配准：全局坐标系下不同视角的定位和定向

PREDATOR: Registration of 3D Point Clouds with Low Overlap

Shengyu Huang* Zan Gojcic* Mikhail Usvyatsov Andreas Wieser Konrad Schindler

ETH Zurich

overlappredator.github.io

宽松评价

Abstract

We introduce PREDATOR, *a model for pairwise* ***p****oint-cloud* ***re****gistration with* ***d****eep* ***at****tention to the* ***o****verlap* ***r****egion. Different from previous work, our model is specifically designed to handle (also) point-cloud pairs with low overlap. Its key novelty is an overlap-attention block for early information exchange between the latent encodings of the two point clouds. In this way the subsequent decoding of the latent representations into per-point features is conditioned on the respective other point cloud, and thus can predict which points are not only salient, but also lie in the overlap region between the two point clouds. The ability to focus on points that are relevant for matching greatly improves performance:* PREDATOR *raises the rate of successful registrations by more than 15 percent points in the low-overlap scenario, and also sets a new state of the art for the* 3DMatch *benchmark with 90.6% registration recall. [Code release]*

1. Introduction

Recent work has made substantial progress in fully automatic, 3D feature-based point cloud registration. At first glance, benchmarks like *3DMatch* [56] appear to be saturated, with multiple state-of-the-art (SoTA) methods [18, 9, 3] reaching nearly 95% feature matching recall and successfully registering >80% of all scan pairs. One may get the impression that the registration problem is solved—but this is actually not the case. We argue that the high success rates are a consequence of lenient evaluation protocols. We have been making our task too easy: existing literature and benchmarks [6, 56, 23] consider only pairs of point clouds with ≥30% overlap to measure performance. Yet, the low-overlap regime is very relevant for practical applications. On the one hand, it may be difficult to ensure high overlap, for instance when moving along narrow corridors, or when closing loops in the presence of occlusions (densely built-up areas, forest, etc.). On the other hand, data acquisition is often costly, so practitioners aim for a low number of scans with only the necessary overlap [52, 53].

*First two authors contributed equally to this work.

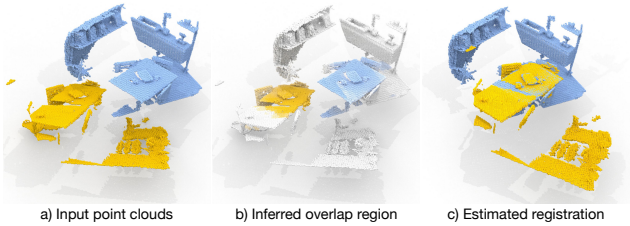

Figure 1: PREDATOR is designed to focus attention on the overlap region, and to prefer salient points in that region, so as to enable robust registration in spite of low overlap.

Driven by the evaluation protocol, the high-overlap scenario became the focus of research, whereas the more challenging low-overlap examples were largely neglected (*cf*. Fig. 1). Consequently, the registration performance of even the best known methods deteriorates rapidly when the overlap between the two point clouds falls below 30%, see Fig. 2. Human operators, in contrast, can still register such low overlap point clouds without much effort.

低重叠
SOTA不好

This discrepancy is the starting point of the present work. To study its reasons, we have constructed a low-overlap dataset *3DLoMatch* from scans of the popular *3DMatch* benchmark, and have analysed the individual modules/steps of the registration pipeline (Fig. 2). It turns out that the effective receptive field of fully convolutional feature point descriptors [9, 3] is local enough and the descriptors are hardly corrupted by non-overlapping parts of the scans. Rather than coming up with yet another way to learn better descriptors, the key to registering low overlap point clouds is *learning where to sample feature points*. A large performance boost can be achieved if the feature points are predominantly sampled from the overlapping portions of the scans (Fig. 2, right).

确定重叠区很重要

We follow this path and introduce PREDATOR, a neural architecture for pairwise 3D point cloud registration that learns to detect the overlap region between two unregistered scans, and to focus on that region when sampling feature

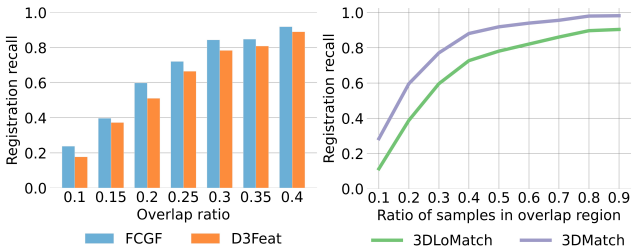

Figure 2: Registration with SoTA methods deteriorates rapidly for pairs with <30% overlap (*left*). By increasing the fraction of points sampled in the overlap region, many failures can be avoided as shown here for FCGF [9] (*right*).

points. The main contributions of our work are:

- an analysis why existing registration pipelines break down in the low-overlap regime
- a novel *overlap attention* block that allows for early information exchange between the two point clouds and focuses the subsequent steps on the overlap region
- a scheme to refine the feature point descriptors, by conditioning them also on the respective other point cloud
- a novel loss function to train *matchability* scores, which help to sample better and more repeatable interest points

Moreover, we make available the *3DLoMatch* dataset, containing the previously ignored scan pairs of *3DMatch* that have low (10-30%) overlap. In our experiments, PREDATOR greatly outperforms existing methods in the low-overlap regime, increasing registration recall by >15 percent points. It also sets a new state of the art on the *3DMatch* benchmark, reaching a registration recall of >90%.

2. Related work

We start this related-work section by reviewing the individual components of the traditional point cloud registration pipelines, before proceeding to newer, end-to-end point-cloud registration algorithms. Finally, we briefly cover recent advances in using contextual information to guide and robustify feature extraction and matching.

Local 3D feature descriptors: Early local descriptors for point clouds [22, 35, 34, 42, 41] aimed to characterise the local geometry by using hand-crafted features. While often lacking robustness against clutter and occlusions, they have long been a default choice for downstream tasks because they naturally generalise across datasets [20]. In the last years, learned 3D feature descriptors have taken over and now routinely outperform their hand-crafted counterparts.

The pioneering 3DMatch method [56] is based on a Siamese 3D CNN that extracts local feature descriptors from a signed distance function embedding. Others [23, 19] first extract hand-crafted features, then map them to a compact representation using multi-layer perceptrons. PPFNet [12], and its self-supervised version PPF-FoldNet [11], combine point pair features with a PointNet [32] architecture to extract descriptors that are aware of the global context. To alleviate artefacts caused by noise and voxelisation, [18] proposed to use a smoothed density voxel grid as input to a 3D CNN. These early works achieved strong performance, but still operate on individual local patches, which greatly increases the computational cost and limits the receptive field to a predefined size.

Fully convolutional architectures [26] that enable dense feature computation over the whole input in a single forward pass [13, 14, 33] have been adopted to design faster 3D feature descriptors. Building on sparse convolutions [8], FCGF [9] achieves a performance similar to the best patch-based descriptors [18], while being orders of magnitude faster. D3Feat [3] complements a fully convolutional feature descriptor with an salient point detector.

Interest point sampling: The classic principle to sample salient rather than random points has also found its way into learned 2D [13, 14, 33, 49] and 3D [54, 3, 27] local feature extraction. All these methods implicitly assume that the saliency of a point fully determines its utility for downstream tasks. Here, we take a step back and argue that, while saliency is desirable for an interest point, it is not sufficient on its own. Indeed, in order to contribute to registration a point should not only be salient, but must also lie in the region where the two point clouds overlap—an essential property that, surprisingly, has largely been neglected thus far.

Deep point-cloud registration: Instead of combining learned feature descriptors with some off-the-shelf robust optimization at inference time, a parallel stream of work aims to embed the differentiable pose estimation into the learning pipeline. PointNetLK [1] combines a PointNet-based global feature descriptor [32] with a Lucas/Kanade-like optimization algorithm [28] and estimates the relative transformation in an iterative fashion. DCP [46] use a DGCNN network [48] to extract local features and computes soft correspondences before using the Kabsch algorithm to estimate the transformation parameters. To relax the need for strict one-to-one correspondence, DCP was later extended to PRNet [47], which includes a *keypoint* detection step and allows for partial correspondence. Instead of simply using soft correspondences, [55] update the similarity matrix with a differentiable Sinkhorn layer [38]. Similar to other methods, the weighted Kabsch algorithm[2] is used to estimate the transformation parameters. Finally, [17, 7, 31] complement a learned feature descriptor with an outlier filtering network, which infers the correspondence weights for later use in the weighted Kabsch algorithm.

Contextual information: In the traditional pipeline, feature extraction is done independently per point cloud. Information is only communicated when computing pairwise similarities, although aggregating contextual information at an earlier stage could provide additional cues to robustify

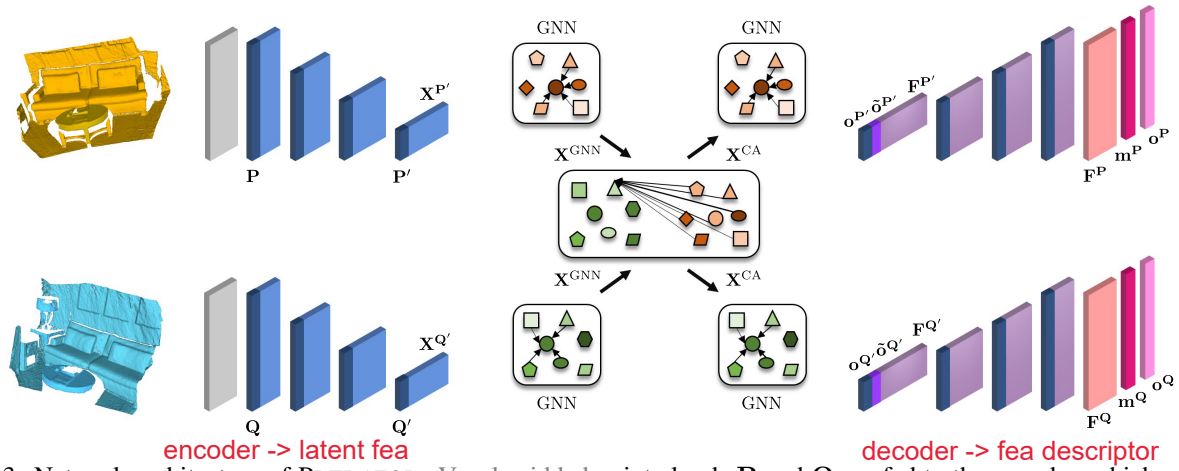

Figure 3: Network architecture of PREDATOR. Voxel-gridded point clouds $\mathbf{P}$ and $\mathbf{Q}$ are fed to the encoder, which extracts the superpoints $\mathbf{P}'$ and $\mathbf{Q}'$ and their latent features $\mathbf{X}^{\mathbf{P}'}$, $\mathbf{X}^{\mathbf{Q}'}$. The overlap-attention module updates the features with co-contextual information in a series of self- (GNN) and cross-attention (CA) blocks, and projects them to overlap $\mathbf{o}^{\mathbf{P}'}$, $\mathbf{o}^{\mathbf{Q}'}$ and cross-overlap $\tilde{\mathbf{o}}^{\mathbf{P}'}$, $\tilde{\mathbf{o}}^{\mathbf{Q}'}$ scores. Finally, the decoder transforms the conditioned features and overlap scores to per-point feature descriptors $\mathbf{F}^{\mathbf{P}}$, $\mathbf{F}^{\mathbf{Q}}$, overlap scores $\mathbf{o}^{\mathbf{P}}$, $\mathbf{o}^{\mathbf{Q}}$, and matchability scores $\mathbf{m}^{\mathbf{P}}$, $\mathbf{m}^{\mathbf{Q}}$.

the descriptors and guide the matching step.

