Chapter 7

High-dimensional Convolutional Networks

7.1 Introduction

Finding structure in noisy data is a general problem that arises in many different disciplines. For example, robust linear regression requires finding the dominant pattern (line, plane) in some given, noisy data. 3D registration of point clouds requires the detection of erroneous correspondences [6]. Structure from motion (SfM) pipelines use verification based on prescribed geometric models to filter wrong image matches [38]. Novel algorithms for the detection of geometric structures can thus benefit many practical applications.

However, detecting geometric structures in noisy data is a hard problem. Data points that belong to a structure frequently comprise only a small fraction of the total amount of points, while the majority of the data points are outliers. Various algorithms and heuristics have been proposed over the years to cope with noisy data [36, 9, 32, 39, 3, 14, 19, 20], but they are usually specific to only a subset of the problems in this domain. Recent works have advocated for using deep networks [46, 33, 49] to learn robust models to classify geometric structures from outliers. Deep networks offer significant flexibility and the promise to replace hand-crafted algorithms and heuristics that can directly be learned from data. However, due to the unstructured nature of the data in geometric problems, existing works have considered such data as an unordered set, although they are embedded in a metric space and follow geometric structures. Also, prior works heavily relied on network structures that are composed of sequences of global pooling operations and multi-layer perceptrons (MLPs) that lack the capacity to model the local spatial geometry [31, 47].

In this work, we introduce a novel type of deep convolutional network that can operate in higher dimensions than merely 2D images or 3D volumes. Our network takes a sparse tensor as input

and employs convolutions as the fundamental computation unit in the high-dimensional spaces. The distinguishing property of our approach is that it is able to effectively leverage local neighborhood relations together with global context even for high-dimensional data. Our network is fully-convolutional, translation invariant, and integrates several best practices that have been successfully applied to various image recognition problems [34, 16, 21].

To demonstrate the effectiveness and generality of our approach, we tackle various geometric pattern recognition problems. We start with the basic problem of robust linear regression and simple geometry detection in high-dimensional noisy data. We show that our network can distinguish inliers from outliers in settings with extremely small signal-to-noise ratio as well as very high dimensions. We then show that our network can be applied to the problem of finding inlier correspondences that follow different geometric patterns and apply it to the filtering of correspondences between 3D scans as well as image correspondences that are subject to epipolar constraints.

Our experimental evaluation shows that our convolutional network is able to reliably detect geometric patterns in high-dimensional data that is heavily contaminated by noise. It can operate in regimes, where existing algorithms break down. Our approach significantly improves 3D registration performance when combined with standard approaches for 3D point cloud alignment [37, 51, 6, 52]. We also tackle the high-dimensional geometric pattern recognition in image correspondences and show that a high-dimensional convolutional network performs on par with state-of-the-art methods [46, 49]. All networks and training scripts are available at "https://github.com/chrischoy/HighDimConvNets.

7.2 Related Work

Robust model fitting. Fitting a geometric model to a set of observations that are contaminated by outliers is a fundamental problem that frequently arises in computer vision and related fields. The most widely used approach for robust geometric model fitting is *RANdom SAmple Consensus* (RANSAC) [13]. Due to its fundamental importance, many variants and improvements of RANSAC have been proposed over the years [36, 42, 9, 32, 23, 41, 39, 3].

Alternatively, algorithms for robust geometric model fitting are frequently derived using techniques from robust statistics [50, 14, 19, 20], where outlier rejection is performed by equipping an estimator with a cost function that is insensitive to gross outliers. While the resulting algorithms are computationally efficient, they require careful initialization and optimization procedures in order to not get stuck in poor local optima [51].

Another line of works proposes to find globally optimal solutions to the consensus maximization problem [24, 45, 5]. However, these approaches are currently computationally too demanding for many practical applications.

3D registration. Finding reliable correspondences between a pair of surfaces is an essential step for 3D reconstruction [6, 10, 29, 12]. The problem has been conventionally framed as an energy

minimization problem which can be solved using various techniques such as branch and bound [44], Riemannian optimization [35], mixed-integer programming [22], robust error minimization [51], semi-definite programming [26, 18], or random sampling [6].

Recently, the use of deep learning architectures for geometric registration tasks has received increased attention. PointNets, which recognize global patterns, are the predominant network architecture that are used for this problem [28, 30, 2].

We propose a high-dimensional convolutional network that can recognize not just global, but also local patterns that are crucial for 3D registration which requires detecting a contiguous geometric structure in a 6-dimensional space.

Image correspondence classification. Similar to the 3D registration problem, the design of effective deep learning architectures to solve classic problems in multiple view geometry, image matching, and camera localization is an active field of research.

Yi *et al.* [46] and Zhang *et al.* [49] pose essential matrix estimation as a correspondence inlier/outlier classification problems. Ranftl and Koltun [33] propose architectures for fundamental matrix estimation. Brachmann and Rother [4] propose to learn a neural network that guides hypothesis sampling in RANSAC for model fitting problems. Dang *et al.* [11] propose a numerically stable loss function for essential matrix estimation.

All of the previous works employ variants of PointNets to classify inliers in an unordered set of putative correspondences. These correspondences are embedded in a metric space and follow geometric structures. However, point-wise multi-layered perceptrons lack the capacity to model the spatial neighborhood of correspondences, which carry information of local geometry. In contrast, we propose a network that directly leverages neighborhood relations defined in the high-dimensional input space in every layer.

7.3 High-Dimensional Convolutional Networks

In this section, we briefly introduce the two main building blocks of our networks – generalized sparse tensors and generalized convolutions – and high-dimensional convolutional neural networks.

