, J., Funkhouser, T.A.: 3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions. IEEE Conference on Computer Vision and Pattern Recognition pp. 199–208 (2017)
29. Zhou, Q.Y., Park, J., Koltun, V.: Fast Global Registration. In: European Conference on Computer Vision. pp. 766–782 (2016)
30. Zhou, Y., Sun, P., Zhang, Y., Anguelov, D., Gao, J., Ouyang, T.Y., Guo, J., Ngiam, J., Vasudevan, V.: End-to-End Multi-View Fusion for 3D Object Detection in LiDAR Point Clouds. In: Conference on Robot Learning. vol. 100, pp. 923–932 (2019)