In 2D feature learning, D2D-Net [49] use an attention mechanism in the bottleneck of an encoder-decoder scheme to aggregate the contextual information, which is later used to condition the output of the decoder on the second image. SuperGlue [36] infuses the contextual information into the learned descriptors with a whole series of self- and cross-attention layers, built upon the message-passing GNN [24]. Early information mixing was previously also explored in the field of deep point cloud registration, where [46, 47] use a transformer module to extract task-specific 3D features that are reinforced with contextual information.

3. Method

PREDATOR is a two-stream encoder-decoder network. Our default implementation uses residual blocks with KPConv-style point convolutions [40], but the architecture is agnostic w.r.t. the backbone and can also be implemented with other formulations of 3D convolutions, such as for instance sparse voxel convolutions [8] (*cf*. Appendix). As illustrated in Fig. 3, the architecture of PREDATOR can be decomposed into three main modules:

1. encoding of the two point clouds into smaller sets of superpoints and associated latent feature encodings, with shared weights (Sec. 3.2);
2. the overlap attention module (in the bottleneck) that extracts co-contextual information between the feature encodings of the two point clouds, and assigns each superpoint two overlap scores that quantify how likely the superpoint itself and its soft-correspondence are located in the overlap between the two inputs (Sec. 3.3);
3. decoding of the mutually conditioned bottleneck representations to point-wise descriptors as well as refined per-point overlap and matchability scores (Sec. 3.4).

Before diving into each component we lay out the basic problem setting and notation in Sec. 3.1.

3.1. Problem setting

Consider two point clouds $\mathbf{P} = \{\mathbf{p}_i \in \mathbb{R}^3 | i = 1..N\}$, and $\mathbf{Q} = \{\mathbf{q}_i \in \mathbb{R}^3 | i = 1..M\}$. Our goal is to recover a rigid transformation $\mathbf{T}^{\mathbf{Q}}_{\mathbf{P}}$ with parameters $\mathbf{R} \in SO(3)$ and $\mathbf{t} \in \mathbb{R}^3$ that aligns $\mathbf{P}$ to $\mathbf{Q}$. By a slight abuse of notation we use the same symbols for sets of points and for their corresponding matrices $\mathbf{P} \in \mathbb{R}^{N \times 3}$ and $\mathbf{Q} \in \mathbb{R}^{M \times 3}$.

Obviously $\mathbf{T}^{\mathbf{Q}}_{\mathbf{P}}$ can only ever be determined from the data if $\mathbf{P}$ and $\mathbf{Q}$ have sufficient overlap, meaning that after applying the ground truth transformation $\overline{\mathbf{T}}^{\mathbf{Q}}_{\mathbf{P}}$ the overlap ratio

$$\frac{1}{N}\left|\left\{\|(\overline{\mathbf{T}}^{\mathbf{Q}}_{\mathbf{P}}(\mathbf{p}_i) - \mathsf{NN}(\overline{\mathbf{T}}^{\mathbf{Q}}_{\mathbf{P}}(\mathbf{p}_i), \mathbf{Q})\|_2 \leq v\right\}\right| > \tau\,, \quad (1)$$

where NN denotes the nearest-neighbour operator w.r.t. its second argument, $\|\cdot\|_2$ is the Euclidean norm, $|\cdot|$ is the set cardinality, and v is a tolerance that depends on the point density.[2] Contrary to previous work [56, 23], where the threshold to even attempt the alignment is typically $\tau > 0.3$, we are interested in low-overlap point clouds with $\tau > 0.1$. Fragments with different overlap ratios are shown in Fig. 4.

3.2. Encoder

We follow [40] and first down-sample raw point clouds with a voxel-grid filter of size V, such that $\mathbf{P}$ and $\mathbf{Q}$ have reasonably uniform point density. In the shared encoder,

[2] For efficiency, v is in practice determined after voxel-grid down-sampling of the two point clouds.

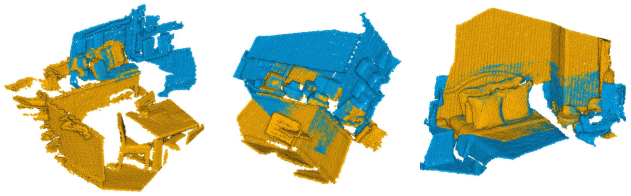

Figure 4: Fragments with different overlap ratios. Overlap is computed relative to the source fragment (orange).

a series of ResNet-like blocks and strided convolutions aggregate the raw points into *superpoints* $\mathbf{P}' \in \mathbb{R}^{N' \times 3}$ and $\mathbf{Q}' \in \mathbb{R}^{M' \times 3}$ with associated features $\mathbf{X}^{\mathbf{P}'} \in \mathbb{R}^{N' \times b}$ and $\mathbf{X}^{\mathbf{Q}'} \in \mathbb{R}^{M' \times b}$. Note that superpoints correspond to a fixed receptive field, so their number depends on the spatial extent of the input point cloud and may be different for the two inputs.

3.3. Overlap attention module

So far, the features $\mathbf{X}^{\mathbf{P}'}$, $\mathbf{X}^{\mathbf{Q}'}$ in the bottleneck encode the geometry and context of the two point clouds. But $\mathbf{X}^{\mathbf{P}'}$ has no knowledge of point cloud $\mathbf{Q}$ and vice versa. In order to reason about their respective overlap regions, some cross-talk is necessary. We argue that it makes sense to add that cross-talk at the level of superpoints in the bottleneck, just like a human operator will first get a rough overview of the overall shape to determine likely overlap regions, and only after that identifies precise feature points in those regions.

Graph convolutional neural network: Before connecting the two feature encodings, we first further aggregate and strengthen their contextual relations individually with a graph neural network (GNN) [48]. In the following, we describe the GNN for point cloud $\mathbf{P}'$. The GNN for $\mathbf{Q}'$ is the same. First, the superpoints in $\mathbf{P}'$ are linked into a graph in Euclidean space with the k-NN method. Let $\mathbf{x}_i \in \mathbb{R}^b$ denote the feature encoding of superpoint $\mathbf{p}'_i$, and $(i,j) \in \mathcal{E}$ the graph edge between superpoints $\mathbf{p}'_i$ and $\mathbf{p}'_j$. The encoder features are then iteratively updated as

$$ {}^{(k+1)}\mathbf{x}_i = \max_{(i,j)\in\mathcal{E}} h_\theta\big(\mathrm{cat}[{}^{(k)}\mathbf{x}_i, {}^{(k)}\mathbf{x}_j - {}^{(k)}\mathbf{x}_i]\big) \;, \tag{2} $$

where $h_\theta(\cdot)$ denotes a linear layer followed by instance normalization [43] and a LeakyReLU activation [29], $\max(\cdot)$ denotes element-/channel-wise max-pooling, and $\mathrm{cat}[\cdot,\cdot]$ means concatenation. This update is performed twice with separate (not shared) parameters θ, and the final GNN features $\mathbf{x}_i^{\mathrm{GNN}} \in \mathbb{R}^{d_b}$ are obtained as

$$ \mathbf{x}_i^{\mathrm{GNN}} = h_\theta(\mathrm{cat}[{}^{(0)}\mathbf{x}_i, {}^{(1)}\mathbf{x}_i, {}^{(2)}\mathbf{x}_i]) \;. \tag{3} $$

Cross-attention block: Knowledge about potential overlap regions can only be gained by mixing information about both point clouds. To this end we adopt a cross-attention block [36] based on the message passing formulation [16]. First, each superpoint in $\mathbf{P}'$ is connected to all superpoints in $\mathbf{Q}'$ to form a bipartite graph. Inspired by the Transformer architecture [45], vector-valued queries $\mathbf{s}_i \in \mathbb{R}^b$ are used to retrieve the values $\mathbf{v}_j \in \mathbb{R}^b$ of other superpoints based on their keys $\mathbf{k}_j \in \mathbb{R}^b$, where

$$ \mathbf{k}_j = \mathbf{W}_k \mathbf{x}_j^{\mathrm{GNN}} \quad \mathbf{v}_j = \mathbf{W}_v \mathbf{x}_j^{\mathrm{GNN}} \quad \mathbf{s}_i = \mathbf{W}_s \mathbf{x}_i^{\mathrm{GNN}} \tag{4} $$

and $\mathbf{W}_k$, $\mathbf{W}_v$, and $\mathbf{W}_s$ are learnable weight matrices. The messages are computed as weighted averages of the values,

$$ \mathbf{m}_{i\leftarrow} = \sum_{j:(i,j)\in\mathcal{E}} a_{ij}\mathbf{v}_j \;, \tag{5} $$

with attention weights $a_{ij} = \mathrm{softmax}(\mathbf{s}_i^T \mathbf{k}_j / \sqrt{b})$ [36]. I.e., to update a superpoint $\mathbf{p}'_i$ one combines that point's query with the keys and values of all superpoints $\mathbf{q}'_j$. In line with the literature, in practice we use a multi-attention layer with four parallel attention heads [45]. The co-contextual features are computed as

$$ \mathbf{x}_i^{\mathrm{CA}} = \mathbf{x}_i^{\mathrm{GNN}} + \mathrm{MLP}(\mathrm{cat}[\mathbf{s}_i, \mathbf{m}_{i\leftarrow}]) \;, \tag{6} $$

with $\mathrm{MLP}(\cdot)$ denoting a three-layer fully connected network with instance normalization [43] and ReLU [30] activations after the first two layers. The same cross-attention block is also applied in reverse direction, so that information flows in both directions, $\mathbf{P}' \rightarrow \mathbf{Q}'$ and $\mathbf{Q}' \rightarrow \mathbf{P}'$.

