PPF-FoldNet: Unsupervised Learning of Rotation Invariant 3D Local Descriptors

Haowen Deng [†✣*] Tolga Birdal[0000−0001−7915−7964] [†✣] Slobodan Ilic [†✣]

[†] Technische Universitat München, Germany [✣] Siemens AG, Germany
[*] National University of Defense Technology, China

Abstract. We present PPF-FoldNet for unsupervised learning of 3D local descriptors on pure point cloud geometry. Based on the folding-based auto-encoding of well known point pair features, PPF-FoldNet offers many desirable properties: it necessitates neither supervision, nor a sensitive local reference frame, benefits from point-set sparsity, is end-to-end, fast, and can extract powerful rotation invariant descriptors. Thanks to a novel feature visualization, its evolution can be monitored to provide interpretable insights. Our extensive experiments demonstrate that despite having six degree-of-freedom invariance and lack of training labels, our network achieves state of the art results in standard benchmark datasets and outperforms its competitors when rotations and varying point densities are present. PPF-FoldNet achieves 9% higher recall on standard benchmarks, 23% higher recall when rotations are introduced into the same datasets and finally, a margin of $> 35\%$ is attained when point density is significantly decreased.

Keywords: 3D deep learning, local features, descriptors, rotation invariance

1 Introduction

Local descriptors are one of the essential tools used in computer vision, easing the tasks of object detection, pose estimation, SLAM or image retrieval [23, 27]. While being well established in the 2D domain, 3D local features are still known to lack good discriminative power and repeatability. With the advent of deep learning, many areas in computer vision shifted from hand crafted labor towards a problem specific end-to-end learning. Local features are of course no exception. Already in 2D, learned descriptors significantly outperform their engineered counterparts [49, 28]. Thus, it was only natural for the scholars to tackle the task of 3D local feature extraction employing similar approaches [18, 51, 8]. However, due to the inherent ambiguities and less informative nature of sole geometry, extracting 3D descriptors on point sets still poses an unsolved problem, even for learning-based methods.

Up until now, deep learning of local features in 3D has suffered from one or more of the following: **a**) being supervised and requiring an abundant amount of labels in form of pairs, triplets or N-tuples [51, 8], **b**) being sensitive to 6DoF rotations [51, 8], **c**) involving significant hand-crafted input preparation [18] and **d**) unsatisfactory performance [18, 30]. In this paper, we map out an elegant architecture to tackle all of these problems and present PPF-FoldNet: an unsupervised, high-accuracy, 6DoF transformation invariant, sparse and fast 3D local feature learning network. PPF-FoldNet operates

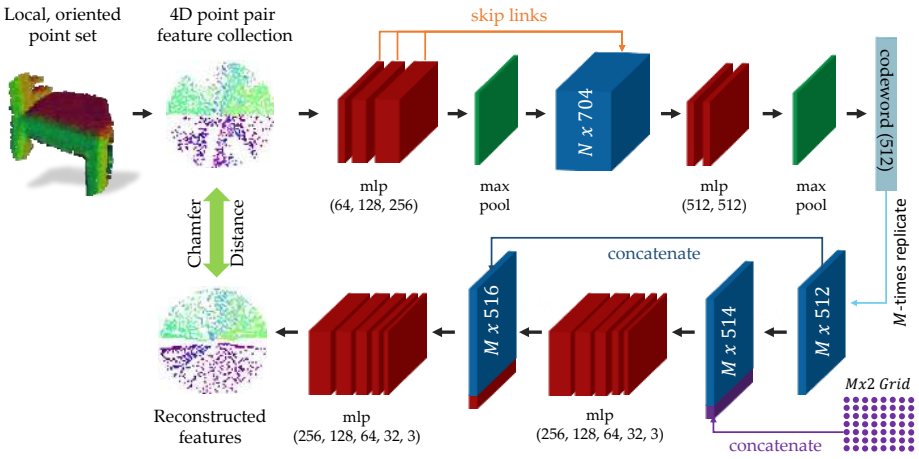


Fig. 1. PPF-FoldNet: The point pair feature folding network. The point cloud local patches are first converted into PPF representations, and then sent into the encoder to get compressed codewords. The decoder tries to reconstruct full PPFs from these codewords by folding. This forces the codewords to keep the most critical and discriminative information. The learned codewords are proven to be robust and effective as we will show across extensive evaluations.

directly on point sets, taking into account the point sparsity and permutation invariant set property, deals well with density variations, while significantly outperforming its rotation-variant counterparts even based on the standard benchmarks.

Our network establishes theoretical rotation invariance inspired by use a point pair feature (PPF) [4, 3, 8] encoding of the local 3D geometry into patches. In contrast to PPFNet [8], we do not incorporate the original points or normals into the encoding. The collection of these 4D PPFs are then sent to a FoldingNet-like end to end auto-encoder (AE) [48], trained to auto-reconstruct the PPFs, using a set distance. Our encoder is simpler than in FoldingNet and for decoding, we propose a similar folding scheme, where a low dimensional 2D grid lattice is folded onto a 4D PPF space and monitor the network evolution by a novel lossless visualization of the PPF space. Our overall architecture is based on PointNet [30] to achieve permutation invariance and to fully utilize the sparsity. Training our AE is far easier than training, for example, 3DMatch [51], because we do not need to sample pairs or triplets from a pre-annotated large dataset and we benefit from linear time complexity to the number of patches.

Extensive evaluations demonstrate that PPF-FoldNet outperforms the state of the art across the standard benchmarks in which severe rotations are avoided. When arbitrary rotations are introduced into the input, our descriptors outperform related approaches by a large margin including even the best competitor, Khoury et al.'s CGF [18]. Moreover, we report better performance as the input sparsifies, as well as good generalization properties. Our qualitative evaluations will uncover how our network operates and give valuable interpretations. In a nutshell, our contributions can be summarized as:

– An auto-encoder, that unifies a PointNet encoding with a FoldingNet decoder,

- Use of well established 4D PPFs in this modified auto-encoder to learn rotation invariant 3D local features without supervision.
- A novel look at the invariance of point pair features and derived from it, a new way of visualizing PPFs and monitoring the network progress.

2 Prior Art

Following their hand-crafted counterparts [35, 34, 40, 16, 10], 3D deep learning methods started to enjoy a deep-rooted history. Initial attempts to learn from 3D data used the naive dense voxel grid representation [51, 46, 26, 11]. While being straightforward extensions of 2D architectures, such networks did not perform as efficiently and robustly as 2D CNNs [21]. Hence, they are superseded by networks taking into account the spatial sparsity by replacing the dense grids with octrees [33, 38, 43] or kd-trees [20].

Another family of works acknowledges that 3D surfaces live on 2D submanifolds and seek to learn projections rather than the space of actual input. A reduction of dimension to two makes it possible to benefit from developments in 2D CNNs such as Res-Nets [13]: LORAX [9] proposes a *super-point* to depth map projection. Kehl et al. [17] operate on the RGB-D patches that are natural projections onto the camera plane. Huang et al. [15] anchor three local cameras to each 3D keypoint and collect multi-channel projections to learn a semi-global representation. Cao et al. [7] use spherical projections to aid object classification. Tatarchenko et al. propose convolutions in the tangent space as a way of operating on the local 2D projection [39].