7.3.1 Sparse Tensor and Convolution

A tensor is a multi-dimensional array that represents high-order data. A D-th order tensor $\mathcal{T}$ requires D indices to uniquely access its element and we denote such indices or a coordinate as $\mathbf{x} = [x^1, ..., x^D]$ and the element at the coordinate as $\mathcal{T}[x^1, ..., x^D]$ similar to how we access components in a matrix. Likewise, a sparse tensor is a high-dimensional extension of a sparse matrix where the majority of

the elements are 0, or concisely,

$$\mathcal{T}[x_i^1, x_i^2, \cdots, x_i^D] = \begin{cases} \mathbf{f}_i & \text{if } (x_i^1, x_i^2, \cdots, x_i^D) \in \mathcal{C} \\ 0 & \text{otherwise,} \end{cases} \tag{7.1}$$

where $\mathcal{C} = \{\mathbf{x}_i \mid \mathbf{x}_i \in \mathbb{N}^D, \mathcal{T}[\mathbf{x}_i] \neq \mathbf{0}\}_{i=0}^N$ is the set of coordinates with non-zero values, N is the number of non-zero elements, and $\mathbf{f}_i$ is the non-zero value at i-th coordinate. A sparse feature map is a $(D+1)$-th order tensor with $\mathbf{f}_i \in \mathbb{R}^{N_{D+1}}$ as we use the last dimension to denote the feature dimension.

A sparse tensor has the constraint that $\mathbf{x}_i \in \mathbb{N}^D$. We extend the sparse tensor coordinates to integer indices $\mathbf{x}_i \in \mathbb{Z}^D$ and define $\mathcal{T} \in \mathbb{R}^{\aleph_0^D \times N_{D+1}}$ where $\aleph_0$ denotes the cardinality of the integer space $|\mathbb{Z}|$ to define a generalized sparse tensor.

A convolution on this generalized sparse tensor can then be defined as a simple extension of the generalized sparse convolution [7]:

$$\mathbf{f}_{\mathbf{u}}^{\text{out}} = \sum_{\mathbf{i} \in \mathcal{N}^D(\mathbf{u})} \mathbf{W}_{\mathbf{i}} \mathbf{f}_{\mathbf{u}+\mathbf{i}}^{\text{in}} \quad \text{for} \quad \mathbf{u} \in \mathcal{C}^{\text{out}}, \tag{7.2}$$

where $\mathcal{C}^{\text{out}}$ are the output locations that are predefined by the user and $\mathcal{N}^D(\mathbf{u})$ defines a set of neighbors of $\mathbf{u}$ which is defined by the shape of the convolution kernel. For example, if the convolution kernel is rectangular with size K, the set of neighbor are all non-zero element of the sparse tensor centered at $\mathbf{u}$ within the L_∞-ball with diameter K.

7.3.2 Convolutional Networks

We design a high-dimensional fully-convolutional neural network for sparse tensors (sparse tensor networks) based on generalized convolution [7, 8]. We use U-shaped networks [34] to capture large receptive fields while maintaining the original resolution at the final layer. The network has residual connections [16] within layers with the same resolution and across the network to speed up convergence and to recover the lost spatial resolution in the last layer. The network architecture is illustrated in Fig. 7.1.

To ensure computational efficiency in high dimensions, we use the cross-shape kernel [7] for all convolutions. We denote the kernel size by K. The cross-shape kernel has non-zero weights only for the K nearest neighbors along each axis which results in one weight parameter for the center location and K weight parameters for each axis. Note that a cross-shaped kernel is similar to separable convolution, where a full convolution is approximated by D one-dimensional convolutions of size K. Both types of kernels are rank-1 approximations of the full hyper-cubic kernel K^D, but separable convolution requires KD matrix multiplications, whereas the cross-shape kernel requires only $(K-1)D+1$ matrix multiplications.

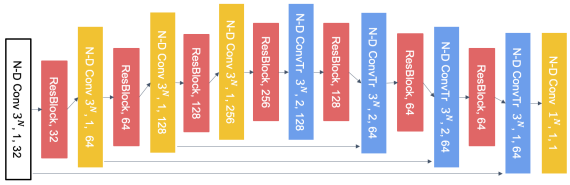


Figure 7.1: A generic U-shaped high-dimensional convolutional network architecture. The numbers next the each block indicates kernel size, stride, and the number of channels. The strided convolutions that reduce the resolution of activations are shifted upward to indicate different levels of resolution.

7.3.3 Implementation

We extend Choy *et al.* [7], which support arbitrary kernel shapes, to implement our high-dimensional convolutional networks. To efficiently implement the sparse tensor networks, we need an efficient data structure that can generate a new sparse tensor as well as find neighbors within the sparse tensor. Choy *et al.* [7] use a hash table that is efficient for both insertion and search. We replaced the hash table with a faster and more efficient variant [1]. In addition, as the neighbor search can be run in parallel, we create an iterator function that can run in parallel with OpenMP [27] by dividing the table into smaller parallelization blocks.

Lastly, U-shaped networks generate hierarchical feature maps that expand the receptive field and allow larger context to the neurons. Choy *et al.* [7] use stride-K convolutions with kernel size $\geq K$ to generate lower resolution hierarchical feature maps, but such implementation requires iterating over at least K^D elements within a hyper cubic kernel as the coordinates are stored in a hash table, which results in $O(NK^D)$ complexity where N is the cardinality of input. Consequently, it becomes infeasible to store the weights on GPU for high-dimensional spaces. Instead of strided convolutions, we propose an efficient implementation of stride-K sum pooling layers with kernel size K [8]. Instead of iterating over all possible neighbors, we iterates over all input coordinates and floors them down to multiples of K, which require only $O(N)$ complexity.

7.4 Geometric Pattern Recognition

We propose an approach based on a convolutional network to recognize geometric patterns in high-dimensional space. Specifically, we classify each point $\mathbf{x}_i$ in high-dimensional data $\mathcal{X} = \{\mathbf{x}_i\}_{i=0}^N$ either as an inlier or as an outlier. We start by validating our approach on synthetic datasets of varying dimensions, before we show results on 3D registration and essential matrix estimation from image correspondences.