Overlap scores of the bottleneck points: The above update with co-contextual information is done for each superpoint in isolation, without considering the local context within each point cloud. We therefore, explicitly update the local context after the cross-attention block using another GNN that has the same architecture and underlying graph (within-point cloud links) as above, but separate parameters θ. This yields the final latent feature encodings $\mathbf{F}^{\mathbf{P}'} \in \mathbb{R}^{N' \times b}$ and $\mathbf{F}^{\mathbf{Q}'} \in \mathbb{R}^{M' \times b}$, which are now conditioned on the features of the respective other point cloud. Those features are linearly projected to overlap scores $\mathbf{o}^{\mathbf{P}'} \in \mathbb{R}^{N'}$ and $\mathbf{o}^{\mathbf{Q}'} \in \mathbb{R}^{M'}$, which can be interpreted as probabilities that a certain superpoint lies in the overlap region. Additionally, one can compute *soft correspondences* between superpoints and from the correspondence weights predict the *cross-overlap score* of a superpoint $\mathbf{p}'_i$, i.e., the probability that its correspondence in $\mathbf{Q}'$ lies in the overlap region:

$$ \tilde{o}_i^{\mathbf{P}'} := \mathbf{w}_i^T \mathbf{o}^{\mathbf{Q}'}, \quad w_{ij} := \mathrm{softmax}\big(\frac{1}{t}\langle \mathbf{f}_i^{\mathbf{P}'}, \mathbf{f}_j^{\mathbf{Q}'} \rangle\big) \;, \tag{7} $$

where $\langle \cdot,\cdot \rangle$ is the inner product, and t is the temperature parameter that controls the soft assignment. In the limit $t \rightarrow 0$, Eq. (7) converges to hard nearest-neighbour assignment.

3.4. Decoder

Our decoder starts from conditioned features $\mathbf{F}^{\mathbf{P}'}$, concatenates them with the overlap scores $\mathbf{o}^{\mathbf{P}'}$, $\tilde{\mathbf{o}}^{\mathbf{P}'}$, and outputs per-point feature descriptors $\mathbf{F}^{\mathbf{P}} \in \mathbb{R}^{N \times 32}$ and refined

per-point overlap and matchability scores $\mathbf{o}^{\mathbf{P}}, \mathbf{m}^{\mathbf{P}} \in \mathbb{R}^N$. The matchability can be seen as a "conditional saliency" that quantifies how likely a point is to be matched correctly, given the points (resp. features) in the other point cloud $\mathbf{Q}$.

The decoder architecture combines NN-upsampling with linear layers, and includes skip connections from the corresponding encoder layers. We deliberately keep the overlap score and the matchability separate to disentangle the reasons why a point is a good/bad candidate for matching: in principle a point can be unambiguously matchable but lie outside the overlap region, or it can lie in the overlap but have an ambiguous descriptor. Empirically, we find that the network learns to predict high matchability mostly for points in the overlap; probably reflecting the fact that the ground truth correspondences used for training, naturally, always lie in the overlap. For further details about the architecture, please refer to Appendix and the source code.

3.5. Loss function and training

PREDATOR is trained end-to-end, using three losses w.r.t. ground truth correspondences as supervision.

Circle loss: To supervise the point-wise feature descriptors we follow[3] [3] and use the circle loss [39], a variant of the more common triplet loss. Consider again a pair of overlapping point clouds $\mathbf{P}$ and $\mathbf{Q}$, this time aligned with the ground truth transformation. We start by extracting the points $\mathbf{p}_i \in \mathbf{P}_p \subset \mathbf{P}$ that have at least one (possibly multiple) correspondence in $\mathbf{Q}$, where the set of correspondences $\mathcal{E}_p(\mathbf{p}_i)$ is defined as points in $\mathbf{Q}$ that lie within a radius r_p around $\mathbf{p}_i$. Similarly, all points of $\mathbf{Q}$ outside a (larger) radius r_s form the set of negatives $\mathcal{E}_n(\mathbf{p}_i)$. The circle loss is then computed from n_p points sampled randomly from $\mathbf{P}_p$:

最小化类内
最大化类间

$$\mathcal{L}_c^{\mathbf{P}} = \frac{1}{n_p}\sum_{i=1}^{n_p} \log\Big[1 + \sum_{j\in\mathcal{E}_p} e^{\beta_p^j (d_i^j - \Delta_p)} \cdot \sum_{k\in\mathcal{E}_n} e^{\beta_n^k (\Delta_n - d_i^k)}\Big], \tag{8}$$

where $d_i^j = ||\mathbf{f}_{\mathbf{p}_i} - \mathbf{f}_{\mathbf{q}_j}||_2$ denotes distance in feature space, and Δ_n, Δ_p are negative and positive margins, respectively. The weights $\beta_p^j = \gamma(d_i^j - \Delta_p)$ and $\beta_n^k = \gamma(\Delta_n - d_i^k)$ are determined individually for each positive and negative example, using the empirical margins $\Delta_p := 0.1$ and $\Delta_n := 1.4$ with hyper-parameter γ. The reverse loss $\mathcal{L}_c^{\mathbf{Q}}$ is computed in the same way, for a total circle loss $\mathcal{L}_c = \frac{1}{2}(\mathcal{L}_c^{\mathbf{P}} + \mathcal{L}_c^{\mathbf{Q}})$.

Overlap loss: The estimation of the overlap probability is cast as binary classification and supervised using the overlap loss $\mathcal{L}_o = \frac{1}{2}(\mathcal{L}_o^{\mathbf{P}} + \mathcal{L}_o^{\mathbf{Q}})$, where

$$\mathcal{L}_o^{\mathbf{P}} = \frac{1}{|\mathbf{P}|}\sum_{i=1}^{|\mathbf{P}|} \bar{o}_{\mathbf{p}_i} \log(o_{\mathbf{p}_i}) + (1 - \bar{o}_{\mathbf{p}_i}) \log(1 - o_{\mathbf{p}_i}). \tag{9}$$

[3] Added to the repository after publication, not mentioned in the paper.

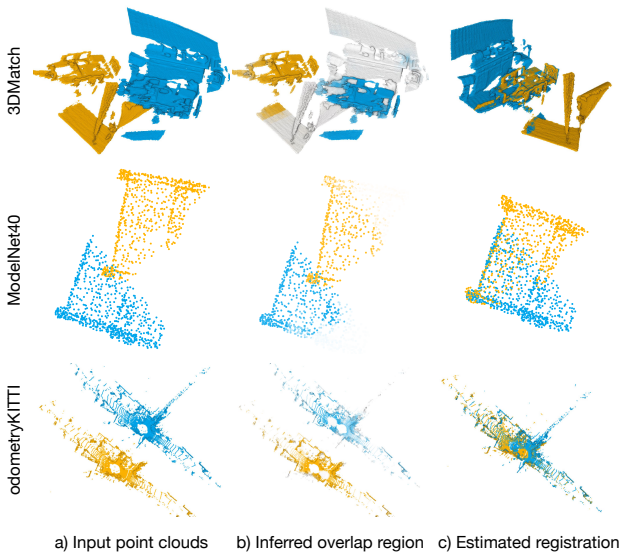

Figure 5: Example results of PREDATOR that succeeds in attending to the overlap region to enable robust registration.

The ground truth label $\bar{o}_{\mathbf{p}_i}$ of point $\mathbf{p}_i$ is defined as

$$\bar{o}_{\mathbf{p}_i} = \begin{cases} 1, & ||\bar{\mathbf{T}}_{\mathbf{P}}^{\mathbf{Q}}(\mathbf{p}_i) - \mathsf{NN}(\bar{\mathbf{T}}_{\mathbf{P}}^{\mathbf{Q}}(\mathbf{p}_i), \mathbf{Q})||_2 < r_o \\ 0, & \text{otherwise} \end{cases}, \tag{10}$$

with overlap threshold r_o. The reverse loss $\mathcal{L}_o^{\mathbf{Q}}$ is computed in the same way. The contributions from positive and negative examples are balanced with weights inversely proportional to their relative frequencies.

Matchability loss: Supervising the matchability scores is more difficult, as it is not clear in advance which are the right points to take into account during correspondence search. We follow a simple intuition: good keypoints are those that can be matched successfully at a given point during training, with the current feature descriptors. Hence, we cast the prediction as binary classification and generate the ground truth labels on the fly. Again, we sum the two symmetric losses, $\mathcal{L}_m = \frac{1}{2}(\mathcal{L}_m^{\mathbf{P}} + \mathcal{L}_m^{\mathbf{Q}})$, with

$$\mathcal{L}_m^{\mathbf{P}} = \frac{1}{|\mathbf{P}|}\sum_{i=1}^{|\mathbf{P}|} \overline{m}_{\mathbf{p}_i} \log(m_{\mathbf{p}_i}) + (1 - \overline{m}_{\mathbf{p}_i}) \log(1 - m_{\mathbf{p}_i}), \tag{11}$$

where ground truth labels $\overline{m}_{\mathbf{p}_i}$ are computed on the fly via nearest neighbour search $\mathsf{NN}_{\mathbf{F}}(\cdot,\cdot)$ in feature space:

$$\overline{m}_{\mathbf{p}_i} = \begin{cases} 1, & ||\bar{\mathbf{T}}_{\mathbf{P}}^{\mathbf{Q}}(\mathbf{p}_i) - \mathsf{NN}_{\mathbf{F}}(\mathbf{p}_i, \mathbf{Q})||_2 < r_m \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

Implementation and training: PREDATOR is implemented in pytorch and can be trained on a single RTX 3090 GPU. At the start of the training we supervise PREDATOR only with the circle and overlap losses, the matchability loss

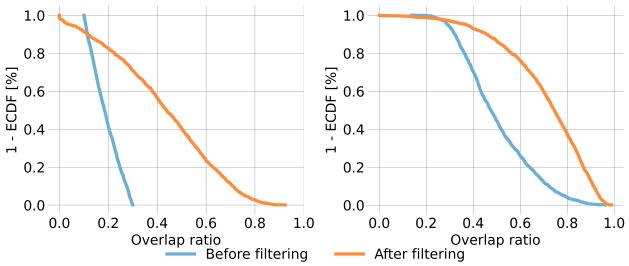

Figure 6: Distribution of the relative overlap ratio before and after filtering the points with the inferred overlap scores, *3DLoMatch* (left) and *3DMatch* (right).

is added only after few epochs, when the point-wise features are already meaningful (i.e., >30% of interest points can be matched correctly). The three loss terms are weighted equally. For more details, please refer to Appendix.

4. Experiments

We evaluate PREDATOR and justify our design choices on real point clouds, using *3DMatch* [56] and *3DLoMatch* (§ 4.1). Additionally, we compare PREDATOR to direct registration methods on the synthetic, object-centric *ModelNet40* [50] (§ 4.2) and evaluate it on large outdoor scenes using odometryKITTI [15] (§ 4.3). More details about the datasets and evaluation metrics are available in the Appndix. Qualitative results are shown in Fig. 5.