Point clouds can be treated as graphs by associating edges among neighbors. This paves the way to the appliance of graph convolutional networks [25]. FoldingNet [48] employs graph-based encoding layers. Wang et al. [44] tackle the segmentation tasks on point sets via graph convolutions networks (GCNs), while Qi et al. [32] apply GCNs to RGB-D semantic segmentation. While showing a promising direction, the current efforts involving graphs on 3D tasks are still supervised, try to imitate CNNs and cannot really outperform their unstructured point-processing counterparts.

Despite all developments in 3D deep learning, there are only a handful of methods that explicitly learn generic local descriptors on 3D data. One of the first methods that learns 3D feature matching, also known as correspondence, is 3DMatch [51]. It uses dense voxel grids to summarize the local geometry and learning is performed via contrastive loss. 3DMatch is weakly supervised by task, does not learn generic descriptors, and is not invariant to rotations. PointNet [30] and PointNet++ [31] work directly on the unstructured point clouds and minimize a multi-task loss, resulting in local and global features. Similar to [51], invariance is not of concern and weak supervision is essential. CGF [18] combines a hand-crafted input preparation with a deep dimensionality-reduction and still uses supervision. However, the input features are not learned but only the embedding. PPFNet [8] improves over all these methods by incorporating global context, but still fails to achieve full invariance and expects supervision.

2.1 Background

From all of the aforementioned developments, we will now pay particular attention to three: PointNet, FoldingNet and PPFNet which combined, give our network its name.

PointNet [30] Direct consumption of unstructured point input in the form of a set within deep networks began by PointNet. Qi et al. proposed to use a point-wise multi layer perceptron (MLP) and aggregated individual feature maps into a global feature by a permutation-invariant max pooling. Irrespective of the input ordering, PointNet can generate per-point local descriptors as well as a global one, which can be combined to solve different problems such as keypoint extraction, 3D segmentation or classification. While not being the most powerful network, it clearly sets out a successful architecture giving rise to many successive studies [31, 29, 2, 36].

FoldingNet [48] While PointNet can work with point clouds, it is still a supervised architecture, and constructing unsupervised extensions like an auto-encoder on points is non-trivial as the upsampling step is required to interpolate sets [50, 31]. Yang et al. offer a different perspective and instead of resorting to costly voxelizations [45], propose *folding*, as a strong decoder alternative. Folding warps an underlying low-dimensional grid towards a desired set, specifically a 3D point cloud. Compared to other unsupervised methods, including GANs [45], FoldingNet achieves superior performance in common tasks such as classification and therefore, in PPF-FoldNet we benefit from its decoder structure, though in a slightly altered form.

PPFNet [8] proposes to learn local features informed by the global context of the scene. To do so, an N-tuple loss is designed, seeking to find correspondences jointly between all patches of two fragments. Features learned in this way are shown to be superior than prior methods and PPFNet is reported to be the state-of-the-art local feature descriptor. However, even if Deng et al. stress the importance of learning permutation and rotation invariant features, the authors only manage to improve the resilience to Euclidean isometries slightly by concatenating PPF to the point set. Moreover, the proposed N-tuple loss still requires supervision. Our work improves on both of these aspects: It is capable of using PPFs only and operating without supervision.

3 PPF-FoldNet

PPF-FoldNet is based on the idea of auto-encoding a rotation invariant but powerful representation of the point set (PPFs), such that the learned low dimensional embedding can be truly invariant. This is different to training the network with many possible rotations of the same input and forcing the output to be a canonical reconstruction. The latter would both be approximate and much harder to learn. Input to our network are local patches which, unlike PPFNet, are individually auto-encoded. The latent low dimensional vector of the auto-encoder, *codeword*, is used as the local descriptor attributed to the point around which the patch is extracted.

3.1 Local Patch Representation

Our input point cloud is a set of oriented points $\mathbf{X} = \{\mathbf{x}_i \in \mathbb{R}^6\}$, meaning that each point is decorated with a local normal (e.g. tangent space) $\mathbf{n} \in \mathbb{R}^3$: $\mathbf{x} = \{\mathbf{p}, \mathbf{n}\} \in \mathbb{R}^6$. A local patch is a subset of the input $\mathbf{\Omega}_{\mathbf{x}_r} \subset \mathbf{X}$ center around a reference point $\mathbf{x}_r$.

We then encode this patch as a collection of pair features, computed between a central reference and all the other points:

$$\mathbf{F}_{\mathbf{\Omega}} = \{\, \mathbf{f}(\mathbf{x}_r, \mathbf{x}_1) \cdots \mathbf{f}(\mathbf{x}_r, \mathbf{x}_i) \cdots \mathbf{f}(\mathbf{x}_r, \mathbf{x}_N) \,\} \in \mathbb{R}^{4 \times N-1},\ i \neq r \tag{1}$$

The features between any pair (point pair features) are then defined to be a map $\mathbf{f} : \mathbb{R}^{12} \rightarrow \mathbb{R}^4$ sending two oriented points to three angles and the pair distance:

$$\mathbf{f} : (\mathbf{x}_r^T, \mathbf{x}_i^T)^T \rightarrow (\angle(\mathbf{n}_r, \mathbf{d}), \angle(\mathbf{n}_i, \mathbf{d}), \angle(\mathbf{n}_r, \mathbf{n}_i), \|\mathbf{d}\|_2)^T \tag{2}$$

$\mathbf{d} = \mathbf{p}_r - \mathbf{p}_i$. An angle computation for non-normalized vectors is given in [3]. Such encoding of the local geometry resembles that of PPFNet [8], but differs in the fact that we ignore the points and normals as they are dependent on the orientation and local reference frame. We instead use pure point pair features, thereby avoiding a canonical frame computation. Note that the dimensionality of this feature is still irreducible without data loss.

