MagneticPillars: Efficient Point Cloud Registration through Hierarchized Birds-Eye-View Cell Correspondence Refinement

Kai Fischer[1,3] Martin Simon[1] Stefan Milz[2,3] Patrick Mäder[3]
[1]Valeo Schalter und Sensoren GmbH [2]Spleenlab GmbH [3]Ilmenau University of Technology
{kai.fischer, martin.simon}@valeo.com {stefan.milz, patrick.maeder}@tu-ilmenau.de

Abstract

Recent point cloud registration approaches often deal with a consecutive determination of coarse and fine feature correspondences for hierarchical pose refinement. Due to the unordered nature of point clouds, a common way to generate a subsampled representation for the coarse matching step is by applying 3D-sensitive convolution approaches. However, expensive grouping mechanisms such as nearest neighbour search have to be used to determine the associated fine features, generating individual associations for each point cloud and leading to an increased overall runtime. Furthermore current methods often tend to predict deficient point correspondences and rely on additional filtering by expensive registration backends like RANSAC impeding their application in time critical systems.

To overcome these challenges, we present MagneticPillars utilizing a Birds-Eye-View (BEV) grid representation, entailing fixed affiliations between coarse and fine feature cells. We show that by extracting correspondences in this manner, a small amount of key points is already sufficient to achieve an accurate pose estimation without external optimization methods like RANSAC. We evaluate our approach on two autonomous driving datasets for the task of point cloud registration by applying SVD as the backend, where we outperform recent state-of-the-art methods, reducing the rotation and translation error by 12% and 40%, respectively, and to top it all off, cutting runtime in half.

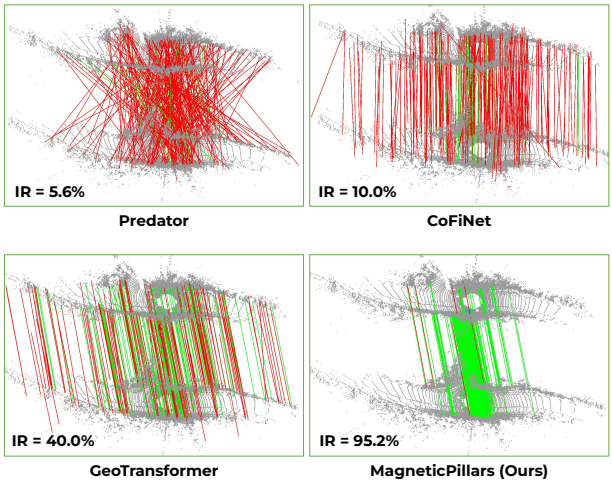

Figure 1. MagneticPillars can produce strong key points and correspondences, correct (green) and false (red), which can directly be used to perform an accurate and efficient pose estimation even via basic methods like SVD, unlike existing methods.

1. Introduction

Localization or Ego-Pose-Estimation (Odometry) is an essential task for autonomous vehicles which at any time have to be aware of their exact location relative to a global or local reference point or mapless position. One way to achieve the desired pose estimation is to establish correspondences between a source and a target measurement, e.g., via LiDAR sensors, which are widely used due to the direct determination of depth information from the surrounding environment. However, unlike camera images, LiDAR point clouds exist in an unordered fashion, implying particular challenges in their data processing.

A recent approach to this problem is presented in [24], which is applied as a feature extractor for various tasks like object classification, 3D semantic segmentation and point cloud registration. Here, deformable 3D convolutional kernels are applied to generate a sub-sampled representation of the point cloud analogous to 2D convolutions encoding information on image data. Point cloud registration methods often utilize the downsampled features to establish preliminary connections between the source and target frame, providing first indications about the alignment of the clouds. By a subsequent upsampling of the clouds, associated finer point features are determined for the coarse super points in order to refine the point correspondences on a local level. Although 3D filtering of the point cloud is able to capture expressive structural cues, generating robust feature

descriptors, it also involves a high computational effort due to the unique unordered shape of each point cloud. Moreover, this results in additional computational complexity in terms of identifying the associations between the coarse and fine feature vectors where expensive grouping strategies are applied consuming a major part of the total runtime. For instance in the case of GeoTransformer [20] around 50% (120ms) of the average computation time (245ms) is consumed by the coarse to fine feature determination. Furthermore recent methods tend to predict erroneous point correspondences as visualized in Figure 1, which are filtered by expensive registration methods like RANSAC again leading to an enormous increase in computation time, unsuitable for integration into time critical autonomous driving systems.

To overcome these challenges, we present MagneticPillars, which extracts robust features from a BEV representation of the source and target clouds. Processing a rasterized version of the point clouds holds the benefit of fixed cloud representations and associations between coarse and fine cell features, leading to a much more efficient correspondence search. Especially for our use case utilizing autonomous driving datasets, where point clouds are mainly captured using radial laser scanners in an outdoor environment, using a BEV is feasible since points arranged in height more likely scan the same object.

Following the basic idea of previous works [14,20,29] of coarse pre-filtering with subsequent correspondence refinement on a local level, we utilize a UNet-like [21] structure as the base architecture providing the fundamental operations for extracting the desired coarse and fine BEV features for our cell-based correspondence search. We show that calculating pillar centroids for the final fine cell correspondences represents strong key points that can be directly fed into Singular Value Decomposition (SVD) for an accurate pose estimation without the need for optimization techniques like RANSAC. More precisely, our main contributions are as followed:

- A novel approach for point cloud registration based on hierarchized cell correspondence refinement
- An efficient processing pipeline for point cloud encoding and decoding in 2D space with a pose estimation based on 3D pillar centroids
- Novel fine and coarse grid cell matching losses guiding a robust match candidate pre-filtering and subsequent correspondence refinement
- Pose estimation via a minimal amount of robust matches without the need of iterative optimization methods like RANSAC which is crucial for real time application requirements like autonomous driving
- Wide range of experiments on two autonomous driving datasets outperforming current state-of-the-art methods in terms of pose estimation accuracy and runtime

2. Related Work

2.1. Direct methods

Direct point cloud registration methods are designed to predict the pose between the source and target input cloud in the form of a translation vector and a rotation matrix in an end-to-end fashion. Starting with ICP [4] and its successors, where the desired transformation is iteratively refined based on optimizing soft correspondences. Works like [19] build up on this concept by applying machine learning and generating corresponding points in the respective other cloud. [8] applies a UNet structure for inlier point prediction followed by weighted procrustes based on an initial transformation proposal with a gradient-based optimizer for pose refinement. Contrary [7] proposes an alternate registration backend for correspondence prediction networks based on a second-order spatial compatibility for matching analysis. [1, 6, 18] employ a deep hierarchical feature embedding for an initial transformation estimation on a coarse level with subsequent refinement on upsampled features. [15] aims to minimize a feature-metric projection error within the training process in a semi-supervised way, implicitly guiding the clouds to overlap. Finally, FINet [27] proposes a dual-branch network for a separate prediction of translation and rotation-attentive features interacting at different scales.