4.1. 3DMatch

Dataset: [56] is a collection of 62 scenes, from which we use 46 scenes for training, 8 scenes for validation and 8 for testing. Official *3DMatch* dataset considers only scan pairs with >30% overlap. Here, we add its counterpart in which we consider only scan pairs with overlaps between 10 and 30% and call this collection *3DLoMatch*[4].

Metrics: Our main metric, corresponding to the actual aim of point cloud registration, is *Registration Recall (RR)*, i.e., the fraction of scan pairs for which the correct transformation parameters are found with RANSAC. Following the literature [56, 19, 9], we also report *Feature Match Recall (FMR)*, defined as the fraction of pairs that have >5% "inlier" matches with <10 cm residual under the ground truth transformation (without checking if the transformation can be recovered from those matches), and *Inlier Ratio (IR)*, the fraction of correct correspondences among the putative matches. Additionally, we use empirical cumulative distribution functions (ECDF) to evaluate the relative overlap ratio. At a specific overlap value, the $(1-\text{ECDF})$ curve shows the fraction of fragment pairs that have relative overlap greater or equal to that value.

[4]Due to a bug in the official implementation of the overlap computation for *3DMatch*, a few (<7%) scan pairs are included in both datasets.

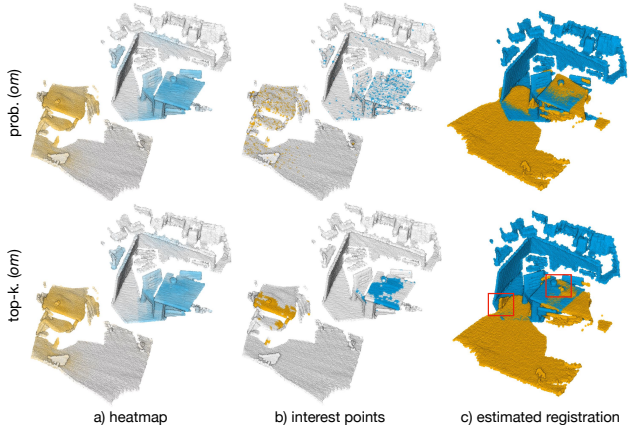

Figure 7: *Top-k (om)* sampling yields clustered interest points, whereas the points obtained with *prob. (om)* sampling are more scattered and thus enable a more robust estimation of the transformation parameters.

	3DMatch					*3DLoMatch*				
# Samples (k)	5000	2500	1000	500	250	5000	2500	1000	500	250
	Inlier ratio (%)									
rand	51.6	49.5	44.5	38.9	32.1	20.4	19.2	16.8	14.3	11.5
top-k (om)	**68.4**	**73.8**	**77.6**	**78.6**	**78.7**	**33.7**	**39.9**	**44.9**	**47.0**	**47.7**
prob. (om)	58.0	58.4	57.1	54.1	49.3	26.7	28.1	28.3	27.5	25.8
	Registration Recall (%)									
rand	86.0	84.8	84.7	81.7	75.3	43.3	45.3	40.4	35.9	28.0
top-k (om)	88.9	87.4	82.0	75.6	64.0	58.5	57.8	53.1	44.9	35.9
prob. (om)	**89.0**	**89.9**	**90.6**	**88.5**	**86.6**	**59.8**	**61.2**	**62.4**	**60.8**	**58.1**

Table 1: Performance of PREDATOR with different interest point sampling strategies; *om* denotes the product of overlap score and matchability score.

Relative overlap ratio: We first evaluate if PREDATOR achieves its goal to focus on the overlap. We discard points with a predicted overlap score $\mathbf{o}_i < 0.5$, compute the overlap ratio, and compare it to the one of the original scans. Fig. 6 shows that more than half (71%) of the low-overlap pairs are pushed over the 30% threshold that prior works considered the lower limit for registration. On average, discarding points with low overlap scores almost doubles the overlap in *3DLoMatch* (133% increase). Notably, it also increases the overlap in standard *3DMatch* by, on average, >50%.

Interest point sampling: PREDATOR significantly increases the effective overlap, but does that improve registration performance? To test this we use the product of the overlap scores $\mathbf{o}$ and matchability scores $\mathbf{m}$ to bias interest point sampling. We compare two variants: *top-k (om)*, where we pick the top-k points according to the multiplied scores; and *prob. (om)*, where we instead sample points with probability proportional to the multiplied scores.

For a more comprehensive assessment we follow [3] and report performance with different numbers of sampled interest points. Tab. 1 shows that any of the informed sampling strategies greatly increases the *inlier ratio*, and as

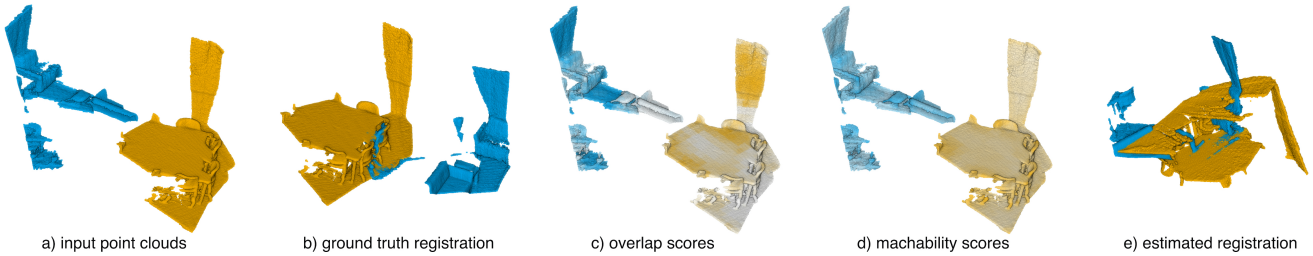

Figure 8: An extreme case where the overlap is insufficient for registration even with the proposed attention mechanism.

	3DMatch					*3DLoMatch*				
# Samples	5000	2500	1000	500	250	5000	2500	1000	500	250
	Registration Recall (%)									
3DSN [18]	78.4	76.2	71.4	67.6	50.8	33.0	29.0	23.3	17.0	11.0
FCGF [9]	85.1	84.7	83.3	81.6	71.4	40.1	41.7	38.2	35.4	26.8
D3Feat [3]	81.6	84.5	83.4	82.4	77.9	37.2	42.7	46.9	43.8	39.1
PREDATOR	**89.0**	**89.9**	**90.6**	**88.5**	**86.6**	**59.8**	**61.2**	**62.4**	**60.8**	**58.1**

Table 2: Results on the *3DMatch* and *3DLoMatch* datasets.

overlap attention			*3DMatch*			*3DLoMatch*		
ov.	$\times$*ov.*	*cond.*	FMR	IR	RR	FMR	IR	RR
			96.4	39.6	82.6	72.2	14.5	38.9
✓			94.6	38.3	84.1	67.1	14.3	42.8
✓	✓		96.4	50.8	87.7	73.8	20.9	56.5
✓		✓	95.7	52.1	88.0	72.5	21.2	57.5
✓	✓	✓	**96.7**	**58.0**	**89.0**	**78.6**	**26.7**	**59.8**

Table 3: Ablation of the network architecture. *ov.* denotes upsampling the overlap scores; *cond.* denotes conditioning the bottleneck features on the respective other point cloud; $\times$*ov.* denotes upsampling the cross overlap scores.

a consequence also the *registration recall*. The gains are larger when fewer points are sampled. In the low-overlap regime the inlier ratios more than triple for up to 1000 points. We observe that, as expected, high inlier ratio does not necessarily imply high registration recall: our scores are apparently well calibrated, so that *top-k (om)* indeed finds most inliers, but these are often clustered and too close to each other to reliably estimate the transformation parameters (Fig. 7). We thus use the more robust *prob. (om)* sampling, which yields the best *registration recall*. It may be possible to achieve even higher registration recall by combining *top-k (om)* sampling with non-maxima suppression. We leave this for future work.

Comparison to feature-based methods: We compare PREDATOR to recent feature-based registration methods: 3DSN [19], FCGF [9] and D3Feat [3], see Tab. 2. Even though PREDATOR can not solve all the cases (*cf*. Fig. 8), it greatly outperforms existing methods on the low-overlap *3DLoMatch* dataset, improving registration recall by 15.5-19.7 percent points (pp) over the closest competitor—variously FCGF or 3DFeat. Moreover, it also consistently reaches the highest registration recall on standard *3DMatch*, showing that its attention to the overlap pays off even for scans with moderately large overlap. In line with our motivation, what matters is not so much the choice of descriptors, but finding interest points that lie in the overlap region – especially if that region is small.

Comparison to direct registration methods: We also tried to compare PREDATOR to recent methods for direct registration of partial point clouds. Unfortunately, for both PRNet [47] and RPM-Net [55], training on *3DMatch* failed to converge to reasonable results, as already observed in [7]. It appears that their feature extraction is specifically tuned to synthetic, object-centric point clouds. Thus, in a further attempt we replaced the feature extractor of RPM-Net with FCGF. This brought the registration recall on *3DMatch* to 54.9%, still far from the 85.1% that FCGF features achieve with RANSAC. We conclude that direct pairwise registration is at this point only suitable for geometrically simple objects in controlled settings like *ModelNet40*.

Ablations study: We ablate our overlap attention module in Tab. 3. We first compare PREDATOR with a baseline model, in which we completely remove the proposed overlap attention module. That baseline, combined with random sampling, achieves the 2$^{\text{nd}}$-highest FMR on both benchmarks, but only reaches 82.6%, respectively 38.9% RR. By adding the overlap scores, RR increases by 1.5, respectively 3.9 pp on *3DMatch* and *3DLoMatch*. Additionally upsampling conditioned feature scores or cross overlap scores further improves performance, especially on *3DLoMatch*. All three parts combined lead to the best overall performance. For further ablation studies, see Appendix.

4.2. ModelNet40

Dataset: [50] contains 12,311 CAD models of man-made objects from 40 different categories. We follow [55] to use 5,112 samples for training, 1,202 samples for validation, and 1,266 samples for testing. Partial scans are generated following [55]. In addition to *ModelNet* which has 73.5% pairwise overlap on average, we generate *ModelLoNet* with lower (53.6%) average overlap. For more details see Appendix.

Metrics: We follow [55] and measure the performance using the *Relative Rotation Error (RRE)* (geodesic distance between estimated and GT rotation matrices), the *Relative Translation Error (RTE)* (Euclidean distance between the

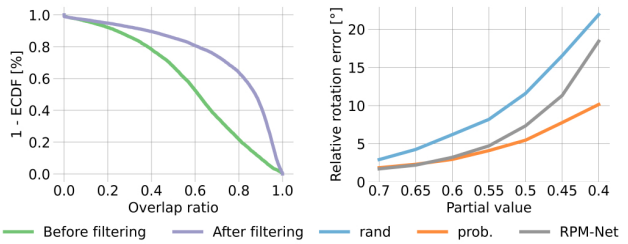

Figure 9: Improved relative overlap ratio after filtering the points with the inferred overlap scores on 8862 *ModelNet* partial scans(*left*). Owing to the improved overlap ratio, PREDATOR is robust to the changes of partial value p_v, while the performance of RPM-Net drops rapidly (*right*). *rand* and *prob.* denote the random and *prob. (om)* biased sampling of 450 interest points, respectively.