Proposition 1. *PPF representation* $\mathbf{f}$ *around* $\mathbf{x}_r$ *explains the original oriented point pair up to a rotation and reflection about the normal of the reference point.*

Proof. Let us consider two oriented points $\mathbf{x}_1$ and $\mathbf{x}_2$. We can always write the components of the associated point pair feature $\mathbf{f}(\mathbf{x}_1, \mathbf{x}_2)$ as follows:

$$\mathbf{n}_1^T \mathbf{n}_2 = f_1 \qquad \mathbf{n}_1^T \mathbf{d}_n = f_2 \qquad \mathbf{n}_2^T \mathbf{d}_n = f_3 \tag{3}$$

where $\mathbf{d}_n = \mathbf{d}/\|\mathbf{d}\|$. We now try to recover the original pair given its features. First, it is possible to write:

$$\begin{bmatrix} \mathbf{n}_1^T \\ \mathbf{n}_2^T \\ \mathbf{d}_n^T \end{bmatrix} \begin{bmatrix} \mathbf{n}_1 & \mathbf{n}_2 & \mathbf{d}_n \end{bmatrix} = \begin{bmatrix} 1 & f_1 & f_2 \\ f_1 & 1 & f_3 \\ f_2 & f_3 & 1 \end{bmatrix} \tag{4}$$

given that all vectors are of unit length. In matrix notation, Eq. 4 can be written as $\mathbf{A}^T\mathbf{A} = \mathbf{K}$. Then, by singular value decomposition, $\mathbf{K} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ and thus $\mathbf{A} = \mathbf{U}\mathbf{S}^{1/2}\mathbf{V}^T$. Note that, any orthogonal matrix (rotation and reflection) $\mathbf{R}$ can now be applied to $\mathbf{A}$ without changing the outcome: $(\mathbf{R}\mathbf{A})^T\mathbf{R}\mathbf{A} = \mathbf{A}^T\mathbf{R}^T\mathbf{R}\mathbf{A} = \mathbf{A}^T\mathbf{A} = \mathbf{K}$. Hence, such decomposition is up to finite-dimensional linear isometries: rotations and reflections. Since we know that the local patch is centered at the reference point $\mathbf{p}_r = \mathbf{0}$, we are free to choose an $\mathbf{R}$ such that the normal vector of $\mathbf{p}_r$ ($\mathbf{n}_r$) is aligned along one of the canonical axes, say $+\mathbf{z} = [0, 0, 1]^T$ (freely chosen):

$$\mathbf{R} = \mathbf{I} + [\mathbf{v}]_x + [\mathbf{v}_x]^2 \frac{1 - n_r^z}{\|v\|} \tag{5}$$

where $\mathbf{v} = \mathbf{n}_r \times \mathbf{z}$, n_r^z is the z component of $\mathbf{n}_r$ and $\mathbf{I}$ is identity. $[\cdot]_x$ denotes skew symmetric cross product matrix. Because now $\mathbf{R}\mathbf{n}_r = \mathbf{z}$, any rotation θ and reflection ϕ about $\mathbf{z}$ would result in the same vector $\mathbf{z} = \mathbf{R}_z(\theta, \phi)\mathbf{z}$, $\forall \theta, \phi \in \mathbb{R}$. Any paired point can then be found in the canonical frame, uniquely up to two parameters as $\mathbf{p}_r \leftarrow \|\mathbf{d}\|\mathbf{R}_z(\theta.\phi)\mathbf{R}\mathbf{d}_n$, $\mathbf{n}_r \leftarrow \mathbf{R}_z(\theta, \phi)\mathbf{R}\mathbf{n}_r$. □

In the case where reflections are ignored (as they are unlikely to happen in a 3D world), this leaves a single degree of freedom, rotation angle around the normal. Also note once again that for the given local representation, the reference point $\mathbf{p}_r$ is common to all the point pairs.

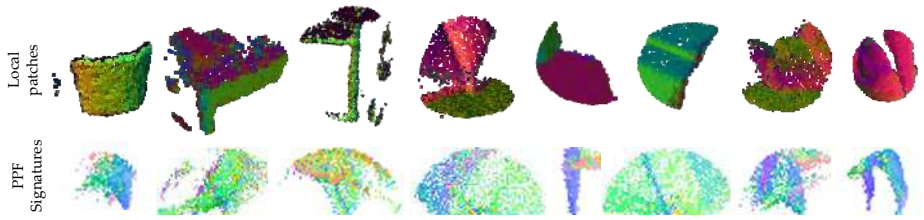


Fig. 2. Visualisation of some local patches and their correspondent PPF Signatures.

Visualizing PPFs PPFs exist in a 4D space and thus it is not trivial to visualize them. While simple solutions such as PCA would work, we prefer a more geometrically meaningful and simpler solution. Proposition 1 allows us to compute a signature of a set of point pairs by orienting the vectors $(\mathbf{n}_1, \mathbf{n}_2, \mathbf{d})$ individually for all points in order to align the difference vectors $\{\mathbf{d}_i\}$ with the $x-z$ plane by choosing an appropriate $\mathbf{R}_z(\theta.\phi)$. Such a transformation would not alter the features as shown. In this way, the paired points can be transformed onto a common plane (image), where the location is determined by the difference vector, in polar coordinates. The normal of the second point would not lie in this plane but can be encoded as colors in that image. Hence, it is possible to obtain a 2D visualization, without any data loss, i.e. all components of the vector contribute to the visualization. In Fig. 2 we provide a variety of local patch and PPF visualizations from the datasets of concern.

3.2 PPF Auto-Encoder and Folding

PPF-FoldNet employs a PointNet-like encoder with skip-links and a FoldingNet-like decoding scheme. It is designed to operate on 4D-PPFs, as summarized in Fig. 1.

Encoder The input to our network, and thus to the encoder, is $\mathbf{F}_{\mathbf{\Omega}}$, a local PPF representation, as in §3.1. A three-layer, point-wise MLP (Multi Layer Perceptron) follows the input layer and subsequently a max-pooling is performed to aggregate the individual features into a global one, similar to PointNet [30]. The low level features are then concatenated with this global feature using skip-links. This results in a more powerful representation. Another two-layer MLP finally redirects these features to a final encoding, the codeword, which is of dimension 512.

Proposition 2. *The encoder structure of PPF-FoldNet is permutation invariant.*

Sketch of the proof. The encoder is composed of per-data-point functions (MLP), RELU layers and max-pooling, all of which either do not affect the point order or are individually shown to be permutation invariant [30, 48]. Moreover, it is shown that composition of functions is also invariant [48] and so is our encoder. We refer the reader to the references for further details. □

In summary, altering the order of the PPF set will not affect the learned representation.

Decoder Our decoder tries to reconstruct the whole set of point PPFs using a single codeword, which in return, also forces the codeword to be informative and distill the most distinctive information from the high-dimensional input space. However, inspired by FoldingNet, instead of trying to upsample or interpolate point sets, the decoder will try to deform a low-dimensional grid structure guided by the codeword. Each grid point is concatenated to a replica of the codeword, resulting in an $M \times 514$ vector as input to what is referred as *folding operation* [48]. Folding can be a highly non-linear operation and is thus performed by two consecutive MLPs: the first folding results in a deformed grid, which is appended once again to the codewords and propagates through the second MLP, reconstructing the input PPFs. Moreover, in contrast to FoldingNet [48], we try to reconstruct a higher dimensional set, 4D vs 3D (2D manifold); we are better off using a deeper MLP - 5-layer as opposed to the 3-layer of [48].

Other than simplifying and strengthening the decoding, the folding is also beneficial in making the network interpretable. For instance, it is possible to monitor the grid during subsequent iterations and envisage how the network evolves. To do so, §4.4 will trace the PPF sets by visualizing them as described in §3.1.