2.2. Feature matching methods

Feature matching methods aim to extract feature descriptors from the respective input clouds, which are subsequently matched by applying optimization techniques like RANSAC. In this context, D3Feat [3] uses KPConv [24] to predict dense, per point features and detection scores in a joint learning process, while SpinNet [2] explicitly focuses on rotation invariant feature extraction, converting the input cloud surfaces into a carefully designed cylindrical space with subsequent local pattern extraction. 3DFeat-Net [28] proposes a weakly supervised approach, using a three-branch siamese network, determining a triplet loss based on anchor, positive, and negative input clouds. FCGF [9] generates features solely using 3D convolutional layers deploying a ResUNet architecture. A more recent approach in [14] tackles the challenge of point cloud registration with low overlap by explicitly guiding the network to learn the specific overlap regions via attention modules.

2.3. Correspondence matching methods

In contrast, correspondence-based methods directly predict point matches between the respective input clouds with a certain confidence, which can be directly used for pose

estimation in solvers like SVD significantly reducing registration runtime. Here [10–12] aim to find correspondences between extracted key points by feature generation and matching via graph neural networks. FIRE-Net [26] proposes a combined feature encoder for interactive local and global feature extraction via graph generation and filtering. CoFiNet [29] conducts a coarse to fine feature matching based on a KPConv encoder-decoder architecture. GeoTransformer [20] extends this idea by including an overlap-aware circle loss and geometric transformers for a more distinct super-point matching. In [16], the authors use KPConv solely for feature encoding with a transformer-only backend for registration. Lepard [17] disentangles the cloud encoding in a separated feature and positional space, dealing with cloud matching in rigid and deformable scenes. In contrast, [23] is dealing with unsupervised point cloud registration by evaluating inliers via the geometric difference between a source and a pseudo-target neighbourhood.

While many methods use 3D processing frontends like KPConv [3, 16, 17, 20, 29] to encode the necessary point cloud features, we show that reducing the matching problem to a cell correspondence search via coarse pre-filtering and subsequent high-level refinement is sufficient for an efficient and robust feature matching.

3. Methodology

The overall processing pipeline of our MagneticPillars architecture is displayed in Figure 2. First the pillar feature aggregation module is used for feature extraction on the two input clouds, which are subsequently down-sampled by feature encoder layers to generate the coarse grid representation. Multiple self- and cross-attention layers are applied to determine coarse cell matches in between the clouds, representing initial guesses for correspondences which are later refined utilizing the fine feature vectors at the top layer of the feature decoder. Finally, pose estimation is performed by applying SVD on pillar centroids determined for each of the matched fine cells.

3.1. Problem Description

Given two sets of points P^K and P^L, the aim of point cloud registration is to determine a transformation matrix T_E which transforms P^L into the reference frame of P^K, resulting in overlapping clouds and thus minimizing the overall point-to-point distance. We define a voxel grid rasterization of P^K and P^L, namely $V^K, V^L \in \mathbb{R}^3$ with grid dimensions H, W, D as well as its projection to BEV $X^K, X^L \in \mathbb{R}^2$ with dimension H, W. Our goal is to establish grid correspondences of BEV key cells where the associated key points are derived by calculating the centroids $\tilde{\pi}^K, \tilde{\pi}^L \in \mathbb{R}^3$ based on the points assigned to the respective cells $\tilde{x}_h^K, \tilde{x}_h^L$, $h \in \{1, \dots, n_k\}$ with infinite height, yielding a rectangular pillar-shaped point aggregation. Here n_k denotes the number of extracted final cell correspondences and key points, respectively. Based on the pillar centroid matches, methods such as Singular Value Decomposition (SVD) can be applied to solve for the desired transformation T_E. Finally, the overall goal of our approach can be denoted as:

$$f_{\tilde{\pi}^L \rightarrow \tilde{\pi}^K}(P^L) = T_E \cdot P^L \approx P^K \tag{1}$$

3.2. Pillar Feature Aggregation

Assuming a fixed grid size of HxWxD, based on the voxel grid representations V^K and V^L for each cell the associated points of the original cloud are gathered. Based on the grouped points, the respective voxel centroids $v_i^K \in \mathbb{R}^3$ and $v_i^L \in \mathbb{R}^3$ are calculated and applied to transform the global point coordinates into local ones resulting in the following voxel cell feature vectors for cloud K, with $i \in \{1, \dots, n_v\}$ and n_v denoting the number of voxel per cloud:

$$f_i^K = \left\{ \left[(p_{i,j}^K - v_i^K), r_{i,j} \right], \dots \right\}, \quad j \in \{1, \dots, n_z\} \tag{2}$$

With n_z representing the maximum number of points per voxel, $p_{i,j}^K$ the corresponding 3D coordinates, and $r_{i,j}$ the reflectance value per point per voxel. To maintain a static grid representation, voxels with more or fewer point allocations than n_z are cut and zero-padded, respectively. Feature vector f_i^L and all other following determinations are performed analogously for cloud P^L.