	ModelNet			*ModelLoNet*		
Methods	RRE	RTE	*CD*	RRE	RTE	*CD*
DCP-v2 [46]	11.975	0.171	0.0117	16.501	0.300	0.0268
RPM-Net [55]	**1.712**	**0.018**	**0.00085**	7.342	**0.124**	**0.0050**
PREDATOR (*rand*)	2.407	0.028	0.00120	10.985	0.175	0.0097
PREDATOR (*prob. (om)*)	1.739	0.019	0.00089	**5.235**	0.132	0.0083

Table 4: Evaluation results on *ModelNet* and *ModelLoNet*. 450 points are sampled for RANSAC with *rand* / *prob.*.

estimated and GT translations), and the *Chamfer distance (CD)* between the two registered scans.

Relative overlap ratio: We again evaluate if PREDATOR focuses on the overlap region. We extract 8,862 test pairs by varying the completeness of the input point clouds from 70 to 40%. Fig. 9 shows that PREDATOR substantially increases the relative overlap and reduces the number of pairs with overlap <70% by more than 40 pp.

Comparison to direct registration methods: To be able to compare PREDATOR to RPM-Net [55] and DCP [46], we resort to the synthetic, object-centric dataset they were designed for. We failed to train PRNet [47] due to random crashes of the original code (also observed in [7]).

Remarkably, PREDATOR can compete with methods specifically tuned for *ModelNet*, and in the low-overlap regime outperforms them in terms of *RRE*, see Tab. 4. Moreover, we observe a large boost by sampling points with overlap attention (*prob. (om)*) rather than randomly (*rand*). Fig. 9 (right) further underlines the importance of sampling in the overlap: PREDATOR is a lot more robust in the low overlap regime (≈8° lower RRE at completeness 0.4).

4.3. odometryKITTI

Dataset: [15] contains 11 sequences of LiDAR-scanned outdoor driving scenarios. We follow [9] and use sequences 0-5 for training, 6-7 for validation, and 8-10 for testing. In line with [9, 3] we further refine the provided ground truth poses using ICP [5] and only use point cloud pairs that are at most 10 m away from each other for evaluation.

Method	*RTE [cm]* ↓	*RRE [°]* ↓	RR ↑
3DFeat-Net [54]	25.9	0.57	96.0
FCGF [9]	9.5	0.30	96.6
D3Feat* [3]	7.2	0.30	**99.8**
PREDATOR (*rand*)	8.8	0.34	**99.8**
PREDATOR (*prob. (om)*)	**6.8**	**0.27**	**99.8**

Table 5: Evaluation of PREDATOR on *odometryKITTI*, following the evaluation protocol employed by D3Feat [3].

Comparision to the SoTAs: We compare PREDATOR to 3DFeat-Net [54], FCGF [9] and D3Feat* [3][5] As shown in Tab. 5, PREDATOR performs on-par with the SoTA. The results also corroborate the impact of our overlap attention which again outperforms the random sampling baseline.

Computational complexity: With $O(n^2)$ complexity the cross-attention module represents the memory bottleneck of PREDATOR. Furthermore, n cannot be selected freely but results from the interplay of (i) the resolution of the initial voxel grid, (ii) the network architecture (number of strided convolution layers), and (iii) the spatial extent of the scene. Nevertheless, by executing the cross-attention at the *superpoint* level, with greatly reduced n, we are able to apply PREDATOR to large outdoor scans like *odometryKITTI* on a single GPU. For even larger scenes, a simple engineering trick could be to split them into parts, as often done for semantic segmentation.

5. Conclusion

We have introduced PREDATOR, a deep model designed for pairwise registration of low-overlap point clouds. The core of the model is an overlap attention module that enables early information exchange between the point clouds' latent encodings, in order to infer which of their points are likely to lie in their overlap region.

There are a number of directions in which PREDATOR could be extended. At present it is tightly coupled to fully convolutional point cloud encoders, and relies on having a reasonable number of *superpoints* in the bottleneck. This could be a limitation in scenarios where the point density is very uneven. It would also be interesting to explore how our overlap-attention module can be integrated into direct point cloud registration methods and other neural architectures that have to handle two inputs with low overlap, e.g. in image matching [36]. Finally, registration in the low-overlap regime is challenging and PREDATOR cannot solve all the cases. A user study could provide a better understanding of how PREDATOR compares to human operators.

Acknowledgements. This work was sponsored by the NVIDIA GPU grant.

[5] We find that the released D3Feat code fails to reproduce the results in the paper, possible due to hyper-parameter changes.

References

[1] Yasuhiro Aoki, Hunter Goforth, Rangaprasad Arun Srivatsan, and Simon Lucey. PointnetLK: Robust & efficient point cloud registration using Pointnet. In *CVPR*, 2019. 2

[2] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE TPAMI*, 9(5):698–700, 1987. 2

[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 *CVPR*, 2020. 1, 2, 5, 6, 7, 8, 12, 13

[4] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *ICML*, 2009. 12

[5] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992. 8

[6] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. Robust reconstruction of indoor scenes. In *CVPR*, 2015. 1, 11

[7] Christopher Choy, Wei Dong, and Vladlen Koltun. Deep global registration. In *CVPR*, 2020. 2, 7, 8

[8] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, 2019. 2, 3

[9] Christopher Choy, Jaesik Park, and Vladlen Koltun. Fully convolutional geometric features. In *ICCV*, 2019. 1, 2, 6, 7, 8, 13

[10] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *ACM SIGGRAPH*, 1996. 11

[11] Haowen Deng, Tolga Birdal, and Slobodan Ilic. PPFFoldNet: Unsupervised learning of rotation invariant 3d local descriptors. In *ECCV*, 2018. 2

[12] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppfnet: Global context aware local features for robust 3d point matching. In *CVPR*, 2018. 2, 11

[13] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPR Workshops*, 2018. 2

[14] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-Net: A trainable CNN for joint detection and description of local features. In *CVPR*, 2019. 2

[15] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 6, 8

[16] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *ICML*, 2017. 4

[17] Zan Gojcic, Caifa Zhou, Jan D Wegner, Leonidas J Guibas, and Tolga Birdal. Learning multiview 3d point cloud registration. In *CVPR*, 2020. 2

[18] Zan Gojcic, Caifa Zhou, Jan D Wegner, and Andreas Wieser. The perfect match: 3d point cloud matching with smoothed densities. In *CVPR*, 2019. 1, 2, 7, 13

[19] Zan Gojcic, Caifa Zhou, and Andreas Wieser. Learned compact local feature descriptor for TLS-based geodetic monitoring of natural outdoor scenes. In *ISPRS Annals*, 2018. 2, 6, 7

[20] Yulan Guo, Mohammed Bennamoun, Ferdous Sohel, Min Lu, Jianwei Wan, and Jun Zhang. Performance evaluation of 3D local feature descriptors. In *ACCV*, 2014. 2

[21] Maciej Halber and Thomas A. Funkhouser. Structured global registration of RGB-D scans in indoor environments. *arXiv preprint arXiv:1607.08539*, 2016. 11

[22] A.E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE TPAMI*, 21:433–449, 1999. 2

[23] Marc Khoury, Qian-Yi Zhou, and Vladlen Koltun. Learning compact geometric features. In *ICCV*, 2017. 1, 2, 3

[24] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2017. 3

[25] Kevin Lai, Liefeng Bo, and Dieter Fox. Unsupervised feature learning for 3d scene labeling. In *ICRA*, 2014. 11

[26] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 2

[27] Fan Lu, Guang Chen, Yinlong Liu, Zhongnan Qu, and Alois Knoll. Rskdd-net: Random sample-based keypoint detector and descriptor. *NeurIPS*, 2020. 2

[28] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI*, 1981. 2

[29] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013. 4

[30] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010. 4

[31] G Dias Pais, Srikumar Ramalingam, Venu Madhav Govindu, Jacinto C Nascimento, Rama Chellappa, and Pedro Miraldo. 3DRegNet: A deep neural network for 3d point registration. In *CVPR*, 2020. 2

[32] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 2

[33] Jerome Revaud, Philippe Weinzaepfel, César De Souza, Noe Pion, Gabriela Csurka, Yohann Cabon, and Martin Humenberger. R2D2: Repeatable and reliable detector and descriptor. *arXiv preprint arXiv:1906.06195*, 2019. 2

[34] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (FPFH) for 3D registration. In *ICRA*, 2009. 2

[35] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz. Aligning point cloud views using persistent feature histograms. In *IROS*, 2008. 2

[36] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 3, 4, 8

[37] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In *CVPR*, 2013. 11

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

[39] Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. Circle loss: A unified perspective of pair similarity optimization. In *CVPR*, 2020. 5

[40] 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 *CVPR*, 2019. 3

[41] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique shape context for 3D data description. In *ACM Workshop on 3D Object Retrieval*, 2010. 2

[42] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique signatures of histograms for local surface description. In *ECCV*, 2010. 2

[43] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 4

[44] Julien Valentin, Angela Dai, Matthias Nießner, Pushmeet Kohli, Philip Torr, Shahram Izadi, and Cem Keskin. Learning to navigate the energy landscape. In *3DV*, 2016. 11

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 4

[46] Yue Wang and Justin M Solomon. Deep closest point: Learning representations for point cloud registration. In *ICCV*, 2019. 2, 3, 8

[47] Yue Wang and Justin M Solomon. PRNet: Self-supervised learning for partial-to-partial registration. In *NeurIPS*, 2019. 2, 3, 7, 8

[48] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph CNN for learning on point clouds. *ACM TOG*, 38(5), 2019. 2, 4

[49] Olivia Wiles, Sebastien Ehrhardt, and Andrew Zisserman. D2D: Learning to find good correspondences for image matching and manipulation. *arXiv preprint arXiv:2007.08480*, 2020. 2, 3

[50] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015. 6, 7

[51] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. Sun3d: A database of big spaces reconstructed using sfm and object labels. In *ICCV*, 2013. 11

[52] Zhenpei Yang, Jeffrey Z Pan, Linjie Luo, Xiaowei Zhou, Kristen Grauman, and Qixing Huang. Extreme relative pose estimation for rgb-d scans via scene completion. In *CVPR*, 2019. 1

[53] Zhenpei Yang, Siming Yan, and Qixing Huang. Extreme relative pose network under hybrid representations. In *CVPR*, 2020. 1

[54] Zi Jian Yew and Gim Hee Lee. 3dfeat-net: Weakly supervised local 3d features for point cloud registration. In *ECCV*, pages 630–646. Springer, 2018. 2, 8

[55] Zi Jian Yew and Gim Hee Lee. RPM-Net: Robust point matching using learned features. In *CVPR*, 2020. 2, 7, 8, 11

[56] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3DMatch: learning local geometric descriptors from RGB-D reconstructions. In *CVPR*, 2017. 1, 2, 3, 6, 11

A. Appendix

In this supplementary material, we first provide rigorous definitions of evaluation metrics (Sec. A.1), then describe the data pre-processing step (Sec. A.2), network architectures (Sec. A.4) and training on individual datasets (Sec. A.3) in more detail. We further provide additional results (Sec. A.5), ablation studies (Sec. A.6) as well as a runtime analysis (Sec. A.7). Finally, we show more visualisations on *3DLoMatch* and *ModelLoNet* benchmarks (Sec. A.8).