Chamfer Loss Note that as size of the grid M, is not necessarily the same as the size of the input N, and the correspondences in 4D PPF space are lost when it comes to evaluating the loss. This requires a distance computation between two unequal cardinality point pair feature sets, which we measure via the well known Chamfer metric:

$$d(\mathbf{F}, \hat{\mathbf{F}}) = \max\left\{ \frac{1}{|\mathbf{F}|} \sum_{\mathbf{f} \in \mathbf{F}} \min_{\hat{\mathbf{f}} \in \hat{\mathbf{F}}} \|\mathbf{f} - \hat{\mathbf{f}}\|_2,\ \frac{1}{|\hat{\mathbf{F}}|} \sum_{\mathbf{f} \in \hat{\mathbf{F}}} \min_{\mathbf{f} \in \mathbf{F}} \|\mathbf{f} - \hat{\mathbf{f}}\|_2 \right\} \tag{6}$$

where $\hat{}$ operator refers to the reconstructed (estimated) set.

Implementation details PPF-FoldNet uses Tensorflow framework [1]. The initial values of all variables are initialized randomly by Xavier's algorithm. Global loss is minimized with an ADAM optimizer [19]. Learning rate starts at 0.001 and exponentially decays after every 10 epochs, truncated at 0.0001. We use batches of size 32.

4 Experimental Evaluation

4.1 Datasets and Preprocessing

To fully drive the network towards learning varieties of local 3D geometries and gain robustness to different noises present in real data, we use the 3DMatch Benchmark Dataset [51]. This dataset is a large ensemble of the existing ones such as Analysis-by-Synthesis [41], 7-Scenes [37], SUN3D [47], RGB-D Scenes v.2 [22] and Halber and Funkhouser [12]. It contains 62 scenes in total, and we reserve 54 of them for training and validation. 8 are for benchmarking. 3DMatch already provided fragments fused from 50 consecutive depth frames of the 8 test scenes, and we follow the same pipeline to generate fragments from the training scenes. Test fragments lack the color information and therefore we resort to using only the 3D shape. This also makes our network insensitive to illumination changes.

Table 1. Our results on the standard 3DMatch benchmark. *Red Kitchen* data is from 7-scenes [37] and the rest imported from SUN3D [47].

	Spin Image [16]	SHOT [35]	FPFH [34]	3DMatch [51]	CGF [18]	PPFNet [8]	FoldNet [48]	Ours	Ours-5K
Kitchen	0.1937	0.1779	0.3063	0.5751	0.4605	**0.8972**	0.5949	0.7352	0.7866
Home 1	0.3974	0.3718	0.5833	0.7372	0.6154	0.5577	0.7179	0.7564	**0.7628**
Home 2	0.3654	0.3365	0.4663	**0.7067**	0.5625	0.5913	0.6058	0.625	0.6154
Hotel 1	0.1814	0.208	0.2611	0.5708	0.4469	0.5796	0.6549	0.6593	**0.6814**
Hotel 2	0.2019	0.2212	0.3269	0.4423	0.3846	0.5769	0.4231	0.6058	**0.7115**
Hotel 3	0.3148	0.3889	0.5000	0.6296	0.5926	0.6111	0.6111	0.8889	**0.9444**
Study	0.0548	0.0719	0.1541	0.5616	0.4075	0.5342	**0.7123**	0.5753	0.6199
MIT Lab	0.1039	0.1299	0.2727	0.5455	0.3506	**0.6364**	0.5844	0.5974	0.6234
Average	0.2267	0.2382	0.3589	0.5961	0.4776	0.6231	0.6130	0.6804	**0.7182**

Prior to operation, we downsample the fused fragments with spatial uniformity [5] and compute surface normals using [14] in a 17-point neighborhood. A reference point and its neighbors within 30 cm vicinity form a local patch. The number of points in a local patch is thus flexible, which makes it difficult to organize data into regular batches. To facilitate training as well as to increase the representation robustness to noises and different point densities, each local patch is down-sampled. For a fair comparison with other methods in the literature, we use 2048 points, but also provide an extended version that uses 5K since we are not memory bound, as for example, PPFNet [8] is. The preparation stage ends with the PPFs calculated for the assembled local patches.

4.2 Accuracy Assessment Techniques

Let us assume that a pair of fragments $\mathbf{P} = \{\mathbf{p}_i \in \mathbb{R}^3\}$ and $\mathbf{Q} = \{\mathbf{q}_i \in \mathbb{R}^3\}$ are aligned by an associated rigid transformation $\mathbf{T} \in SE(3)$, resulting in a certain overlap. We then define a non-linear feature function $g(\cdot)$ for mapping from input points to feature space, and in our case, this summarizes the PPF computation and encoding as a codeword. The feature for point $\mathbf{p}_i$ is $g(\mathbf{p}_i)$, and $g(\mathbf{P})$ is the pool of features extracted for the points in $\mathbf{P}$. To estimate the rigid transformation between $\mathbf{P}$ and $\mathbf{Q}$, the typical approach finds a set of matching pairs in each fragment and associates the correspondences. The inter point pair set $\mathbf{M}$ is formed by the pairs $(\mathbf{p}, \mathbf{q})$ that lie mutually close in the feature space by applying nearest neighbor search NN:

$$\mathbf{M} = \{\{\mathbf{p}_i, \mathbf{q}_i\}, g(\mathbf{p}_i) = NN(g(\mathbf{q}_i), g(\mathbf{P})),\ g(\mathbf{q}_i) = NN(g(\mathbf{p}_i), g(\mathbf{Q}))\} \quad (7)$$

True matches set $\mathbf{M}_{gnd}$ is the set of point pairs with a Euclidean distance below a threshold τ_1 under ground-truth transformation $\mathbf{T}$.

$$\mathbf{M}_{gnd} = \{\{\mathbf{p}_i, \mathbf{q}_i\} : (\mathbf{p}_i, \mathbf{q}_i) \in \mathbf{M}, ||\mathbf{p}_i - \mathbf{T}\mathbf{q}_i||_2 < \tau_1\} \quad (8)$$

We now define an inlier ratio for $\mathbf{M}$ as the percentage of true matches in $\mathbf{M}$ as $r_{in} = |\mathbf{M}_{gnd}|/|\mathbf{M}|$. To successfully estimate the rigid transformation based on $\mathbf{M}$ via registration algorithms, r_{in} needs to be greater than τ_2. For example, in a common RANSAC pipeline, achieving 99.9% confidence in the task of finding a subset with at least 3 correct matches $\mathbf{M}$, with an inlier ratio $\tau_2 = 5\%$ requires at least 55258 iterations. Theoretically, given $r_{in} > \tau_2$, it is highly probable a reliable local registration algorithm

Table 2. Our results on the rotated 3DMatch benchmark. *Red Kitchen* data is from 7-scenes [37] and the rest imported from SUN3D [47].