Following the works and promising results of [10, 11], we chose a separate feature encoding for global and local cloud information. In this context, the multi-layer-perceptron (MLP) based positional-encoder (global) and pillar-encoder (local) are introduced. While the first one is designed to encode the global consistency of the point clouds by processing the voxel centroids v^K and v^L, the latter is constructing the finer local structures based on f_i^K and f_i^L. Eventually, the resulting final feature vectors g^K and g^L are obtained according to:

$$g_i^K = Pos(v_i^K) + Pil(f_i^K), \quad g_i^K \in \mathbb{R}^C \tag{3}$$

Where C denotes the output feature dimension. Note that the layers for the encoders are shared for both input clouds in order to generate a common feature composition. To establish a 2-dimensional grid representation, we accumulate all feature vectors g along the height dimension D, resulting in a BEV grid with a dimensionality of $H, W, C \cdot D$, which subsequently can be processed by conventional image processing methods. Additionally, in this context, a binary vector $B^K, B^L \in \{1, 0\}$ is calculated, which expresses the occupancy of each grid cell, holding 1 if the respective cell includes at least one point and 0 otherwise.

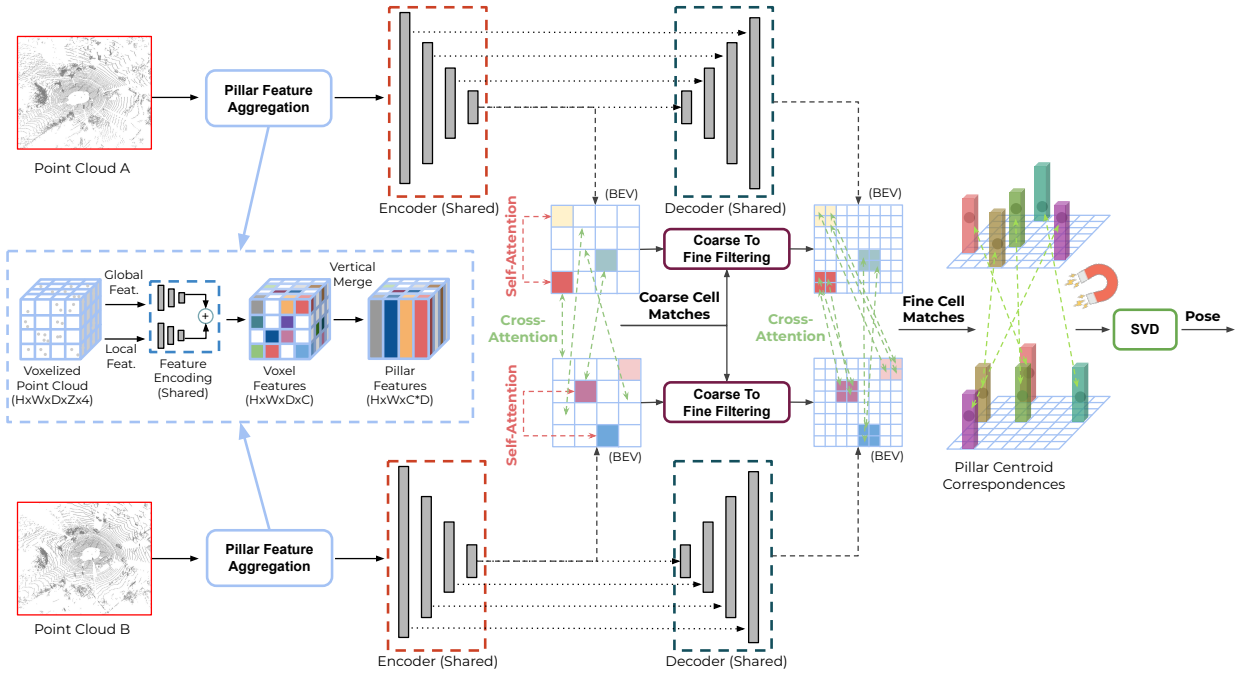

Figure 2. MagneticPillars processing pipeline consisting of: 1) Pillar feature aggregation, 2) Shared birds-eye-view feature downsampling and coarse cell match estimation, 3) Shared feature upsampling and fine cell match estimation, 4) Pillar centroid determination for the predicted fine grid cell correspondences, and 5) Cloud registration by pose estimation backend, e.g. Singular Value Decomposition (SVD).

3.3. Coarse Cell Matching

For our downsampling blocks, we follow the architecture of [30] utilizing double convolution layers followed by batch normalization and ReLU activation. Dimension reduction is performed by Max Pooling with a kernel size of 2 in each direction, bisecting the grid size with each layer. Assuming a network depth of n_l downsampling layers, the final grid size at the bottom of the encoder is defined by $H' = H/2^{n_l}$, $W' = W/2^{n_l}$ for each of the two input clouds with a feature dimension of C'. In order to determine correspondences between the coarse cells, multiple self- and cross-attention layers [25] are applied onto the $n_d = H' \cdot W'$ coarse feature descriptors of each point cloud, which are subsequently normalized using optimal transport [22], resulting in score matrix S_c. Based on the calculated matching scores, the n_c highest ranked coarse cell correspondences are selected for further processing, which helps to reduce the general computational complexity.

3.4. Fine Cell Matching

Subsequently, to recover the original grid size, upsampling is conducted n_l times by double convolutional layers, batch normalization, and Relu activation, as well as bilinear interpolation for grid enlargement. The output of the top layer holds the original grid size of H, W with a feature dimension of C'' representing the final fine cell descriptors. Due to the linear down and up-sampling strategy, the $2^{n_l} \cdot 2^{n_l}$ fine features for the n_c coarse cells of each cloud can be directly gathered on which a single cross-attention layer is applied to generate the fine feature correspondences. Note that in this step, the respective fine feature vectors are matched based on their coarse correspondences, again reducing the overall complexity. This leads to a set S_f of n_c fine score matrices with dimension $n_f \times n_f$ with $n_f = 2^{2n_l}$. After applying optimal transport for normalization, the scores for unoccupied cells according to B^K, B^L are multiplied by 0 to suppress invalid cells. Finally, the n_k highest ranked matching scores are selected as fine cell correspondences of $\tilde{x}^K, \tilde{x}^L$, which build the base for the following key point generation.