A.1. Evaluation metrics

The evaluation metrics, which we use to assess model performance in Sec. 4 of the main paper and Sec. A.5 of this supplementary material, are formally defined as follows:

Inlier ratio looks at the set of putative correspondences $(\mathbf{p}, \mathbf{q}) \in \mathcal{K}_{ij}$ found by reciprocal matchingin feature space, and measures what fraction of them is "correct", in the sense that they lie within a threshold $\tau_1 = 10\,\text{cm}$ after registering the two scans with the ground truth transformation $\overline{T}_{\mathbf{P}}^{\mathbf{Q}}$:

$$\text{IR} = \frac{1}{|\mathcal{K}_{ij}|} \sum_{(\mathbf{p},\mathbf{q})\in\mathcal{K}_{ij}} \left[||\overline{\mathbf{T}}_{\mathbf{P}}^{\mathbf{Q}}(\mathbf{p}) - \mathbf{q}||_2 < \tau_1 \right], \tag{13}$$

with $[\cdot]$ the Iverson bracket.

Feature Match recall (FMR) [12] measures the fraction of point cloud pairs for which, based on the number of inlier correspondences, it is *likely* that accurate transformation parameters can be recovered with a robust estimator such as RANSAC. Note that FMR only checks whether the inlier ratio is above a threshold $\tau_2 = 0.05$. It does not test if the transformation can actually be determined from those correspondences, which in practice is not always the case, since their geometric configuration may be (nearly) degenerate, e.g., they might lie very close together or along a straight edge. A single pair of point clouds counts as suitable for registration if

$$IR > \tau_2 \tag{14}$$

Registration recall [6] is the most reliable metric, as it measures end-to-end performance on the actual task of point cloud registration. Specifically, it looks at the set of ground truth correspondences $\mathcal{H}_{ij}^*$ after applying the estimated transformation $T_{\mathbf{P}}^{\mathbf{Q}}$, computes their root mean square error,

$$\text{RMSE} = \sqrt{\frac{1}{|\mathcal{H}_{ij}^*|} \sum_{(\mathbf{p},\mathbf{q})\in\mathcal{H}_{ij}^*} ||\mathbf{T}_{\mathbf{P}}^{\mathbf{Q}}(\mathbf{p}) - \mathbf{q}||_2^2}, \tag{15}$$

and checks for what fraction of all point pairs $\text{RMSE} < 0.2$. In keeping with the original evaluation script of *3DMatch*, immediately adjacent point clouds are excluded, since they have very high overlap by construction.

Chamfer distance measures the quality of registration on synthetic data. We follow [55] and use the *modified* Chamfer distance metric:

$$\begin{aligned} \tilde{CD}(\mathbf{P},\mathbf{Q}) = &\frac{1}{|\mathbf{P}|} \sum_{\mathbf{p}\in\mathbf{P}} \min_{\mathbf{q}\in\mathbf{Q}_{\text{raw}}} \|\mathbf{T}_{\mathbf{P}}^{\mathbf{Q}}(\mathbf{p}) - \mathbf{q}\|_2^2 + \\ &\frac{1}{|\mathbf{Q}|} \sum_{\mathbf{q}\in\mathbf{Q}} \min_{\mathbf{p}\in\mathbf{P}_{\text{raw}}} \|\mathbf{q} - \mathbf{T}_{\mathbf{P}}^{\mathbf{Q}}(\mathbf{p})\|_2^2 \end{aligned} \tag{16}$$

where $\mathbf{P}_{\text{raw}} \in \mathbb{R}^{2048\times 3}$ and $\mathbf{Q}_{\text{raw}} \in \mathbb{R}^{2048\times 3}$ are *raw* source and target point clouds, $\mathbf{P} \in \mathbb{R}^{717\times 3}$ and $\mathbf{Q} \in \mathbb{R}^{717\times 3}$ are *input* source and target point clouds.

Relative translation and rotation errors (RTE/RRE) measures the deviations from the ground truth pose as:

$$\begin{aligned} \text{RTE} &= \|\mathbf{t} - \bar{\mathbf{t}}\|_2 \\ \text{RRE} &= \arccos\left(\frac{\text{trace}(\mathbf{R}^T\bar{\mathbf{R}}) - 1}{2}\right) \end{aligned} \tag{17}$$

where $\mathbf{R}$ and $\mathbf{t}$ denote the estimated rotation matrix and translation vector, respectively.

Empirical Cumulative Distribution Function (ECDF) measures the distribution of a set of values:

$$\text{ECDF}(x) = \frac{\left|\{o_i < x\}\right|}{|O|} \tag{18}$$

where O is a set of values(ovelap ratios in our case) and $x \in [\min\{O\}, \max\{O\}]$.

A.2. Dataset preprocessing

3DMatch: [56] is a collection of 62 scenes, combining earlier data from Analysis-by-Synthesis [44], 7Scenes [37], SUN3D [51], RGB-D Scenes v.2 [25], and Halber *et al*. [21]. The official benchmark splits the data into 54 scenes for training and 8 for testing. Individual scenes are not only captured in different indoor spaces (e.g., bedrooms, offices, living rooms, restrooms) but also with different depth sensors (e.g., Microsoft Kinect, Structure Sensor, Asus Xtion Pro Live, and Intel RealSense). *3DMatch* provides great diversity and allows our model to generalize across different indoor spaces. Individual scenes of *3DMatch* are split into point cloud fragments, which are generated by fusing 50 consecutive depth frames using TSDF volumetric fusion [10]. As a preprocessing step, we apply voxel-grid downsampling to all point clouds, and if multiple points fall into the same voxel, we randomly pick one.

ModelNet40: For each CAD model of *ModelNet40*, 2048 points are first generated by uniform sampling and scaled to fit into a unit sphere. Then we follow [55] to produce partial scans: for source partial point cloud, we uniformly

*First two authors contributed equally to this work.

	n_p	γ	V	r_p	r_s	r_o	r_m
3DMatch	256	24	0.025	0.0375	0.1	0.0375	0.05
ModelNet	384	64	0.06	0.018	0.06	0.04	0.04
odometryKITTI	512	48	0.3	0.21	0.75	0.45	0.3

Table 6: Hyper-parameters configurations for different datasets.

sample a plane through the origin that splits the unit sphere into two half-spaces, shift that plane along its normal until $\lfloor 2048 \cdot p_v \rfloor$ points are on one side, and discard the points on the other side; the target point cloud is generated in the same manner; then the two resulting, partial point clouds are randomly rotated, translated and jittered with Gaussian noise. For the rotation, we sample a random axis and a random angle $<45°$. The translation is sampled in the range $[-0.5, 0.5]$. Gaussian noise is applied per coordinate with $\sigma = 0.05$. Finally, 717 points are randomly sampled from the $\lfloor 2048 \cdot p_v \rfloor$ points.

odometryKITTI: The dataset was captured using a Velodyne HDL-64 3D laser scanner by driving around the mid-size city of Karlsruhe, in rural areas and on highways. The ground truth poses are provided by GPS/IMU system. We follow [3] to use ICP to reduce the noise in the ground truth poses.

A.3. Implementation and training

For 3DMatch/Modelnet/KITTI, we train PREDATOR using Stochastic Gradient Descent for 30/ 200/ 150 epochs, with initial learning rate 0.005/ 0.01/ 0.05, momentum 0.98, and weight decay 10^{-6}. The learning rate is exponentially decayed by 0.05 after each epoch. Due to memory constraints we use batch size 1 in all experiments. The dataset-dependent hyper-parameters which include number of negative pairs in circle loss n_p, temperature factor γ, voxel size V, search radius for positive pair r_p, safe radius r_s, overlap and matchability radius r_o and r_m are given in Tab. 6. On odometryKITTI dataset, we take the curriculum learning [4] strategy to gradually learn sharper local descriptors by adjusting n_p. For more details please see our code.

A.4. Network architecture

The detailed network architecture of PREDATOR is depicted in Fig. 11. Our model is built on the KPConv implementation from the D3Feat repository.[7] We complement each KPConv layer with instance normalisation Leaky ReLU activations. The l-th strided convolution is applied to a point cloud dowsampled with voxel size $2^l \cdot V$. Upsampling in the decoder is performed by querying the associated feature of the closest point from the previous layer.

With ≈20k points after voxel-grid downsampling, the point clouds in *3DMatch* are much denser than those of *ModelNet40* with only 717 points. Moreover, they also have larger spatial extent with bounding boxes up to $3 \times 3 \times 3$ m^3, while *ModelNet40* point clouds are normalised to fit into a unit sphere. To account for these large differences, we slightly adapt the encoder and decoder per dataset, but keep the same overlap attention model. Differences in network hyper-parameters are shown in Tab. 7.

[7] https://github.com/XuyangBai/D3Feat.pytorch

	# strided convolutions	convolution radius	first conv. feature dim.	final feature dim.
3DMatch	3	2.5	64	32
ModelNet	2	2.75	256	96
odometryKITTI	3	4.25	128	32

Table 7: Different network configurations for *3DMatch*, *ModelNet* and *odometryKITTI* datasets.

A.5. Additional results

Detailed registration results: We report detailed per-scene *Registration Recall (RR)*, *Relative Rotation Error (RRE)* and *Relative Translation Error (RTE)* in Tab. 8. RRE and RTE are only averaged over successfully registered pairs for each scene, such that the numbers are mot dominated by gross errors from complete registration failures. We get the highest RR and lowest or second lowest RTE and RRE for almost all scenes, this further shows that our overlap attention module together with probabilistic sampling supports not only robust, but also accurate registration.

Feature match recall: Finally, Fig. 10 shows that our descriptors are robust and perform well over a wide range of thresholds for the allowable inlier distance and the minimum inlier ratio. Notably, PREDATOR consistently outperforms D3Feat that uses a similar KPConv backbone.