	Spin Image [16]	SHOT [35]	FPFH [34]	3DMatch [51]	CGF [18]	PPFNet [8]	FoldNet [48]	Ours	Ours-5K
Kitchen	0.1779	0.1779	0.2905	0.004	0.4466	0.002	0.0178	0.7352	**0.7885**
Home 1	0.4487	0.3526	0.5897	0.0128	0.6667	0.0000	0.0321	0.7692	**0.7821**
Home 2	0.3413	0.3365	0.4712	0.0337	0.5288	0.0144	0.0337	0.6202	**0.6442**
Hotel 1	0.1814	0.2168	0.3009	0.0044	0.4425	0.0044	0.0133	0.6637	**0.6770**
Hotel 2	0.1731	0.2404	0.2981	0.0000	0.4423	0.0000	0.0096	0.6058	**0.6923**
Hotel 3	0.3148	0.3333	0.5185	0.0096	0.6296	0.0000	0.0370	0.9259	**0.963**
Study	0.0582	0.0822	0.1575	0.0000	0.4178	0.0000	0.0171	0.5616	**0.6267**
MIT Lab	0.1169	0.1299	0.2857	0.026	0.4156	0.0000	0.0260	0.6104	**0.6753**
Average	0.2265	0.2337	0.364	0.0113	0.4987	0.0026	0.0233	0.6865	**0.7311**

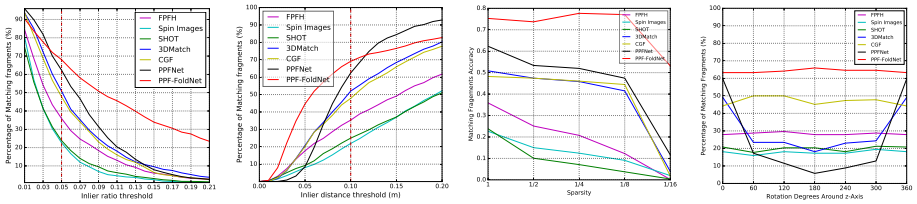
would work, regardless of the robustifier. Therefore instead of using the local registration results to judge the quality of features, which would be both slow and not very straightforward, we define $\mathbf{M}$ with $r_{in} > \tau_2$ votes for a correct match of two fragments.

Each scene in the benchmark contains a set of fragments. Fragment pairs $\mathbf{P}$ and $\mathbf{Q}$ having an overlap above 30% under the ground-truth alignment are considered to match. Together they form the set of fragment pairs $\mathbf{S} = \{(\mathbf{P}, \mathbf{Q})\}$ that are used in evaluations. The quality of features is measured by the recall R of fragment pairs matched in $\mathbf{S}$:

$$R = \frac{1}{|\mathbf{S}|} \sum_{i=1}^{|\mathbf{S}|} \mathbb{1}\left(r_{in}\big(\mathbf{S}_i = (\mathbf{P}_i, \mathbf{Q}_i)\big) > \tau_2 \right) \tag{9}$$

4.3 Results

Feature quality evaluation We first compare the performance of our features against the well-accepted works on the 3DMatch benchmark with $\tau_1 = 10$ cm and $\tau_2 = 5\%$. Tab. 1 tabulates the findings. The methods selected for comparison comprise 3 handcrafted features (Spin Images [16], SHOT [35], FPFH [34]) and 4 state-of-the-art deep learning based methods (3DMatch [51], CGF [18], PPFNet [8], FoldingNet [48]). Note that FoldingNet has never been tested on local descriptor extraction before. It is apparent that, overall, our PPF-FoldNet could match far more fragment pairs in comparison to the other methods, except for scenes *Kitchen* and *Home*, where PPFNet and 3DMatch achieve a higher recall respectively. In all the other cases, PPF-FoldNet outperforms the state of the art by a large margin, $> 9\%$ on average. PPF-FoldNet has a recall of 68.04% when using 2K sample points (the same as PPFNet), while PPFNet remains on 62.32%. Moreover, because PPF-FoldNet has no memory bottleneck, it can achieve an additonal 3% improvement in comparison with the 2K version, when 5K points are used. Interestingly, FPFH is also constructed from a type of PPF features [34], but in a form of manual histogram summarization. Compared to FPFH, PPF-FoldNet has 32.15% and 35.93% higher recall using 2K and 5K points respectively. It demonstrates the unprecedented strength of our advanced method in compressing the PPFs. In order to optimally reconstruct PPFs in the decoder, the network forces the bottleneck codeword to be compact as well as distilling the most critical and distinctive information in PPFs.



(a) (b) (c) (d)

Fig. 3. Evaluations on 3DMatch benchmark: **(a)** Results of different methods under varying inlier ratio threshold **(b)** Results of different methods under varying point distance threshold **(c)** Evaluating robustness again point density **(d)** Evaluations against rotations around z-axis

To illustrate that parameters in the evaluation metric are not tuned for our own good, we also repeat the experiments with different τ_1 and τ_2 values. The results are shown in Fig. 3(a) and Fig. 3(b). In Fig. 3(a), τ_1 is fixed at 10 cm, τ_2 increases gradually from 1% to 20%. When τ_2 is above 4%, PPF-FoldNet always has a higher recall than the other methods. Below 4%, some other methods may obtain a higher recall but this is too strict for most of the registration algorithms anyway. It is further noteworthy that when τ_2 is set to 20%, the point where PPF-FoldNet still gets a recall above 20%, the performance of the other methods falls below 5%. This justifies that PPF-FoldNet is capable of generating many more sets of matching points with a high inlier ratio r_{in}. This offers a tremendous benefit for the registration algorithms. In Fig. 3(b), τ_2 is fixed at 5%, τ_1 increases gradually from 0 cm to 20 cm. When τ_1 is smaller than 12 cm, PPF-FoldNet consistently generates higher recall. This finding indicates that PPF-FoldNet matches more point pairs with a small distance error in the Euclidean space, which could efficiently decrease the rigid transformation estimation errors.

Tests on rotation invariance To demonstrate the outstanding rotation-invariance property of PPF-FoldNet, we take random fragments out of the evaluation set and gradually rotate them around the z-axis from 60° to 360° in steps of 60°. The matching results are shown in Fig. 3(d). As expected, both PPFNet and 3DMatch perform poorly as they operate on rotation-variant input representations. Hand crafted features or CGF also demonstrate robustness to rotations thanks to the reliance on the local reference frame (LRF). However, PPF-FoldNet stands out as the best approach with a much higher recall which furthermore does not require computation of local reference frames.

To further test how those methods perform under situations with severe rotations, we rotate all the fragments in 3DMatch benchmark with randomly sampled axes and angles over the whole rotation space, and introduce a new benchmark – *Rotated 3DMatch Benchmark*. The same evaluation is once again conducted on this new benchmark. Keeping the accuracy evaluations identical, our results are shown in Tab. 2. 3DMatch and PPFNet completely failed under this new benchmark because of the variables introduced by large rotations. Once again, PPF-FoldNet, surpasses all other methods, achieving the best results in all the scenes, predominates the runner-up CGF by large margins of 18.78% and 23.24% respectively when using 2K and 5K points.

Table 3. Accuracy comparison of different PPF representations.