3.5. Pose Estimation via Pillar Centroid Correspondences

The final 3D points $\tilde{\pi}^K$ and $\tilde{\pi}^L$ used for pose estimation are determined based on the predicted n_k fine cell matches for each cloud. In this context, the n_z points per voxel gathered from the original cloud for the pillar feature aggregation are used to determine the corresponding pillar centroid for each cell. Here the respective voxels are stacked in the

height dimension, resulting in a total of $D \cdot n_z$ points per cell upon which the centroids can be determined. Note that the zero-padded entries of the voxels are neglected during the process of the key point calculation. Finally, the n_k extracted pillar centroids and the correspondence scores from the fine cell matching can be used to determine the desired pose via weighted SVD.

3.6. Loss Functions

Inspired by previous works like [20, 29], our loss function is built upon two terms targeting the coarse $\mathcal{L}_c$ and fine $\mathcal{L}_f$ feature matching, respectively, balanced by a factor λ:

$$\mathcal{L} = \mathcal{L}_c + \lambda \mathcal{L}_f \tag{4}$$

3.6.1 Fine Matching Loss

The fine matching loss is calculated utilizing Negative Log Likelihood on the set of fine score matrices S_f with weighting W_f. Contrary to existing methods, in our 2D use case the loss function is applied on the $n_c \cdot n_f$ pillar centroids π^K, π^L in order to express the affiliations of the corresponding fine feature cells. In this context T_{GT} is applied to determine the respective nearest neighbours, filling W_f with either 1 if the nearest neighbour's distance is below a certain threshold τ_f and 0 otherwise. Finally, the fine matching loss can be formulated as followed:

$$\mathcal{L}_f = -\frac{1}{n_c \cdot n_f} \sum_{b=1,k=1,l=1}^{n_c,n_f+1,n_f+1} W_f(b,k,l) \cdot \log(S_f(b,k,l)) \tag{5}$$

$$W_f(b,k,l) = \begin{cases} 1, & \text{if } ||(T_{GT} \cdot \pi^L_{b,l}) - \pi^K_{b,k}|| < \tau_f, \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

In case no match was determined, additional dustbin dimension entries $W_f(b, n_f + 1, l)$ and $W_f(b, k, n_f + 1)$ are assigned to 1.

3.6.2 Coarse Matching Loss

The coarse matching loss is calculated in a similar way to $\mathcal{L}_f$, however instead of applying a binary weight matrix to the coarse score matrix S_c, we propose a soft weighting matrix W_c since we found that it benefits the training process and final network performance. A comparison between soft and hard weighting within the coarse loss function is depicted in Section 4.2.3.

Therefore contrary to inspecting the sole existence of a nearest neighbour in the respective other cloud for the coarse cells, we aim to find the amount of nearest neighbours for the fine cells associated to each coarse cell.

We define two sets of voxel centroids v^K, v^L, their corresponding fine cells x^K and x^L when projected to BEV, as well as their affiliations to the coarse cell grids $\bar{x}^K_m, \bar{x}^L_m, m \in \{1, \ldots, n_d\}$ with $n_d = H' \cdot W'$. Applying the ground truth transformation matrix on v^L so that $T_{GT} \cdot v^L \approx v^K$ the fine nearest neighbour cells in the respective other cloud $x^{K \to L}_i$ and $x^{L \to K}_i$ as well as their affiliations to the coarse scale $\bar{x}^{K \to L}_m, \bar{x}^{L \to K}_m$ can be determined by applying a threshold of $\tau_c = 0.3m$. Assuming a network depth of n_l, here the maximum number of possible fine nearest neighbour matches for a coarse cell is 2^{2n_l}. Finally we define the $n_d \times n_d$ matrices $N^{K \to L}$ and $N^{L \to K}$, expressing the amount of nearest neighbours of each coarse cell $\bar{x}^K_m$ and $\bar{x}^L_m$ assigned to cells of the respective other cloud $\bar{x}^{K \to L}_m, \bar{x}^{L \to K}_m$. The values are subsequently normalized by the total number of nearest neighbours per cell, filling the entries of W_c by selecting the minimum between the two matrices. Furthermore, dustbin dimensions are filled with 1 minus the number of matches per cell proportionally to the total number of possible matches. In this context the coarse matching loss $\mathcal{L}_c$ can be defined as:

$$\mathcal{L}_c = -\frac{1}{n_d} \sum_{k=1,l=1}^{n_d+1,n_d+1} W_c(k,l) \cdot \log(S_c(k,l)) \tag{7}$$

$$\begin{aligned} W_c(k,l) &= min\left(\frac{N^{K \to L}(k,l)}{\sum_l N^{K \to L}(k,l)}, \frac{N^{L \to K}(k,l)}{\sum_k N^{L \to K}(k,l)}\right), \\ W_c(k, n_d + 1) &= 1 - \frac{\sum_l N^{K \to L}(k,l)}{2^{2n_l}}, \\ W_c(n_d + 1, l) &= 1 - \frac{\sum_k N^{L \to K}(k,l)}{2^{2n_l}}, k, l \in \{1, \ldots, n_d\} \end{aligned} \tag{8}$$

4. Experiments

We evaluate the performance of MagneticPillars against recent state-of-the-art point cloud registration methods concerning estimation accuracy and runtime on the two autonomous driving datasets KITTI [13], and Nuscenes [5]. More specifically, we chose the feature-based methods 3DFeat-Net [28], FCGF [9], SpinNet [2], D3Feat [3], Predator [14], and the correspondence prediction approaches CoFiNet [29], RegTr [16] and GeoTransformer [20] based on the publicly available implementations. For a fair comparison, the following experiments were conducted on a reference system featuring a *NVIDIA Geforce RTX 3090* graphics card.

If not stated otherwise, the baseline parametrization for our approach in the following investigations is $n_c = 30$, $n_l = 4$, with $n_k = 250$ extracted key points, trained with combined coarse and fine loss applying weighted SVD as registration solver.