A.6. Additional ablation studies

Ablations of matchability score: We find that probabilistic sampling guided by the product of the overlap and matchability scores attains the highest RR. Here we further analyse the impact of each individual component. We first construct a baseline which applies random sampling (*rand*) over conditioned features, then we sample points with probability proportional to overlap scores (*prob. (o)*), to matchability scores (*prob. (m)*), and to the combination of the two scores (*prob. (om)*). As shown in Tab. 9, *rand* fares clearly worse, in all metrics. Compared to *prob. (om)*, either *prob. (o)* or *prob. (m)* can achieve comparable results on *3DMatch*; the performance gap becomes big on the more challenging *3DLoMatch* dataset, where our *prob. (om)* is around 4 pp better in terms of RR.

Ablations of overlap attention module with FCGF: To demonstrate the flexibility of our model, we additionally add proposed overlap attention module to FCGF model. We train it on *3DMatch* dataset with our proposed loss for 100

	3DMatch										*3DLoMatch*									
	Kitchen	Home 1	Home 2	Hotel 1	Hotel 2	Hotel 3	Study	MIT Lab	Avg.	STD	Kitchen	Home 1	Home 2	Hotel 1	Hotel 2	Hotel 3	Study	MIT Lab	Avg.	STD
# Sample																				
	449	106	159	182	78	26	234	45	160	128	**524**	283	222	210	138	42	237	70	191	154
Registration Recall (%) ↑																				
3DSN [18]	90.6	90.6	65.4	89.6	82.1	80.8	68.4	60.0	78.4	11.5	51.4	25.9	44.1	41.1	30.7	36.6	14.0	20.3	33.0	11.8
FCGF [9]	**98.0**	94.3	68.6	**96.7**	91.0	**84.6**	76.1	71.1	85.1	11.0	60.8	42.2	53.6	53.1	38.0	26.8	16.1	30.4	40.1	14.3
D3Feat [3]	96.0	86.8	67.3	90.7	88.5	80.8	78.2	64.4	81.6	10.5	49.7	37.2	47.3	47.8	36.5	31.7	15.7	31.9	37.2	**10.6**
Ours	97.6	**97.2**	**74.8**	**98.9**	**96.2**	**88.5**	**85.9**	**73.3**	**89.0**	**9.6**	**71.5**	**58.2**	**60.8**	**77.5**	**64.2**	**61.0**	**45.8**	**39.1**	**59.8**	11.7
Relative Rotation Error (°) ↓																				
3DSN [18]	1.926	1.843	2.324	2.041	1.952	2.908	2.296	2.301	2.199	0.321	3.020	3.898	3.427	3.196	3.217	3.328	4.325	3.814	3.528	0.414
FCGF [9]	**1.767**	1.849	**2.210**	**1.867**	1.667	2.417	**2.024**	**1.792**	**1.949**	0.236	**2.904**	3.229	3.277	2.768	**2.801**	**2.822**	3.372	4.006	3.147	0.394
D3Feat [3]	2.016	2.029	2.425	1.990	1.967	**2.400**	2.346	2.115	2.161	**0.183**	3.226	3.492	3.373	3.330	3.165	2.972	3.708	3.619	3.361	**0.227**
Ours	1.861	**1.806**	2.473	2.045	**1.600**	2.458	2.067	1.926	2.029	0.286	3.079	**2.637**	**3.220**	**2.694**	2.907	3.390	**3.046**	**3.412**	**3.048**	0.273
Relative Translation Error (m) ↓																				
3DSN [18]	0.059	0.070	0.079	0.065	0.074	0.062	0.093	0.065	0.071	**0.010**	0.082	0.098	0.096	0.101	**0.080**	0.089	0.158	**0.120**	0.103	0.024
FCGF [9]	0.053	0.056	0.071	**0.062**	0.061	0.055	0.082	0.090	0.066	0.013	0.084	0.097	**0.076**	0.101	0.084	0.077	0.144	0.140	0.100	0.025
D3Feat [3]	0.055	0.065	0.080	0.064	0.078	**0.049**	0.083	0.064	0.067	0.011	0.088	0.101	0.086	0.099	0.092	**0.075**	0.146	0.135	0.103	0.023
Ours	**0.048**	**0.055**	**0.070**	0.073	**0.060**	0.065	**0.080**	**0.063**	**0.064**	0.010	**0.081**	**0.080**	0.084	**0.099**	0.096	0.077	**0.101**	0.130	**0.093**	**0.016**

Table 8: Detailed results on the *3DMatch* and *3DLoMatch* datasets.

matchability	overlap	*3DMatch* FMR	*3DMatch* IR	*3DMatch* RR	*3DLoMatch* FMR	*3DLoMatch* IR	*3DLoMatch* RR
		96.2	51.6	86.0	74.9	20.4	43.3
✓		96.1	54.0	**89.2**	75.5	21.9	52.2
	✓	96.2	56.7	89.1	78.3	26.1	57.4
✓	✓	**96.7**	**58.0**	89.0	**78.6**	**26.7**	**59.8**

Table 9: Different combinations of scores used for probabilistic sampling.

# Samples	*3DMatch* 5000	2500	1000	500	250	*3DLoMatch* 5000	2500	1000	500	250
Registration Recall (%)										
FCGF [9]	85.1	84.7	83.3	81.6	71.4	40.1	41.7	38.2	35.4	26.8
FCGF+OA	**89.1**	**88.9**	**88.7**	**87.5**	**85.4**	**57.8**	**58.3**	**59.8**	**58.7**	**55.9**

Table 10: Ablation of the proposed overlap attention module with sparse convolution backbone. FCGF + OA denotes adding proposed overlap attention module to FCGF model.

epochs, the results are shown in Tab. 10. It shows that FCGF can also greatly benefit from the overlap attention module. Registration recall almost doubles when sampling only 250 points on the challenging *3DLoMatch* benchmark.

A.7. Timings

We compare the runtime of PREDATOR with FCGF[8] [9] and D3Feat[9] [3] on *3DMatch*. For all three methods we set voxel size $V = 2.5$ cm and batch size 1. The test is run on a single GeForce GTX 1080 Ti with Intel(R) Core(TM) i7-7700K CPU @ 4.20GHz, 32GB RAM. The most time-consuming step of our model, and also of D3Feat, is the data loader, as we have to pre-compute the neighborhood indices before the forward pass. With its smaller encoder and decoder, but the additional overlap attention module, PREDATOR is still marginally faster than D3Feat. FCGF has a more efficient data loader that relies on sparse convolution and queries neighbors during the forward pass. See Tab. 11.

[8] All experiments were done with MinkowskiEngine v0.4.2.

[9] We use its PyTorch implementation.

	data loader	encoder	overlap attention	decoder	overall
FCGF [9]	**6**	414	—	25	445
D3Feat [3]	200	11	—	63	274
Ours	191	**9**	70	**1**	**271**

Table 11: Runtime per fragment pair in milli-seconds, averaged over 1623 test pairs of *3DMatch*.

A.8. Qualitative visualization

We show more qualitative results in Fig. 12 and Fig. 13 for *3DLoMatch* and *ModelLoNet* respectively. The input points clouds are rotated and translated here for better visualization of overlap and matchability scores.

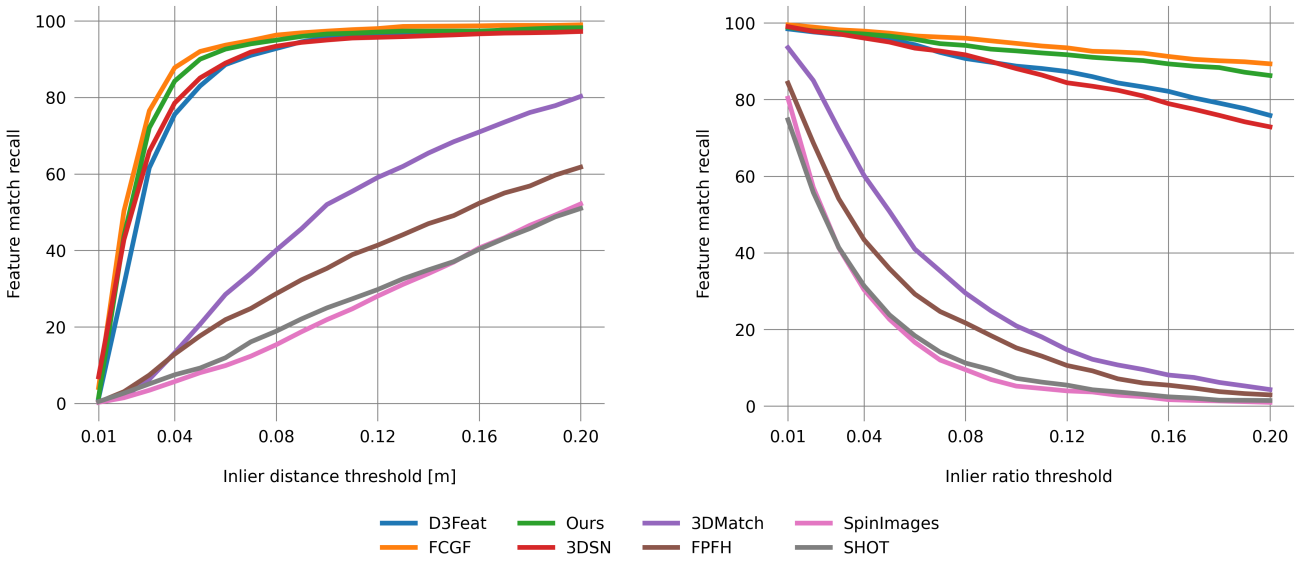

Figure 10: Feature matching recall in relation to inlier distance threshold τ_1 (left) and inlier ratio threshold τ_2 (right)