	Kitchen	Home 1	Home 2	Hotel 1	Hotel 2	Hotel 3	Study	MIT Lab	Average
PPFH	0.534	0.622	0.486	0.341	0.346	0.574	0.233	0.351	0.436
Bobkov1	0.514	0.635	0.510	0.403	0.433	0.611	0.281	0.481	0.483
Our-PPF	0.506	0.635	0.495	0.350	0.385	0.667	0.267	0.403	0.463

Sparsity evaluation Thanks to the sparse representation of our input, PPF-FoldNet is also robust in respect of the changes in point cloud density and noise. Fig.3(c) shows the performance of different methods when we gradually decrease the points in the fragment from 100% to only 6.25%. We can see that PPF-FoldNet is least affected by the decrease in point cloud density. In particular, when only 6.25% points are left in the fragments, the recall for PPF-FoldNet is still greater than 50% while PPFNet remains around 12% and the other methods almost fail. The results of PPFNet and PPF-FoldNet together demonstrate that PPF representation offers more robustness in respect of point densities, which is a common problem existing in many point cloud representations.

Can PPF-FoldNet operate with different PPF constructions? We now study 3 identical networks, trained for 3 different PPF formulations: ours, PPFH (the PPF used in FPFH [34]) and Bobkov1 et al. [6]. The latter has an added component of *occupancy ratio* based on grid space. We use a subset of 3DMatch benchmark to train all networks for a fixed number of iterations and test on the rotated fragments. Tab. 3 presents our findings: all features perform similarly. Thus, we do not claim the superiority of our PPF representation, but stress that it is simple, easy to compute, intuitive and easy to visualize. Due to the voxelization, *Bobkov1* is significantly slower than the other methods, and due to the lack of an LRF, our PPF is faster than *PPFH*'s. Using stronger pair primitives would favor PPF-FoldNet as our network is agnostic to the PPF construction.

Runtime We run our algorithm on a machine loaded with NVIDIA TitanX Pascal GPU and an Intel Core i7 3.2GHz CPU. On this hardware, computing features of an entire fragment via FPFH [34] takes 31.678 seconds, whereas PPF-FoldNet achieves a $10\times$ speed-up with 3.969 seconds, despite having similar theoretical complexity. In particular, our input preparation for PPF extraction runs in 2.616 seconds, and the inference in 1.353. This is due to 1) PPF-FoldNet requiring only a single pass over the input, 2) our efficient network accelerated on GPU powered Tensorflow.

4.4 Qualitative Evaluations

Visualizing the matching result From the quantitative results, PPF-FoldNet is expected to have better and more correct feature matches, especially when arbitrary rigid transformations are applied. To show this visually, we run different methods and ours across several fragments undergoing varying rotations. In Fig. 4 we show the matching regions, over uniformly sampled [5] keypoints on these fragments. It is clear that our algorithm performs the best among all others in discovering the most correct correspondences.

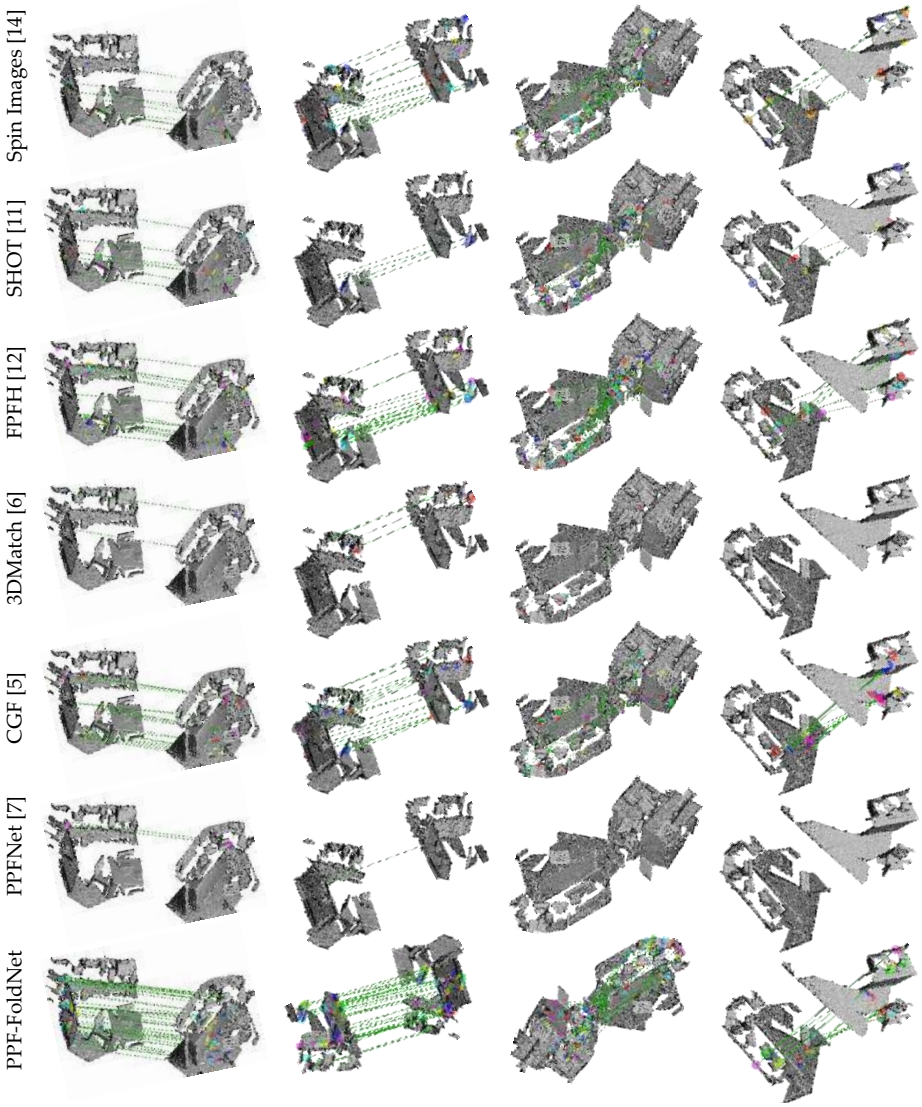


Fig. 4. Qualitative results of matching across different fragments and for different methods. When severe transformations are present, only hand-crafted algorithms, CGF and our method achieves satisfactory matches. However, for PPF-FoldNet, the number of matches are significantly larger.