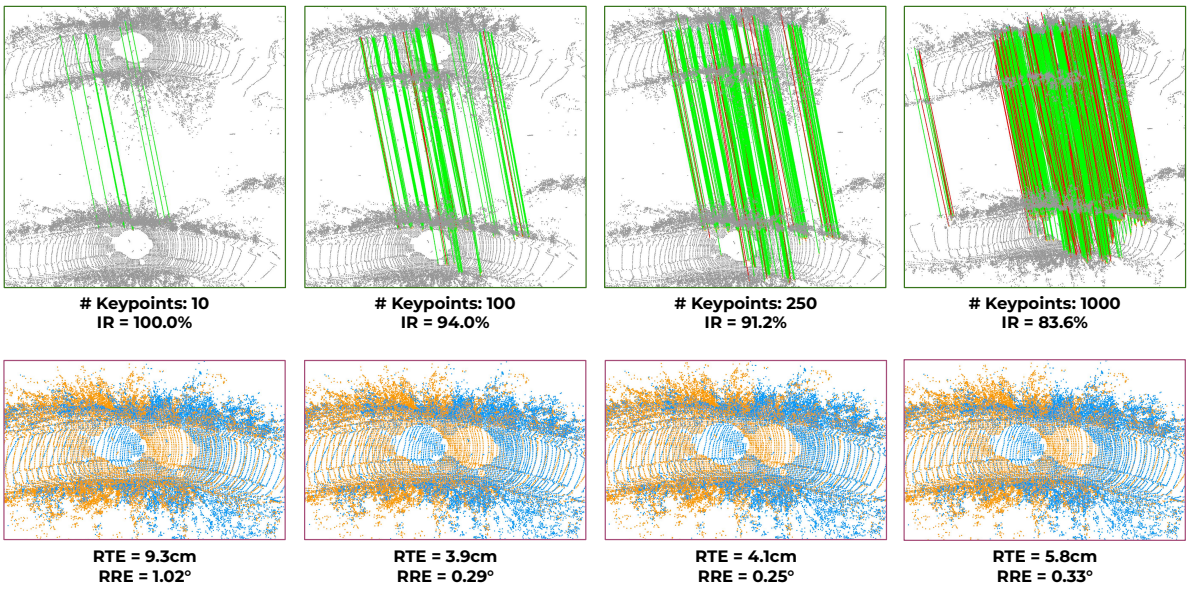

Figure 3. Qualitative results demonstrate the robustness of the predicted key points and their correspondences by MagneticPillars for different numbers of extracted points. Green and red lines at the top represent correct and false correspondences, while the bottom row shows the application of the resulting pose, determined by applying SVD on the predicted pillar point correspondences. As a result, selecting the 10 highest-ranked predictions is already sufficient to establish an accurate pose estimation, proving our robust feature generation and matching.

4.1. Data Pre-Processing and Metrics

For dataset generation, we follow the same preprocessing as stated in the related works [3, 9, 14, 20, 29] filtering point cloud pairs in a radius of at least $10m$ throughout the dataset. Regarding the KITTI Odometry dataset, we again followed the previous works and chose sequences $00 - 05$ for training, $06, 07$ for validation, and $08 - 10$ for testing and utilized the ICP refinement on the ground truth poses T_{GT}. For Nuscenes we followed the same data processing, selecting frames with their respective counterpart in a $10m$ radius. Here we chose the official training, validation and test split to generate the subsets and again applied ICP to counteract possible errors in the ground truth poses.

Key metrics used in our experiments are *Registration Recall (RR)*, *Relative Translational Error (RTE)* and *Relative Rotational Error (RRE)* as well as the general runtime *T* of the respective methods. Here RRE describes the geodesic distance between the ground truth and estimated rotation matrix, RTE the euclidian distance of the translation vectors, and RR the percentage of transformations where RTE and RRE are below the thresholds of: RTE $< 0.6m$ and RRE $< 5°$. Furthermore, we also report the *Inlier Ratio (IR)*, which expresses the fraction of valid point correspondences.

4.2. Evaluation on the KITTI Dataset

4.2.1 Registration Validation

We start off our validation on the KITTI dataset by comparing the transformation accuracy of the considered methods with respect to different numbers of extracted key points and registration backends, listed in Table 1. Here we follow the parametrization of previous works [14, 20, 29] for selecting the number of points and registration solvers. With regard to the RANSAC registration performance choosing 5000 extracted feature points, our MagneticPillars approach is able to reach the state-of-the-art value of 99.5% RR and second lowest RTE of $6.4cm$, whereby Predator ranked best with $6.0cm$. Furthermore, we feature the second lowest inference time with 0.112 seconds per frame, solely being surpassed by FCGF, which, however shows worse registration accuracy. Although sharing the same parametrization and environment throughout all methods, RANSAC registration time highly fluctuates among the approaches, presumably originating from different periods until achieving convergence.

Taking a look at the results featuring only 250 extracted key points applying weighted SVD as registration backend shows the robustness of our extracted feature points and matching technique. Here we reach the highest values in not only terms of RR (99.5%), RRE ($0.387°$) and RTE ($6.8cm$),

but also second lowest model inference time with a much lower registration time since no iterative optimization is performed in this context. Overall, using SVD as backend, correspondence prediction methods show a better registration accuracy compared to the feature matching methods. In this context, SpinNet and 3DFeatNet generally failed to predict valid point correspondences and poses, with Predator still performing best out of the feature generation methods with a RR of 10.8%.

Applying Local-Global-Registration (LGR), introduced in [20], which basically represents an iterative application of weighted SVD, we are able to produce the best results in terms of RR (99.5%), RTE ($5.4cm$) and runtimes ($0.112s + 0.005s$). Generally, LGR is able to counteract possible false predictions, distorting the single-shot SVD pose estimations.

Table 1. Registration performance of the considered methods on the KITTI test dataset with respect to different pose estimation backends and number of key points n_k.