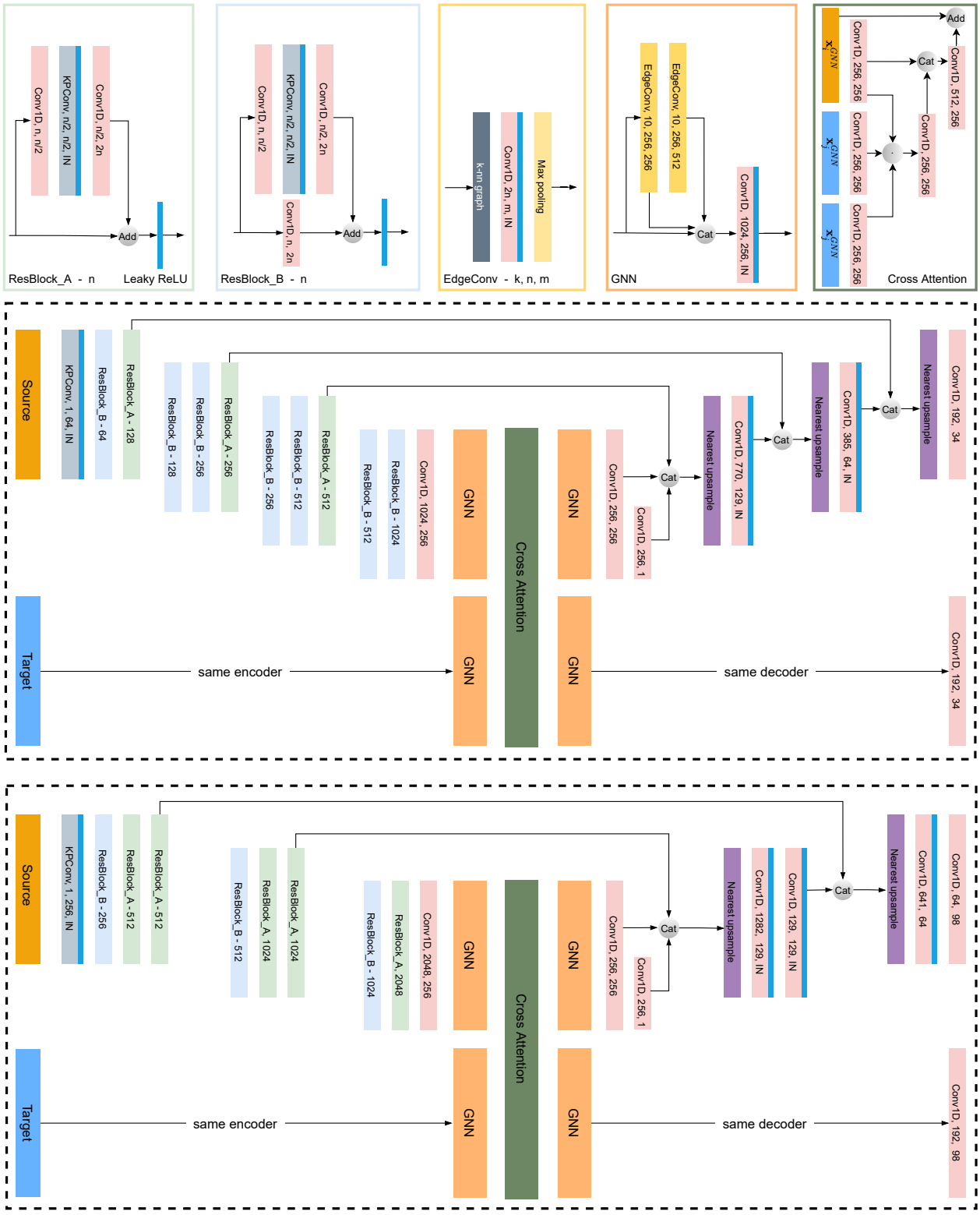

Figure 11: Network architecture of PREDATOR for *3DMatch* (*middle*) and *ModelNet* (*bottom*). In the cross attention module, for each (query $\mathbf{s}_i \in \mathbb{R}^{b\times 1}$, key $\mathbf{k}_i \in \mathbb{R}^{b\times 1}$, value $\mathbf{v}_i \in \mathbb{R}^{b\times 1}$), $\bigodot$ denotes first reshape them into shape $(4, \frac{b}{4})$(4 heads), then compute scores matrix $\mathbf{S}$ from $\mathbf{s}_i$ and $\mathbf{k}_i$, finally get message update from $\mathbf{v}_i$ and reshape back to $(b, 1)$.

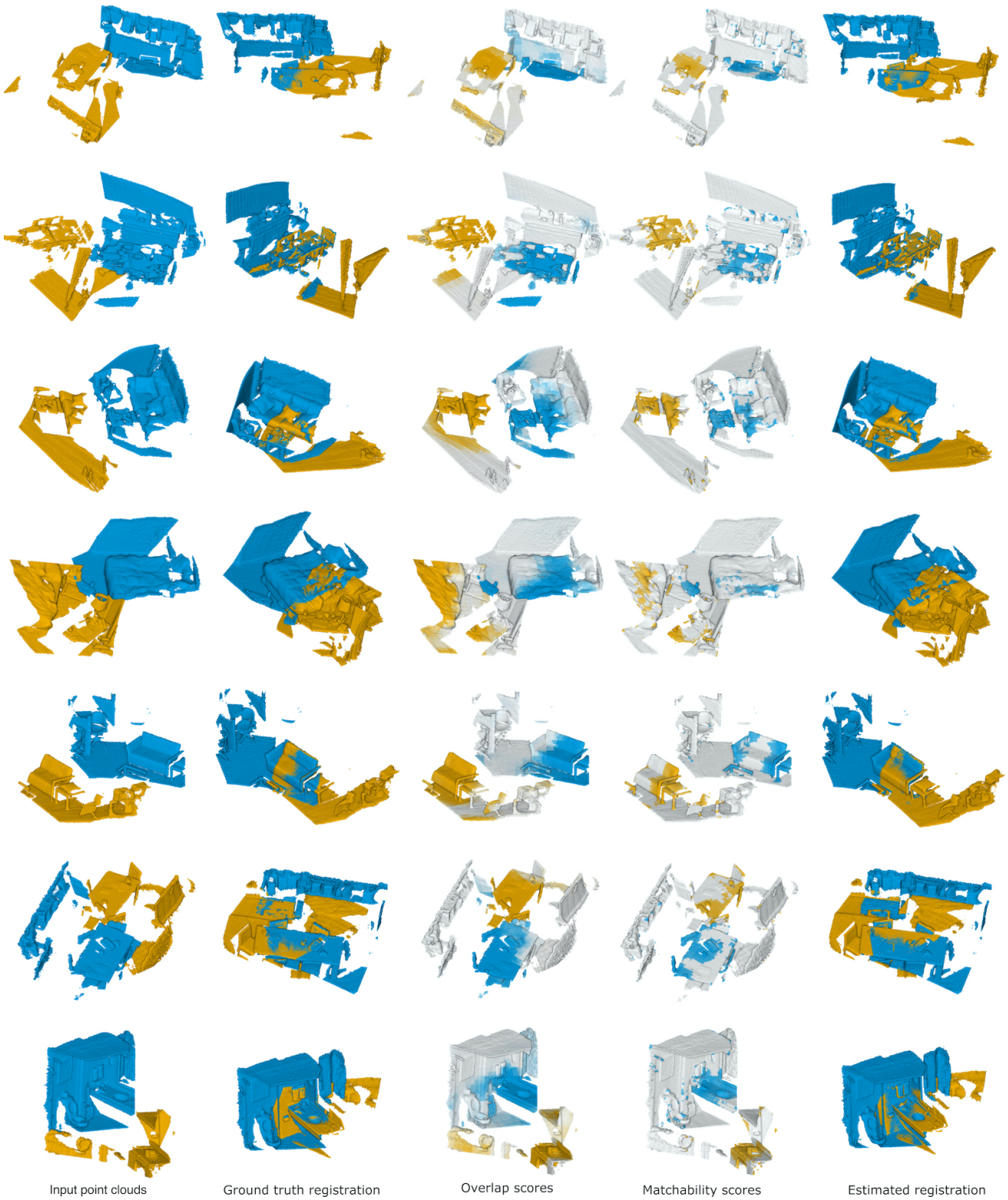

Figure 12: Example results on *3DLoMatch*.

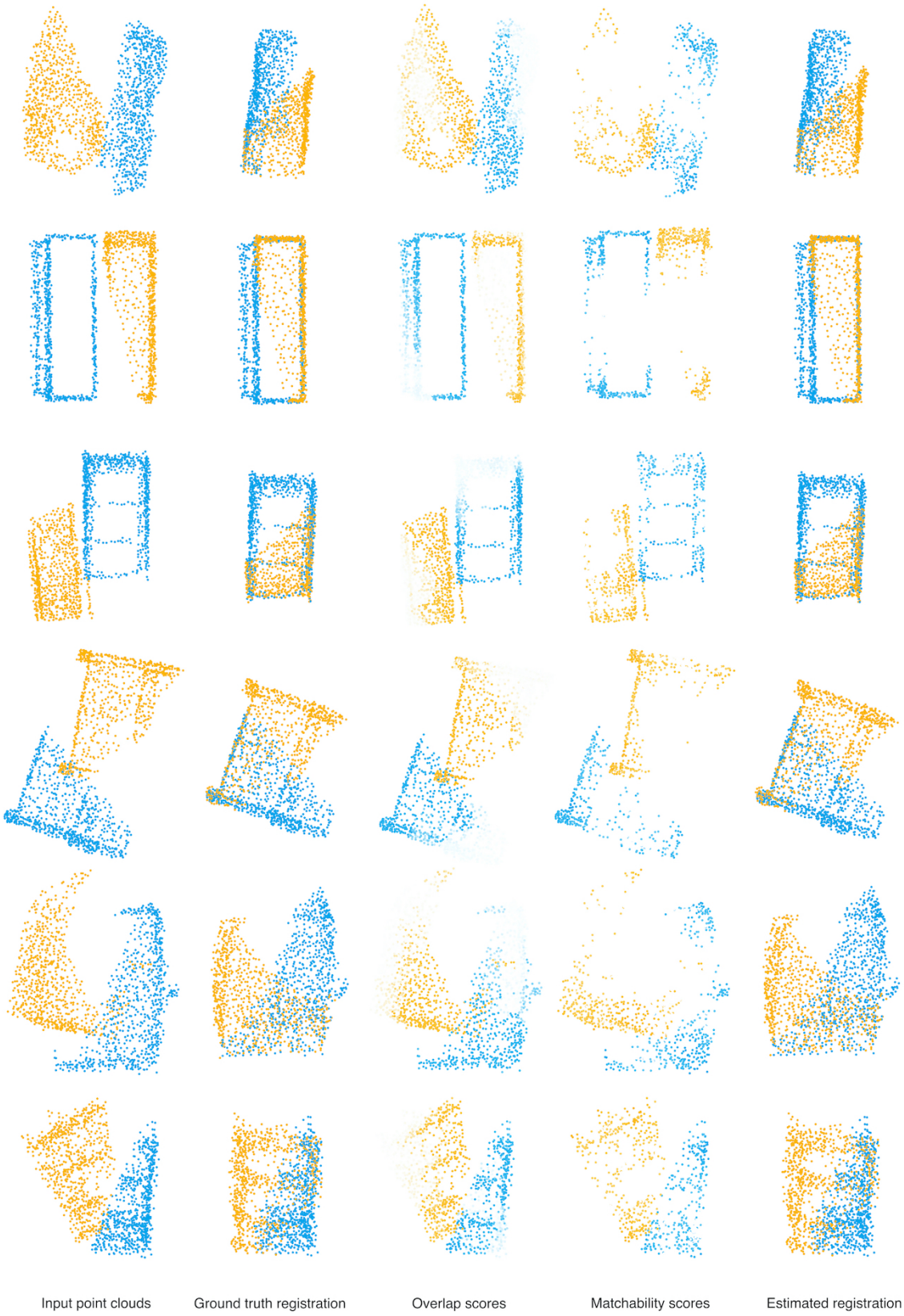

Figure 13: Example results on *ModelLoNet*.