For all experiments, we first quantize the input coordinates to create a sparse tensor of order $D + 1$, where the last dimension denotes the feature channels. The network then predicts a logit score for each non-zero element in the sparse tensor to indicate if a point is part of the geometric

Table 7.1: High-dimensional pattern detection and linear regression: Mean Squared Error (MSE) of the predicted line and F1 and the Average Precision (AP) of different methods. MSE: lower the better, F1 and AP: higher the better

		Qi *et al.* [31]			Zaheer *et al.* [47] + BN + IN			Yi *et al.* [46]			Ours		
Dim.	Inlier Ratio	MSE	F1	AP (AUC)	MSE	F1	AP (AUC)	MSE	F1	AP (AUC)	MSE	F1	AP (AUC)
4	15.59%	1.337	0.025	0.164	0.014	0.845	0.904	1.42E-5	0.993	0.999	**2.33E-5**	**0.998**	**0.999**
8	5.54%	2.369	0.022	0.065	0.005	0.865	0.975	2.46E-5	0.996	0.999	**1.64E-5**	**0.999**	**0.999**
16	2.75%	3.854	0.012	0.034	0.189	0.481	0.747	0.145	0.742	0.956	**3.39E-5**	**0.999**	**0.999**
24	0.40%	5.372	0.021	0.011	0.201	0.192	0.644	0.842	0.595	0.662	**5.34E-5**	**0.994**	**0.996**
32	0.07%	6.715	0.012	6.71E-5	2.513	0.010	0.327	-	0.0	0.052	**0.010**	**0.669**	**0.689**

Table 7.2: High-dimensional pattern detection and subspace detection: F1 and the Average Precision (AP) of different methods. F1 and AP: higher the better

		Qi *et al.* [31]		Zaheer *et al.* [47] + BN + IN		Yi *et al.* [46]		Ours	
Dim.	Inlier Ratio	F1	AP (AUC)	F1	AP (AUC)	F1	AP (AUC)	F1	AP (AUC)
4	29.96%	0.0	0.315	0.980	0.996	**0.993**	**0.999**	0.991	0.998
8	8.07%	0.0	0.088	0.985	0.999	0.990	0.999	**0.998**	**0.999**
16	0.34%	0.0	0.004	0.155	0.299	0.182	0.359	**0.951**	**0.961**
24	0.01%	0.0	1.61E-4	0.032	0.133	0.0	0.081	**0.304**	**0.346**
32	4.64E-3%	0.0	5.56E-5	0.0	0.221	0.0	0.023	**0.138**	**0.240**

Qi *et al.* [31] Zaheer *et al.* [47] + BN Yi *et al.* [46] Ours

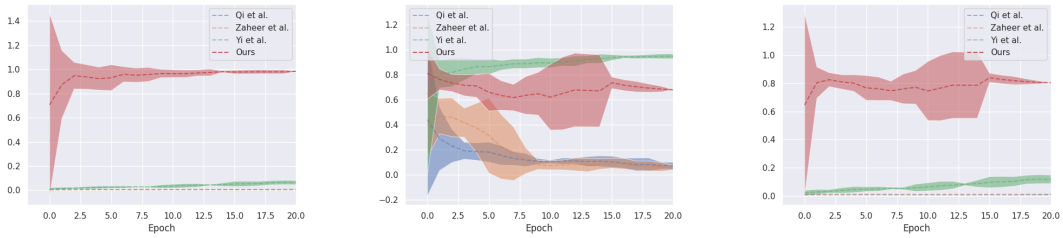
Figure 7.2: 8-dimensional linear regression projected to a 2-dimensional plane for visualization. Black dots are noise and blue dots are samples from a 8-dimensional line. The dotted red line is the prediction using each method. The signal-to-noise ratio is merely 3.83%.

pattern or if it is an outlier.

7.4.1 Linear Regression

We first test the capabilities of our fully-convolutional networks on simple high-dimensional linear regression problems. Our dataset consists of a large amount of noise uniformly sampled from the D-dimensional space and a small number of samples from a line with Gaussian noise. Thus, the number of noise increases exponentially to the dimension $O(L^D)$ and the number of inliers increases sublinearly $O(\sqrt{LD})$ where L is the length of the space. We put a detailed experiment setup in the supplementary materials. The network predicts a likelihood score for each non-zero element in the input sparse tensor and we threshold inliers with probability ≥ 0.5. We estimate the line equation from the predicted inliers using the unweighted least square method.

We use different PointNet variants as the baselines for this experiment [31, 47, 46]. For Zaheer *et*



Precision Recall F1

Figure 7.3: 32-dimensional linear regression: we plot the running mean and standard deviation of precision, recall and F1 score on the validation set after each epoch. Compared with other networks, the convolutional network converges faster due to the skip connections across U-network structure and convolution.

al. [47], we were not able to get reasonable results with the network architecture proposed in the paper. We thus augmented the architecture with batch normalization and instance normalization layers after each linear transformation similar to Yi *et al.* [46], which boosted performance significantly. For all experiments and networks, we use the cross-entropy loss. where $\mathcal{P}$ and $\mathcal{N}$ denote the set of inliers and outliers, respectively. We fixed all training hyperparameters, including loss, batch size, the optimization method, learning rate, and the learning rate scheduler for all baselines and our approach.

We use three metrics to analyze the performance of the networks: Mean Squared Error (MSE), F1 score, and the Average Precision (AP). For the MSE, we estimate the line equation with the Least Squares to fit the line to the inliers. The second metric is the F1-score, the harmonic mean of precision and recall. In many problems, F1-score is the direct indicator of the performance of a classifier, and we also found a strong correlation between F1-score and the mean squared error as well. The final metric we used is the average precision (AP) that measures the area under the precision-recall curve. We report the results in Tab. 7.1 and visualize qualitative examples in Fig. 7.2. Note that Tab 7.1 shows the inlier ratio to indicate the difficulty of each task.