Method	n_k	Estimator	RR ↑ (%)	RTE ↓ (cm)	RRE ↓ (°)	Time ↓ (s) Model	Reg.
3DFeat-Net [28]	5000	RANSAC	18.0	29.1	1.479	34.93	0.156
FCGF [9]	5000	RANSAC	93.0	15.0	0.389	**0.054**	0.078
SpinNet [2]	5000	RANSAC	97.3	9.0	0.490	56.26	0.056
D3Feat [3]	5000	RANSAC	**99.5**	6.9	0.313	0.150	0.114
Predator [14]	5000	RANSAC	**99.5**	**6.0**	**0.275**	0.173	0.100
CoFiNet [29]	5000	RANSAC	**99.5**	7.8	0.360	0.375	**0.027**
RegTr [16]	5000	RANSAC	86.5	22.8	0.419	0.460	0.043
GeoTransformer [20]	5000	RANSAC	**99.5**	7.3	0.289	0.245	0.042
MagneticPillars (Ours)	5000	RANSAC	**99.5**	6.4	0.299	0.112	0.082
3DFeat-Net [28]	250	w. SVD	0	-	-	34.93	**0.001**
FCGF [9]	250	w. SVD	4.3	34.4	2.65	**0.086**	0.003
SpinNet [2]	250	w. SVD	0	-	-	56.26	0.003
D3Feat [3]	250	w. SVD	3.1	37.4	2.244	0.206	0.008
Predator [14]	250	w. SVD	10.8	38.9	1.754	0.173	0.002
CoFiNet [29]	250	w. SVD	33.5	34.1	1.072	0.375	**0.001**
RegTr [16]	250	w. SVD	85.6	22.7	0.472	0.460	**0.001**
GeoTransformer [20]	250	w. SVD	98.9	11.2	0.438	0.245	0.004
MagneticPillars (Ours)	250	w. SVD	**99.5**	**6.8**	**0.387**	0.112	**0.001**
Predator [14]	5000	LGR	85.4	6.6	0.482	0.173	0.298
CoFiNet [29]	5000	LGR	83.4	10.3	0.514	0.375	0.006
RegTr [16]	5000	LGR	86.3	22.3	0.420	0.460	**0.005**
GeoTransformer [20]	5000	LGR	**99.5**	6.0	**0.233**	0.245	0.012
MagneticPillars (Ours)	5000	LGR	**99.5**	**5.4**	0.297	**0.112**	**0.005**

To demonstrate the robustness of MagneticPillars in a more extended context, we list the registration performance applying SVD with a varying number of key points reaching from 5000 all the way to merely 10 extracted matches in Table 2. Here we rank highest in every category for almost all entries, even reaching an average RR of over 97% for only 10 extracted key points. Furthermore, we are able to contain a transformation accuracy $<$14cm and $<1.0°$ throughout all key point selections. A qualitative visualization of our registration performance under a varying number of key points is moreover displayed in Figure 3.

Overall MagneticPillars is able to outperform state-of-the-art methods or gain comparable results in all considered categories on the KITTI Odometry dataset. Mostly comparable to our results is the work of Geotransformer [20] which however features more than twice the computation time, originating from an expensive coarse to fine point aggregation consuming half of the runtime $120ms/245ms$. This can be compensated by our efficient 2D-BEV processing featuring fixed coarse to fine grid cell correspondences.

Table 2. Registration performance of the considered methods on the KITTI test dataset with varying number of extracted key points using SVD for pose estimation.

# Key points n_k	10	25	50	100	250	500	1000	2500	5000
	Registration Recall ↑ (%)								
FCGF [9]	0	0.2	0.4	0.7	4.3	6.5	9.7	10.6	12.3
D3Feat [3]	0.5	0.2	0.5	2.0	3.1	3.6	4.9	10.1	16.0
Predator [14]	0	0.9	1.6	3.9	10.8	17.7	23.4	33.2	38.4
CoFiNet [29]	40.9	39.1	30.6	29.5	33.5	36.9	40.3	42.7	35.7
RegTr [16]	68.8	79.3	82.7	85.1	85.6	86.3	86.0	85.8	85.6
GeoTransformer [20]	90.8	96.6	98.4	98.9	98.9	99.1	**99.1**	**98.9**	98.6
MagneticPillars (Ours)	**97.1**	**99.3**	**99.5**	**99.5**	**99.5**	**99.5**	**99.1**	**98.9**	**98.7**
	Relative Translational Error ↓ (cm)								
FCGF [9]	-	33.5	50.3	39.2	34.4	31.9	34.1	31.7	32.7
D3Feat [3]	36.7	46.0	45.6	49.1	37.4	37.3	38.1	36.8	37.6
Predator [14]	-	45.5	38.5	39.5	38.9	33.2	34.5	32.3	33.5
CoFiNet [29]	30.6	29.7	31.9	32.6	34.1	34.4	34.6	35.8	35.6
RegTr [16]	33.3	28.0	25.6	24.1	22.7	22.4	22.3	22.5	22.5
GeoTransformer [20]	23.0	17.6	14.3	12.6	11.2	10.9	10.5	10.8	**11.9**
MagneticPillars (Ours)	**13.5**	**10.2**	**8.6**	**7.5**	**6.8**	**7.1**	**8.3**	**10.5**	13.0
	Relative Rotational Error ↓ (°)								
FCGF [9]	-	3.173	3.136	2.871	2.65	2.368	2.468	2.274	2.151
D3Feat [3]	3.138	2.865	2.661	2.614	2.244	1.939	1.67	1.35	1.167
Predator [14]	-	3.076	2.914	2.498	1.754	1.313	1.003	0.789	0.759
CoFiNet [29]	1.591	1.568	1.404	1.261	1.072	0.917	0.811	0.713	0.673
RegTr [16]	1.169	0.815	0.622	0.583	0.472	0.466	0.432	0.452	0.464
GeoTransformer [20]	1.071	0.765	0.642	0.526	0.438	0.391	**0.372**	**0.368**	**0.383**
MagneticPillars (Ours)	**0.907**	**0.661**	**0.527**	**0.441**	**0.387**	**0.383**	0.398	0.442	0.480

4.2.2 Generalization

Applying the pre-processing according to Section 4.1 generates around 1350 frame pairs with a distance of $10m$ for the training dataset. In order to show the generalizability of the considered methods to prevent a bias to $10m$ distances, we further conducted experiments on $15m$ and $20m$ frame gaps. Note that training still was performed exclusively on the $10m$ splits along the lines of the previous experiments, but applied on test sets with $15m$ and $20m$ as presented in Table 3.