Monitoring network evolution As our network is interpretable, it is tempting to qualitatively analyze the progress of the network. To do that we record the PPF reconstruction output at discrete time steps and visualize the PPFs as explained in § 3.1. Fig 5 shows such a visualization for different local patches. First, thanks to the representation power, our network achieves high fidelity recovery of PPFs. Note that even though the network

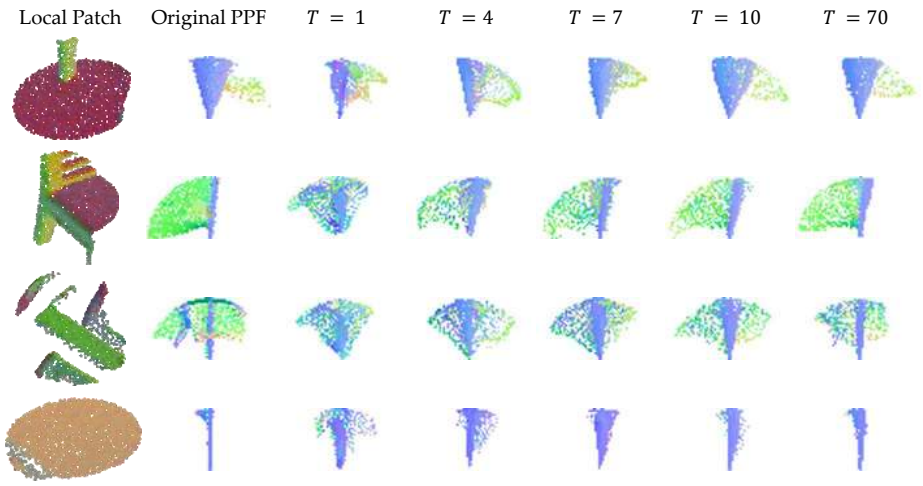


Fig. 5. Visualizing signatures of reconstructed PPFs. As the training converges, the reconstructed PPF signatures become closer to the original signatures. Our network reveals the underlying structure of the PPF space.

starts from a random initialization, it can quickly recover a desired point pair feature set, even after only a small number of iterations. Next, for similar local patches (top and bottom rows), the reconstructions are similar, while for different ones, different.

Visualizing the latent space We now attempt to visualize the learned latent space and assess whether the embedding is semantically meaningful. To do so, we compute a set of codewords and the associated PPF signatures. We then run the Barnes Hut T-SNE algorithm [24, 42] on the extracted codewords and form a two-dimensional embedding space, as shown in Fig. 6. At each 2D location we paint the PPF signature and thereby illustrate the distribution of PPFs along the manifold. We also plot the original patches which generated the codewords and their corresponding signatures as cutouts. Presented in Fig. 6, whenever the patches are geometrically and semantically close, the computed descriptors are close, and whenever the patches have less physical similarity, they are embedded into different parts of the space. This provides insight into the good performance and meaningfulness in the relationships our network could learn. In a further experiment, we extract a feature at each location of the point cloud. Then, we reduce the dimension of the latent space to three via TSNE [24], and colorize each point by the reduced feature vector. Qualitatively justifying the repeatibility of our descriptors, the outcome is shown in Fig. 7. Note that, descriptors extracted by the proposed approach lead to similar colors in matching regions among the different fragments.

5 Concluding Remarks

We have presented PPF-FoldNet, an unsupervised, rotation invariant, low complexity, intuitive and interpretable network in order to learn 3D local features solely from point

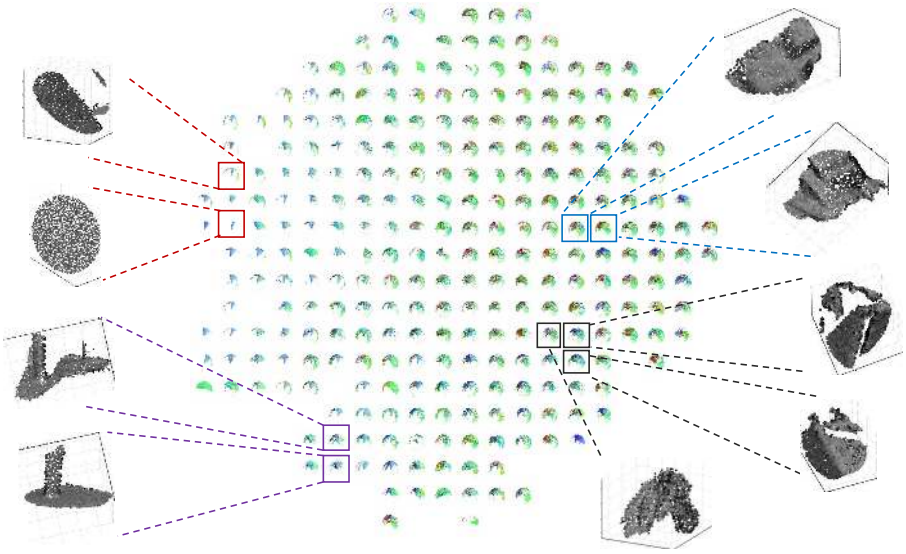


Fig. 6. Visualization of the latent space of codewords, associated PPFs and samples of clustered local 3D patches using TSNE [24, 42].

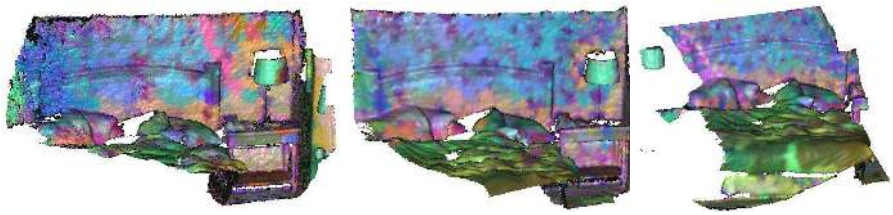


Fig. 7. Visualization of the latent feature space on fragments fused from different views. To map each feature to a color on the fragment, we use TSNE embedding [24]. We reduce the dimension to three and associate each low dimensional vector with an RGB color.

geometry information. Our network is built upon its contemporary ancestors, PointNet, FoldingNet & PPFNet and it inherits best attributes of all. Despite being rotation invariant, we have outperformed all the state-of-the-art descriptors, including supervised ones even in the standard benchmarks under challenging conditions with varying point density. We believe PPF-FoldNet offers a promising new approach to the important problem of unsupervised 3D local feature extraction and see this as an important step towards unsupervised revolution in 3D vision.