In a second experiment, we create another synthetic dataset where the inlier pattern is sampled from a plane spanned by 2 vectors $\{c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \mathbf{c} \mid c_1, c_2 \in \mathbb{R}\}$. The 2 basis vectors are sample uniformly from the D-dimensional unit hypercube. We use the same training procedure for both baselines and our network and report the results in Tab. 7.2.

We found that the convolutional network is more robust to noise in high-dimensional spaces for linear regression than the variants of PointNet since convolutions and hierarchical feature maps can effectively use the geometric data and find spatially local patterns. In addition, we found that training the convolutional network converges very fast, as shown in Fig. 7.3, which is a further indicator that the architecture can effectively leverage the structure of the data.

7.4.2 3D Registration

A typical 3D registration pipeline consists of 1) feature extraction, 2) feature matching, 3) match filtering, 4) global registration. In this section, we show that in the *match filtering* stage the correct (inlier) correspondences form a 6-dimensional geometric structure and extend our simple geometric pattern recognition networks to filtering of putative 3D matches.

Let $\mathcal{X}$ be a set of points sample from 3D geometry, or 3D scanned points, $\mathcal{X} = \{\mathbf{x}_i \mid \mathbf{x}_i \in \mathbb{R}^3\}_{i=1}^N$, and $\mathcal{X}'$ be a subset of $\mathcal{X}$ that went through a rigid transformation T, $\mathcal{X}' = \{T(\mathbf{x}) \mid \mathbf{x} \in \mathcal{S}, \mathcal{S} \subseteq \mathcal{X}\}$, i.e. a 3D scan from a different perspective that has an overlap with $\mathcal{X}$. We denote a correspondence between points $\mathbf{x}_i \in \mathcal{X}$ and $\mathbf{x}'_j \in \mathcal{X}'$ as $\mathbf{x}_i \leftrightarrow \mathbf{x}'_j$. When we form an ordered pair $(\mathbf{x}_i, \mathbf{x}'_j) \in \mathbb{R}^6$, the ground truth correspondences satisfy $T(\mathbf{x}) = \mathbf{x}'$ along the common 3D geometry $\mathcal{S}$ whereas an incorrect correspondence implies $T(\mathbf{x}) \neq \mathbf{x}'$. Thus, the geometry $(\mathbf{x}, \mathbf{x}') \in \mathbb{R}^6$ or $(\mathbf{x}, T(\mathbf{x}))$ for $\mathbf{x} \in \mathcal{S}$ forms a surface in 6-dimensional space.

We thus can use our convolutional neural network to segment the 6-dimensional hypersurface into inlier and outlier correspondences by estimating the likelihood that each correspondence is an inlier. We use the likelihood to filter out outliers from putative correspondences.

Network. We use Yi *et al.* [46] and the 6-dimensional U-shaped convolutional network from Sec. 7.4.1 for this experiment. As the dimension is manageable, we use hypercubic kernels. The network takes a order-6 sparse tensor whose coordinates are discretized correspondences $(\mathbf{x}_i, \mathbf{x}'_j) \in \mathbb{R}^6$. We discretize the coordinates with the voxel size used to extract features. Our baseline Yi *et al.* [46] takes dimensionless mean-centered correspondences without discretization. We train the networks to predict the inlier probability of each correspondence with the balanced cross entropy loss.

Dataset. We use the 3DMatch benchmark [48] for this experiment. The 3DMatch dataset is a composition of a variety of 3D scan datasets [48, 15, 43] and thus covers a wide range of scenes and different types of 3D cameras. We integrate RGB-D images to form fragments of the scenes following [48]. During the training, we randomly rotate each scene on-the-fly to augment the dataset. We use one of the most widely used hand-designed features, FPFH [37] to compute correspondences. Note, however, that our pipeline is agnostic to the choice of feature and can also be used with the latest learned features [8].

We follow the standard procedures in the 3D registration literature to generate putative correspondence. First, since 3D scans often exhibit irregular densities, we down-sample the input point clouds in a voxel grid to produce a regular point point cloud. We use voxel sizes of 2.5cm and 5cm for our experiments. Next, we compute FPFH features and find the nearest neighbor for each point in features space to form putatitive correspondences. The correspondences obtained from this procedure often exhibit a very low inlier ratio, as low as 0.87% with 2.5cm voxel. Among these correspondences, we regard $\mathbf{x} \leftrightarrow \mathbf{x}'$ as an inlier if it satisfies $\|\mathbf{T}(\hat{\mathbf{x}}) - \hat{\mathbf{x}}'\|_2 < \tau$ and all others as outliers. We set τ to be two times the voxel downsampling size.

Table 7.3: Pairwise registration with FPFH [37] on 3DMatch test scenes with 2.5cm downsampling. Translation Error (TE), Rotation Error (RE), success rate. The pairwise registration is successful if TE < 30cm and RE < 15°.

		FPFH + FGR			FPFH + Ours + FGR			FPFH + RANSAC			FPFH + Ours + RANSAC		
	Inlier Ratio	TE	RE	Succ. Rate	TE	RE	Succ. Rate	TE	RE	Succ. Rate	TE	RE	Succ. Rate
Kitchen	1.62%	10.98	4.99	37.15	5.68	2.21	65.61	6.25	2.17	44.47	5.90	1.98	69.57
Home 1	2.71%	11.12	4.40	45.51	6.52	2.08	80.77	7.07	2.19	61.54	6.00	1.87	80.13
Home 2	2.83%	9.61	3.83	36.54	7.13	2.56	64.42	6.47	2.40	50.00	7.86	2.56	69.71
Hotel 1	1.35%	12.31	5.09	33.19	7.95	2.65	76.11	7.48	2.75	48.67	7.38	2.38	80.09
Hotel 2	1.54%	12.27	5.22	25.00	7.86	2.56	69.23	9.54	3.18	47.12	6.40	2.25	70.19
Hotel 3	1.59%	13.52	7.04	27.78	5.39	1.99	72.22	5.91	2.46	59.26	5.85	2.36	81.48
Study	0.87%	16.10	6.01	16.78	9.61	2.64	53.42	10.05	3.01	30.48	8.51	2.23	56.16
Lab	1.59%	10.48	4.80	42.86	7.69	2.44	61.04	8.01	2.31	45.45	6.64	2.12	68.83
Average		12.05	5.17	33.10	7.23	2.39	67.85	7.60	2.56	48.37	6.82	2.22	72.02