While the $15m$ splits generally can be handled by most of the approaches applying RANSAC for outlier reduction, performance of the compared methods drastically reduces on the 20m gaps, while MagneticPillars is still able to generate a RR of almost 96%. Utilizing SVD as registration backend, solely GeoTransformer is able to produce acceptable results, but surpassed by our method by more than

Table 3. Registration performance on the KITTI test dataset for larger frame distances of $15m$ and $20m$ based on the $10m$ trainings.

Method	n_k	Estimator	15m			20m		
			RR ↑	RTE ↓	RRE ↓	RR ↑	RTE ↓	RRE ↓
Predator [14]	5000	RANSAC	93.8	12.1	0.648	25.6	22.9	1.506
CoFiNet [29]	5000	RANSAC	97.5	11.8	0.575	82.2	21.2	1.140
RegTr [16]	5000	RANSAC	0	-	-	0	-	-
GeoTransformer [20]	5000	RANSAC	97.9	9.9	**0.405**	50.5	16.1	1.295
MagneticPillars (Ours)	5000	RANSAC	**98.1**	**8.7**	**0.405**	**95.7**	**11.6**	**0.535**
Predator [14]	250	w. SVD	0	-	-	0	-	-
CoFiNet [29]	250	w. SVD	3.5	35.7	1.515	0	-	-
RegTr [16]	250	w. SVD	0	-	-	0	-	-
GeoTransformer [20]	250	w. SVD	84.2	21.4	0.583	15.0	37.7	1.375
MagneticPillars (Ours)	250	w. SVD	**95.7**	**10.9**	**0.523**	**74.0**	**18.1**	**0.816**

$+11\%$ RR on the $15m$ data and almost $+60\%$ for the $20m$ set. Overall our approach is showing high generalizability and robustness to frames pairs with distances not featured within the training process.

4.2.3 Ablation Study

In order to show the benefits of our introduced smooth coarse matching loss $\mathcal{L}_c$ in Section 3.6.2, we compare the performance of our network at the end of the training process, by exchanging it with a hard binary loss $\mathcal{L}_b$, similar to $\mathcal{L}_f$. In this context the pillar centroid for each coarse cell is determined by averaging the coordinates of all points aggregated by the corresponding fine cell associations. Finally, the desired weight matrix can be calculated according to Equation 6 on the established coarse centroids. As listed in Table 4 we found that utilizing $\mathcal{L}_b$ instead of $\mathcal{L}_c$ in combination with $\mathcal{L}_f$ still leads to convergence of the network but results in a slightly decreased prediction accuracy of 99.1%. However solely using one of the respective loss functions will lead to an erroneous training process which demonstrates the importance of a combined robust cell prefiltering with subsequent correspondence refinement.

Furthermore we performed additional experiments regarding the initial feature extraction by replacing it with a single MLP directly encoding the global point coordinates $p_{i,j}$ (*Merged Enc.*) per voxel, instead of separated local and global cloud information. This again leads to a decrease in estimation accuracy without visible improvements in runtime, supporting our claim for a split local and global feature encoding.

4.3. Evaluation on the Nuscenes Dataset

Finally, we show our registration performance on the Nuscenes dataset, validating against the best-performing methods of our previous investigations. Training is conducted according to the splits described in Section 4.1 for each method for 100 epochs. The results on the test dataset

Table 4. Ablation study on the impact of the proposed loss functions and input feature encoding with $n_k = 250$ applying SVD.

$\mathcal{L}_b$	$\mathcal{L}_c$	$\mathcal{L}_f$	Split Enc.	Merged Enc.	RR ↑	RTE ↓	RRE ↓	T ↓
x			x		0	-	-	0.113
	x		x		0	-	-	0.113
		x	x		0.72	36.4	2.742	0.113
x		x	x		99.1	**6.8**	**0.366**	0.113
	x	x	x		**99.5**	**6.8**	0.387	0.113
	x	x		x	99.1	7.2	0.413	0.113

are listed in Table 5. Here MagneticPillars is able to rank best in terms of registration recall and runtime throughout all categories and generating lowest translational and rotational error applying SVD as registration backend and second lowest for LGR, proving its adaptability to different LiDAR sensors and environments.

Table 5. Registration performance on the Nuscenes test dataset.

Method	n_k	Estimator	RR ↑ (%)	RTE ↓ (cm)	RRE ↓ (°)	Time ↓ (s) Model	Reg.
Predator [14]	5000	RANSAC	**99.9**	**7.2**	0.265	0.128	0.100
CoFiNet [29]	5000	RANSAC	97.0	10.8	0.416	0.288	**0.026**
RegTr [16]	5000	RANSAC	99.1	12.6	**0.250**	0.155	0.045
GeoTransformer [20]	5000	RANSAC	**99.9**	10.0	0.312	0.207	0.029
MagneticPillars (Ours)	5000	RANSAC	**99.9**	10.2	0.356	**0.114**	**0.026**
Predator [14]	250	w. SVD	33.2	37.1	2.184	0.128	0.002
CoFiNet [29]	250	w. SVD	35.0	32.8	1.092	0.288	0.002
RegTr [16]	250	w. SVD	98.8	12.8	0.467	0.155	0.002
GeoTransformer [20]	250	w. SVD	98.8	18.7	0.456	0.207	0.003
MagneticPillars (Ours)	250	w. SVD	**99.2**	**12.1**	**0.386**	**0.114**	**0.001**
Predator [14]	5000	LGR	96.2	**6.8**	0.500	0.128	0.292
CoFiNet [29]	5000	LGR	75.6	11.7	0.602	0.288	0.006
RegTr [16]	5000	LGR	99.0	12.6	0.458	0.155	0.018
GeoTransformer [20]	5000	LGR	99.3	8.7	**0.268**	0.207	0.010
MagneticPillars (Ours)	5000	LGR	**99.8**	8.2	0.283	**0.114**	**0.005**

5. Conclusion

In this work, we present MagneticPillars an efficient and accurate point cloud registration network on BEV representations of LiDAR point clouds. Due to fixed grid size, we are able to directly establish associations between prefiltered coarse cell correspondences and the resulting fine feature vectors. This not only leads to an improved invariance to the constitution of the input point cloud but also yields a massive boost in efficiency, increasing it's utilizability for real-time applications. We extensively validate our approach on the KITTI and Nuscenes datasets mostly outperforming recent point cloud registration methods regarding pose estimation accuracy and overall runtime.