Our architecture can be extended in many directions. One of the most promising of those would be to adapt our features towards tasks like classification and object pose estimation. We conclude with the hypothesis that the generalizability in our unsupervised network should transfer easily into solving other similar problems, giving rise to an open application domain.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., et al.: Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467 (2016)
2. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning Representations and Generative Models for 3D Point Clouds. In: International Conference on Machine Learning (ICML) (2018)
3. Birdal, T., Ilic, S.: Point pair features based object detection and pose estimation revisited. In: 3D Vision. pp. 527–535. IEEE (2015)
4. Birdal, T., Ilic, S.: Cad priors for accurate and flexible instance reconstruction. In: Computer Vision (ICCV), 2017 IEEE International Conference on. pp. 133–142. IEEE (2017)
5. Birdal, T., Ilic, S.: A point sampling algorithm for 3d matching of irregular geometries. In: International Conference on Intelligent Robots and Systems (IROS 2017). IEEE (2017)
6. Bobkov, D., Chen, S., Jian, R., Iqbal, M.Z., Steinbach, E.: Noise-resistant deep learning for object classification in three-dimensional point clouds using a point pair descriptor. IEEE Robotics and Automation Letters **3**(2), 865–872 (2018)
7. Cao, Z., Huang, Q., Karthik, R.: 3d object classification via spherical projections. In: 3D Vision (3DV), 2017 International Conference on. pp. 566–574. IEEE (2017)
8. Deng, H., Birdal, T., Ilic, S.: Ppfnet: Global context aware local features for robust 3d point matching. Computer Vision and Pattern Recognition (CVPR). IEEE **1** (2018)
9. Elbaz, G., Avraham, T., Fischer, A.: 3d point cloud registration for localization using a deep neural network auto-encoder. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017)
10. Guo, Y., Sohel, F.A., Bennamoun, M., Wan, J., Lu, M.: Rops: A local feature descriptor for 3d rigid objects based on rotational projection statistics. In: Communications, Signal Processing, and their Applications (ICCSPA), 2013 1st International Conference on. pp. 1–6. IEEE (2013)
11. Hackel, T., Savinov, N., Ladicky, L., Wegner, J.D., Schindler, K., Pollefeys, M.: SEMANTIC3D.NET: A new large-scale point cloud classification benchmark. In: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. vol. IV-1-W1, pp. 91–98 (2017)
12. Halber, M., Funkhouser, T.: Fine-to-coarse global registration of rgb-d scans. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
14. Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W.: Surface reconstruction from unorganized points, vol. 26.2. ACM (1992)
15. Huang, H., Kalogerakis, E., Chaudhuri, S., Ceylan, D., Kim, V.G., Yumer, E.: Learning local shape descriptors from part correspondences with multiview convolutional networks. ACM Transactions on Graphics **37**(1) (2017)
16. Johnson, A.E., Hebert, M.: Using spin images for efficient object recognition in cluttered 3d scenes. IEEE Transactions on pattern analysis and machine intelligence **21**(5), 433–449 (1999)
17. Kehl, W., Milletari, F., Tombari, F., Ilic, S., Navab, N.: Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation. In: European Conference on Computer Vision. pp. 205–220. Springer (2016)
18. Khoury, M., Zhou, Q.Y., Koltun, V.: Learning compact geometric features. In: The IEEE International Conference on Computer Vision (ICCV) (Oct 2017)

19. Kinga, D., Adam, J.B.: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR) (2015)
20. Klokov, R., Lempitsky, V.: Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In: 2017 IEEE International Conference on Computer Vision (ICCV). pp. 863–872. IEEE (2017)
21. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 25, pp. 1097–1105. Curran Associates, Inc. (2012)
22. Lai, K., Bo, L., Fox, D.: Unsupervised feature learning for 3d scene labeling. In: Robotics and Automation (ICRA), 2014 IEEE International Conference on. pp. 3050–3057. IEEE (2014)
23. Lowe, D.G.: Object recognition from local scale-invariant features. In: Computer vision, 1999. The proceedings of the seventh IEEE international conference on. vol. 2, pp. 1150–1157. Ieee (1999)
24. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(Nov), 2579–2605 (2008)
25. Manessi, F., Rozza, A., Manzo, M.: Dynamic graph convolutional networks. arXiv preprint arXiv:1704.06199 (2017)
26. Maturana, D., Scherer, S.: Voxnet: A 3d convolutional neural network for real-time object recognition. In: Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on. pp. 922–928. IEEE (2015)
27. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: Orb-slam: a versatile and accurate monocular slam system. IEEE Transactions on Robotics **31**(5), 1147–1163 (2015)
28. Noh, H., Araujo, A., Sim, J., Weyand, T., Han, B.: Large-scale image retrieval with attentive deep local features. In: The IEEE International Conference on Computer Vision (ICCV) (Oct 2017)
29. Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J.: Frustum pointnets for 3d object detection from rgb-d data. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
30. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. Proc. Computer Vision and Pattern Recognition (CVPR), IEEE **1**(2), 4 (2017)
31. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 5099–5108. Curran Associates, Inc. (2017)
32. Qi, X., Liao, R., Jia, J., Fidler, S., Urtasun, R.: 3d graph neural networks for rgbd semantic segmentation. In: The IEEE International Conference on Computer Vision (ICCV) (Oct 2017)
33. Riegler, G., Ulusoy, O., Geiger, A.: Octnet: Learning deep 3d representations at high resolutions. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (Jul 2017)
34. Rusu, R.B., Blodow, N., Beetz, M.: Fast point feature histograms (fpfh) for 3d registration. In: Robotics and Automation, 2009. ICRA'09. IEEE International Conference on. pp. 3212–3217. IEEE (2009)
35. Salti, S., Tombari, F., Di Stefano, L.: Shot: Unique signatures of histograms for surface and texture description. Computer Vision and Image Understanding **125**, 251–264 (2014)
36. Shen, Y., Feng, C., Yang, Y., Tian, D.: Neighbors Do Help: Deeply Exploiting Local Structures of Point Clouds. ArXiv e-prints (Dec 2017)

37. Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., Fitzgibbon, A.: Scene coordinate regression forests for camera relocalization in rgb-d images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2930–2937 (2013)
38. Tatarchenko, M., Dosovitskiy, A., Brox, T.: Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In: IEEE International Conference on Computer Vision (ICCV) (2017)
39. Tatarchenko, M., Park, J., Koltun, V., Zhou, Q.Y.: Tangent convolutions for dense prediction in 3d. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
40. Tombari, F., Salti, S., Di Stefano, L.: Unique shape context for 3d data description. In: Proceedings of the ACM workshop on 3D object retrieval. pp. 57–62. ACM (2010)
41. Valentin, J., Dai, A., Nießner, M., Kohli, P., Torr, P., Izadi, S., Keskin, C.: Learning to navigate the energy landscape. In: 3D Vision (3DV), 2016 Fourth International Conference on. pp. 323–332. IEEE (2016)
42. van der Maaten, L.: Barnes-Hut-SNE. ArXiv e-prints (Jan 2013)
43. Wang, P.S., Liu, Y., Guo, Y.X., Sun, C.Y., Tong, X.: O-cnn: Octree-based convolutional neural networks for 3d shape analysis. ACM Transactions on Graphics (TOG) **36**(4), 72 (2017)
44. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic Graph CNN for Learning on Point Clouds. ArXiv e-prints (Jan 2018)
45. Wu, J., Zhang, C., Xue, T., Freeman, B., Tenenbaum, J.: Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In: Advances in Neural Information Processing Systems. pp. 82–90 (2016)
46. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1912–1920 (2015)
47. Xiao, J., Owens, A., Torralba, A.: Sun3d: A database of big spaces reconstructed using sfm and object labels. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1625–1632 (2013)
48. Yang, Y., Feng, C., Shen, Y., Tian, D.: Foldingnet: Point cloud auto-encoder via deep grid deformation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
49. Yi, K.M., Trulls, E., Lepetit, V., Fua, P.: Lift: Learned invariant feature transform. In: European Conference on Computer Vision. pp. 467–483. Springer (2016)
50. Yu, L., Li, X., Fu, C.W., Cohen-Or, D., Heng, P.A.: Pu-net: Point cloud upsampling network. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
51. Zeng, A., Song, S., Nießner, M., Fisher, M., Xiao, J., Funkhouser, T.: 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In: CVPR (2017)