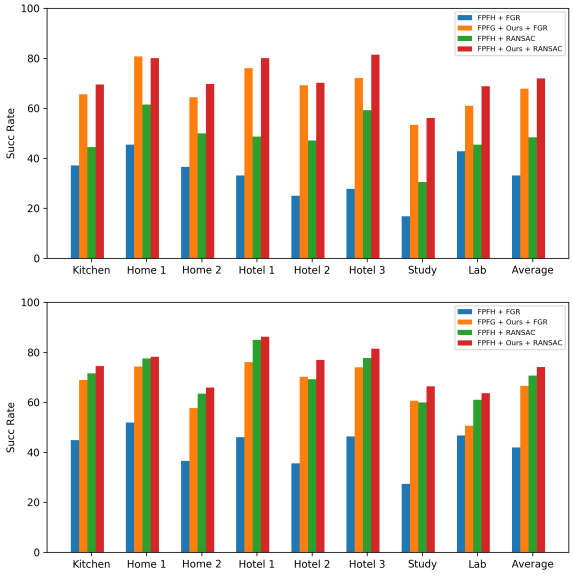


Figure 7.4: The success rate on the 3DMatch benchmark [48] with different voxel sizes: 2.5cm (top) and 5cm (bottom).

Table 7.4: Pairwise registration with FPFH [37] on 3DMatch test scenes with 5cm downsampling. Translation Error (TE), Rotation Error (RE), Recall in percent. The pairwise registration is successful if TE < 30cm and RE < 15°. The time excludes the feature extraction.

		FPFH + FGR			FPFH + Yi *et al.* [46] + FGR			FPFH + Yi *et al.* [46] + RANSAC			FPFH + Ours + FGR		
	SNR	TE	RE	Succ. Rate	TE	RE	Succ. Rate	TE	RE	Succ. Rate	TE	RE	Succ. Rate
Kitchen	4.90%	9.32	3.92	44.86	8.06	3.36	55.53	9.10	3.65	57.71	6.07	2.46	68.97
Home 1	7.50%	9.13	3.53	51.92	8.76	3.23	64.10	9.28	2.99	67.31	7.93	2.59	74.36
Home 2	6.65%	9.02	3.58	36.54	7.96	3.13	45.19	10.02	3.71	53.85	7.99	3.23	57.69
Hotel 1	5.22%	10.20	3.86	46.02	9.14	3.46	57.52	11.25	3.80	61.95	8.71	2.90	76.11
Hotel 2	4.75%	10.69	4.82	35.58	9.74	3.82	50.00	11.06	4.52	56.73	8.18	2.82	70.19
Hotel 3	5.20%	13.10	4.69	46.30	10.36	3.86	57.41	10.59	4.05	68.52	6.57	2.63	74.07
Study	3.83%	14.20	4.74	27.40	12.95	4.01	37.67	12.88	4.09	48.63	11.23	3.12	60.62
Lab	4.98%	9.33	3.60	46.75	7.51	3.26	49.35	8.85	2.94	50.65	6.45	2.04	50.65
Average		10.62	4.09	41.92	9.31	3.52	52.10	10.38	3.72	58.17	7.89	2.72	66.58

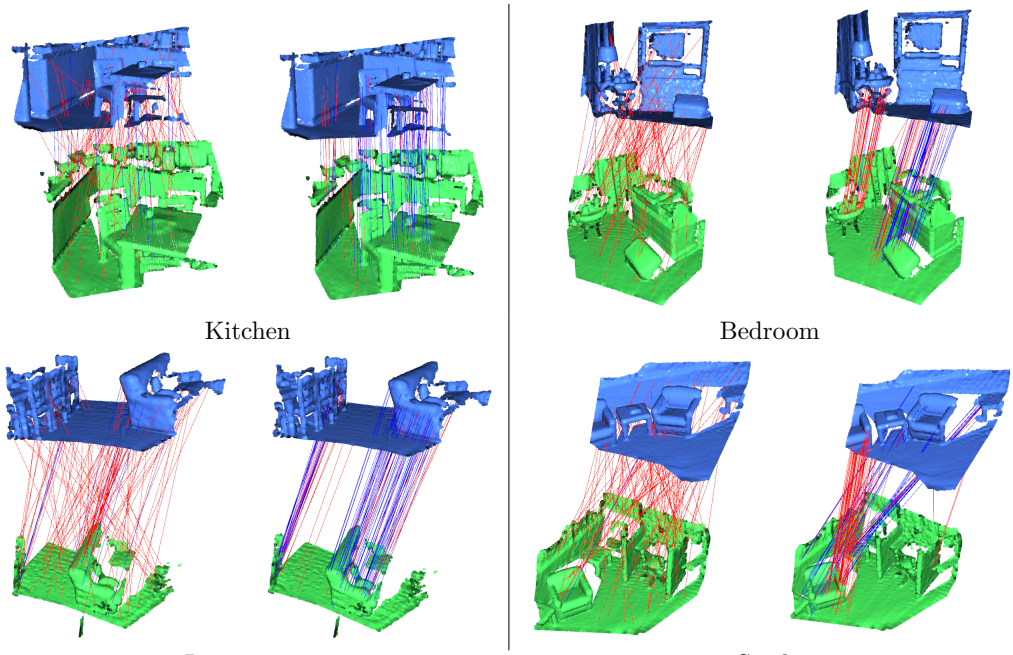


Figure 7.5: Visualization of color-coded correspondences before and after the hyper-surface recognition (Sec. 7.4.2). For each pair, we visualize 100 random correspondences on the left and another 100 random correspondences after the hyper plane detection with probability > 0.5. Red lines are outlier correspondences and blue lines are inlier correpondences. On the bottom right pair, there are two identical chairs. The average inlier ratio is 1.76%.