References

[1] Sk Aziz Ali, Kerem Kahraman, Gerd Reis, and Didier Stricker. Rpsrnet: End-to-end trainable rigid point set registration network using barnes-hut 2 d-tree representation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13100–13110, 2021. 2

[2] Sheng Ao, Qingyong Hu, Boy Ang, Andrew Markham, and Yulan Guo. Spinnet: Learning a general surface descriptor for 3d point cloud registration. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11753–11762, 2021. 2, 5, 7

[3] Xuyang Bai, Zixin Luo, Lei Zhou, Hongbo Fu, Long Quan, and Chiew-Lan Tai. D3feat: Joint learning of dense detection and description of 3d local features. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6359–6367, 2020. 2, 3, 5, 6, 7

[4] Paul J Besl and Neil D McKay. A method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, pages 586–606, 1992. 2

[5] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11621–11631, 2020. 5

[6] Anh-Quan Cao, Gilles Puy, Alexandre Boulch, and Renaud Marlet. Pcam: Product of cross-attention matrices for rigid registration of point clouds. In *IEEE International Conference on Computer Vision (ICCV)*, pages 13229–13238, 2021. 2

[7] Zhi Chen, Kun Sun, Fan Yang, and Wenbing Tao. Sc 2-pcr: A second order spatial compatibility for efficient and robust point cloud registration. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13221–13231, 2022. 2

[8] Christopher Choy, Wei Dong, and Vladlen Koltun. Deep global registration. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2514–2523, 2020. 2

[9] Christopher Choy, Jaesik Park, and Postech Vladlen Koltun. Fully convolutional geometric features. In *IEEE International Conference on Computer Vision (ICCV)*, pages 8958–8966, 2019. 2, 5, 6, 7

[10] Kai Fischer, Martin Simon, Stefan Milz, and Patrick Mäder. Stickylocalization: Robust end-to-end relocalization on point clouds using graph neural networks. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2962–2971, 2022. 3

[11] Kai Fischer, Martin Simon, Florian Ölsner, Stefan Milz, Horst-Michael Groß, and Patrick Mäder. Stickypillars: Robust and efficient feature matching on point clouds using graph neural networks. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 313–323, 2021. 3

[12] Kexue Fu, Shaolei Liu, Xiaoyuan Luo, and Manning Wang. Robust point cloud registration framework based on deep graph matching. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8893–8902, 2021. 3

[13] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 2012. 5

[14] Shengyu Huang, Zan Gojcic, Mikhail Usvyatsov, Andreas Wieser, Konrad Schindler, and Eth Zurich. Predator: Registration of 3d point clouds with low overlap. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4267–4276, 2021. 2, 5, 6, 7, 8

[15] Xiaoshui Huang, Guofeng Mei, and Jian Zhang. Feature-metric registration: A fast semi-supervised approach for robust point cloud registration without correspondences. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11366–11374, 2020. 2

[16] Zi Jian, Yew Gim, and Hee Lee. Regtr: End-to-end point cloud correspondences with transformers. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6677–6686, 2022. 3, 5, 7, 8

[17] Yang Li and Tatsuya Harada. Lepard: Learning partial point cloud matching in rigid and deformable scenes. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5554–5564, 2022. 3

[18] Fan Lu, Guang Chen, Yinlong Liu, Lijun Zhang, Sanqing Qu, Shu Liu, and Rongqi Gu. Hregnet: A hierarchical network for large-scale outdoor lidar point cloud registration. In *IEEE International Conference on Computer Vision (ICCV)*, pages 16014–16023, 2021. 2

[19] Weixin Lu, Guowei Wan, Yao Zhou, Xiangyu Fu, Pengfei Yuan, and Shiyu Song. Deepvcp: An end-to-end deep neural network for point cloud registration. In *IEEE International Conference on Computer Vision (ICCV)*, pages 12–21, 2019. 2

[20] Zheng Qin, Hao Yu, Changjian Wang, Yulan Guo, Yuxing Peng, and Kai Xu. Geometric transformer for fast and robust point cloud registration. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11143–11152, 2022. 2, 3, 5, 6, 7, 8

[21] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351, pages 234–241. Springer Verlag, 2015. 2

[22] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4938–4947, 2020. 4

[23] Yaqi Shen, Le Hui, Haobo Jiang, Jin Xie, and Jian Yang. Reliable inlier evaluation for unsupervised point cloud registration. In *Conference on Artificial Intelligence (AAAI)*, pages 2198–2206, 2022. 3

[24] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for

point clouds. In *IEEE International Conference on Computer Vision (ICCV)*, pages 6411–6420, 2019. 1, 2

[25] 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 (NeurIPS)*, pages 5998–6008, 2017. 4

[26] Bingli Wu, Jie Ma, Gaojie Chen, and Pei An. Feature interactive representation for point cloud registration. In *IEEE International Conference on Computer Vision (ICCV)*, pages 5530–5539, 2021. 3

[27] Hao Xu, Nianjin Ye, Guanghui Liu, Bing Zeng, and Shuaicheng Liu. Finet: Dual branches feature interaction for partial-to-partial point cloud registration. In *Conference on Artificial Intelligence (AAAI)*, pages 2848–2856, 2022. 2

[28] Zi Jian Yew and Gim Hee Lee. 3dfeat-net: Weakly supervised local 3d features for point cloud registration. In *European Conference on Computer Vision (ECCV)*, pages 607–623, 2018. 2, 5, 7

[29] Hao Yu, Fu Li, Mahdi Saleh, Benjamin Busam, and Slobodan Ilic. Cofinet: Reliable coarse-to-fine correspondences for robust point cloud registration. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 23872–23884, 2021. 2, 3, 5, 6, 7, 8

[30] Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, Boqing Gong, and Hassan Foroosh. Polarnet: An improved grid representation for online lidar point clouds semantic segmentation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9601–9610, 2020. 4