Finally, we use a registration method to convert the filtered correspondences into the final registration result. We show results we two different registration methods. The first is Fast Global Registration [51], which minimizes a robust error metric. The second method is a robust variant [52] of RANSAC [13].

Evaluation. We use three standard metrics to evaluate the performance of our network: rotation error, translation error, and success rate. The rotation error measures the absolute angular deviation from the ground truth rotation $\arccos \frac{\mathrm{Tr}(\hat{R}^T R)-1}{2}$. Similarly, the translation error measures the deviation of the translation $\|\hat{\mathbf{t}} - \mathbf{t}\|_2$. When we report these metrics, we exclude registrations that exceed a threshold following [8] since the registration methods [51, 13] return stochastic results if they fail to register an input pair. Finally, the success rate is the ratio of successful registration that has rotation error and translation error below a certain threshold, i.e. if either rotation or translation error exceeds the respective threshold, the registration is considered a failure. For all experiments, we use a rotation error of 15 degrees and a translation error 20cm as the thresholds.

Tab. 7.3 shows the 3D registration pipelines with and without our network to filter the outliers. Note that for FGR [51], we observe a considerable improvement with our network since FGR assume

more accurate correspondences as inputs. The improvement is smaller with RANSAC as it is more robust to a large amount of outlier. Although the inlier ratio is as low as 1% for many 3D scene pairs, out network generates very accurate predictions. Similar to the linear regression experiments in Fig. 7.3, we find that the network converges very quickly. We compare the improvement with respect to Yi *et al.* [46] in Tab. 7.4. We also study the robustness of the 6-dimensional convolutional network on the voxel size in Fig. 7.4 and find that the convolutional network imporoves the registration success rate significantly even for higher inlier ratio with 5cm voxels. Qualitatively, the network filters out outliers very accurately with even under the extreme level of noise (Fig. 7.5).

7.4.3 Filtering Image Correspondences

Table 7.5: Image matching evaluation with YFCC100M dataset.

	LMeDs [36]			MLESAC [42]			Yi *et al.* [46]			Zhang *et al.* [49]			Ours		
	Prec.	Recall	F1	Prec.	Recall	F1	Prec.	Recall	F1	Prec.	Recall	F1	Prec.	Recall	F1
Buckingham	0.213	0.178	0.194	0.294	0.299	0.297	0.497	0.772	0.605	0.535	0.804	0.642	**0.611**	**0.835**	**0.705**
Notre dame	0.335	0.197	0.248	0.489	0.422	0.453	0.581	0.894	0.705	0.679	0.901	0.774	**0.721**	**0.929**	**0.812**
Reichtag	0.380	0.217	0.276	0.573	0.441	0.498	0.747	0.877	0.807	**0.808**	0.878	**0.842**	0.769	**0.897**	0.827
Sacre coeur	0.203	0.104	0.137	0.418	0.292	0.344	0.658	0.871	0.750	**0.724**	0.902	0.803	0.718	**0.932**	**0.811**
Average	0.283	0.174	0.214	0.444	0.364	0.398	0.621	0.854	0.717	0.686	0.871	0.765	**0.704**	**0.898**	**0.789**

In this section, we apply the high-dimensional convolutional network to the task of image correspondence inlier detection. In the projective space $\mathbb{P}^2$, an inlier correspondence $\mathbf{u} \leftrightarrow \mathbf{u}'$ must satisfy $\mathbf{u}'^\top \mathbf{E}\mathbf{u} = 0$, where $\mathbf{E}$ is the essential matrix and $\mathbf{u}$ denotes the normalized homogeneous coordinate $\mathbf{u} = \mathbf{K}^{-1}\mathbf{x}$, $\mathbf{x}$ is the homogeneous image coordinate, and $\mathbf{K}$ is the camera intrinsic matrix. When we expand $\mathbf{u}'^\top \mathbf{E}\mathbf{u} = 0$, we get $u'^1 A u^1 + u'^2 B u^1 + u'^1 C u^2 + u'^2 D u^2 + E u'^1 + F u'^2 + G u^1 + H u^2 + I = 0$, which is a quadrivariate quadratic function. If there is a real valued solution, there are infinitely many solutions that form either an ellipse (sphere), a parablola, or a hyperbola. These are known as conic sections. Thus a set of ground truth image correspondences will form a hyper conic section in a 4-dimensional space. We use a convolutional network to predict the likelihood of each correspondence being an inlier.

Network. We convert a set of putative image correspondences into an order-5 sparse tensor with 4 spatial dimensions and vectorized features. The non-zero coordinates are defined at the discretized concatenated normalized image coordinates $\mathbf{u}$ which defines a consistent metric system which is invariant to varying camera intrinsics used in the dataset. The features are concatenated normalized coordinates. We use a few variants of U-shaped convolutional networks and state-of-the-art baselines for this task. The loss for all networks is the balanced cross entropy.

Dataset: YFCC100M. We use a large-scale photo-tourism dataset YFCC100M [40] for the experiment. The dataset contains 100M Flicker images of tourist hot-spots with metadata, which is curated into 72 locations with camera extrinsics estimated using SfM [17].

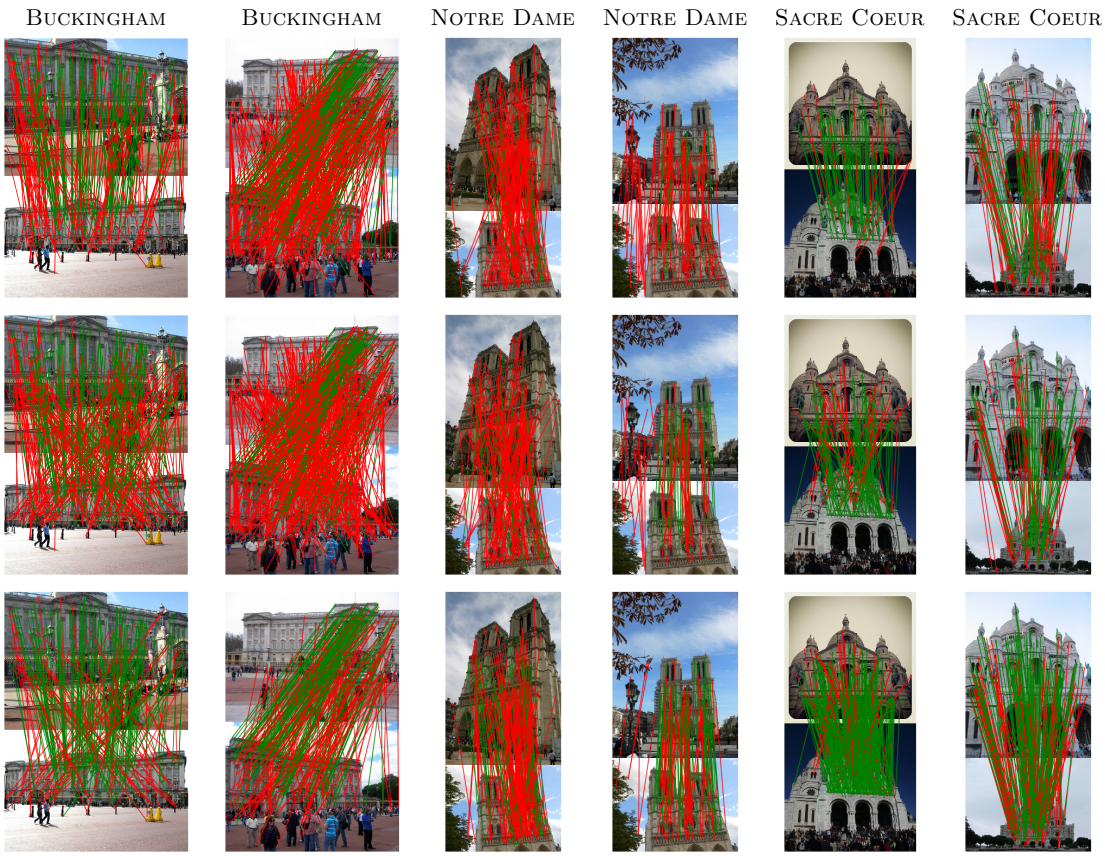


Figure 7.6: Matching results using Yi *et al.* [46] (top), Zhang *et al.* [49] (middle) and ours (bottom). Correspondences are colored as green if their symmetric epipolar distance is lower than 0.01.

We follow Zhang *et al.* [49] to generate the dataset and use 68 locations for training and the others for testing. We filtered any image pairs that have fewer than a specific number of overlapping 3D points from SfM to guarantee there is some overlap between images.

We use SIFT [25] features on to create correspondences and use the ratio-test to filter out noisy correspondences. Since the dataset only provides the camera parameters, we label a correspondence to be an inlier if the symmetric epipolar distance is below a certain threshold:

$$\left(\frac{r}{\sqrt{l_1^2 + l_2^2}} + \frac{r}{\sqrt{l_1'^2 + l_2'^2}}\right) < \tau, \tag{7.3}$$

where $l = \mathbf{u}'^{\top}\mathbf{F}$, $l' = \mathbf{F}\mathbf{u}^{\top}$, $r = \mathbf{u}'^{\top}\mathbf{F}\mathbf{u}$.

Evaluation. We use precision, recall, and F1-score to evaluate correspondence classification accuracy. We used $\tau = 0.01$ for the distance threshold following Yi *et al.* [46] and Zhang *et al.* [49] and classified a correspondence as an inlier if the estimated confidence is greater than 0.5. We use

the same hyper-parameters to train the baseline and ours and report the quantitative results in Tab. 7.5 and qualitative results on Fig. 7.6. We observed that our approach did not outperform the PointNet variants [46, 49] by a large margin. We attribute this to the sparse nature of SIFT keypoints. Unlike FPFH, where we densely sample keypoints, SIFT features are sampled only at sparse keypoints. This leads to fewer neighbors in a high-dimensional space, which consequently can lead to a degeneration of the convolution to a simple multi-layer perceptron.

7.5 Conclusion

Many interesting problems in computer vision involve finding geometric patterns in high-dimensional spaces. In this work, we propose high-dimensional convolutional networks for geometric pattern recognition. We validate it with linear regression, 6-dimensional hyper-surface detection for 3D registration, and 4-dimensional hyper-conic section detection for image correspondences. We will further analyze the network architectures, hyper-parameters, and conditions in which the high-dimensional convolutional network succeeds in the follow-up work.

Bibliography

[1] Martin Ankerl. Robin hood hashing. "https://github.com/martinus/robin-hood-hashing, 2019.

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

[3] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. DSAC - differentiable RANSAC for camera localization. In *CoRR*, 2016.

[4] Eric Brachmann and Carsten Rother. Neural-guided RANSAC: learning where to sample model hypotheses. *CoRR*, abs/1905.04132, 2019.

[5] Tat-Jun Chin, Pulak Purkait, Anders P. Eriksson, and David Suter. Efficient globally optimal consensus maximisation with tree search. *IEEE Transactions on Pattern Anaysis and Machine Intelligence*, 39(4):758–772, 2017.

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

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

[8] Christopher Choy, Jaesik Park, and Vladlen Koltun. Fully convolutional geometric features. In *ICCV*, 2019.

[9] Ondrej Chum and Jiri Matas. Matching with PROSAC - progressive sample consensus. In *CVPR*, 2005.

[10] Angela Dai, Matthias Nießner, Michael Zollöfer, Shahram Izadi, and Christian Theobalt. BundleFusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics 2017 (TOG)*, 2017.

[11] Zheng Dang, Kwang Moo Yi, Yinlin Hu, Fei Wang, Pascal Fua, and Mathieu Salzmann. Eigendecomposition-free training of deep networks with zero eigenvalue-based losses. In *ECCV*, 2018.

[12] Wei Dong, Jaesik Park, Yi Yang, and Michael Kaess. GPU accelerated robust scene reconstruction. In *IROS*, 2019.

[13] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.

[14] Andrew W. Fitzgibbon. Robust registration of 2D and 3D point sets. *Image and Vision Computing*, 21(13-14), 2003.

[15] Ben Glocker, Shahram Izadi, Jamie Shotton, and Antonio Criminisi. Real-time rgb-d camera relocalization. In *ISMAR*, 2013.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[17] Jared Heinly, Johannes L Schonberger, Enrique Dunn, and Jan-Michael Frahm. Reconstructing the world in six days. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3287–3295, 2015.

[18] M. B. Horowitz, N. Matni, and J. W. Burdick. Convex relaxations of SE(2) and SE(3) for visual pose estimation. In *ICRA*, 2014.

[19] Reza Hoseinnezhad and Alireza Bab-Hadiashar. An M-estimator for high breakdown robust estimation in computer vision. *Computer Vision and Image Understanding*, 115(8):1145–1156, 2011.

[20] Peter J Huber. *Robust statistics*. Springer, 2011.

[21] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 448–456. JMLR.org, 2015.

[22] Gregory Izatt and Russ Tedrake. Globally optimal object pose estimation in point clouds with mixed-integer programming. In *In International Symposium on Robotics Research*, 2017.

[23] Karel Lebeda, Jiri Matas, and Ondrej Chum. Fixing the locally optimized RANSAC. In *BMVC*, 2012.

[24] Hongdong Li. Consensus set maximization with guaranteed global optimality for robust geometry estimation. In *ICCV*, 2009.

[25] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

[26] Haggai Maron, Nadav Dym, Itay Kezurer, Shahar Kovalsky, and Yaron Lipman. Point registration via efficient convex relaxation. *ACM Transactions on Graphics (TOG)*, 35(4), 2016.

[27] OpenMP Architecture Review Board. OpenMP application program interface version 3.0, May 2008.

[28] G. Dias Pais, Pedro Miraldo, Srikumar Ramalingam, Venu Madhav Govindu, Jacinto C. Nascimento, and Rama Chellappa. 3DRegNet: A deep neural network for 3D point registration. *CoRR*, abs/1904.01701, 2019.

[29] Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Colored point cloud registration revisited. In *ICCV*, 2017.

[30] Thomas Probst, Danda Pani Paudel, Ajad Chhatkuli, and Luc Van Gool. Unsupervised learning of consensus maximization for 3D vision problems. In *CVPR*, 2019.

[31] 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.

[32] Rahul Raguram, Ondrej Chum, Marc Pollefeys, Jiri Matas, and Jan-Michael Frahm. USAC: A universal framework for random sample consensus. *IEEE Transactions on Pattern Anaysis and Machine Intelligence*, 35(8):2022–2038, 2013.

[33] René Ranftl and Vladlen Koltun. Deep fundamental matrix estimation. In *ECCV*, 2018.

[34] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, 2015.

[35] D. M. Rosen, L. Carlone, A. S. Bandeira, and J. J. Leonard. A certifiably correct algorithm for synchronization over the special euclidean group. In *12th International Workshop on Agorithmic Foundations of Robotics*, 2016.

[36] Peter J. Rousseeuw. Least median of squares regression. *Journal of the American Statistical Association*, 79(388):871–880, 1984.

[37] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *ICRA*, pages 3212–3217, 2009.

[38] Schönberger, Johannes Lutz, and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016.

[39] Ruwan B. Tennakoon, Alireza Bab-Hadiashar, Zhenwei Cao, Reza Hoseinnezhad, and David Suter. Robust model fitting using higher than minimal subset sampling. *IEEE Transactions on Pattern Anaysis and Machine Intelligence*, 38(2):350–362, 2016.

[40] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *arXiv preprint arXiv:1503.01817*, 2015.

[41] Philip H. S. Torr. Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. *International Journal of Computer Vision*, 50(1):35–61, 2002.

[42] P. H. S. Torr and A. Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. *CVIU*, 78(1):138–156, 2000.

[43] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. Sun3D: A database of big spaces reconstructed using sfm and object labels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1625–1632, 2013.

[44] Jiaolong Yang, Hongdong Li, and Yunde Jia. Go-ICP: Solving 3D registration efficiently and globally optimally. In *ICCV*, 2013.

[45] Jiaolong Yang, Hongdong Li, and Yunde Jia. Optimal essential matrix estimation via inlier-set maximization. In *ECCV*, 2014.

[46] Kwang Moo Yi, Eduard Trulls, Yuki Ono, Vincent Lepetit, Mathieu Salzmann, and Pascal Fua. Learning to find good correspondences. In *CVPR*, 2018.

[47] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. In *NeuralIPS*. 2017.

[48] 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.

[49] Jiahui Zhang, Dawei Sun, Zixin Luo, Anbang Yao, Lei Zhou, Tianwei Shen, Yurong Chen, Long Quan, and Hongen Liao. Learning two-view correspondences and geometry using order-aware network. In *ICCV*, 2019.

[50] Zhengyou Zhang. Determining the epipolar geometry and its uncertainty: A review. *International Journal of Computer Vision*, 27(2):161–195, 1998.

[51] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Fast global registration. In *ECCV*, 2016.

[52] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.