Chapter 8

Global Registration

8.1 Introduction

Registering a set of 3D scans or point clouds has been studied extensively for reconstruction, tracking, pose estimation, and 3D object detection [27, 5, 25]. Thus, many methods have been proposed to improve the accuracy of one or more components required for 3D registration, such as features [18, 9, 33, 7], pose optimization [30, 38, 20, 43], and recently, end-to-end feature learning and registration [33, 2].

Recent end-to-end registration networks have advantages in many areas where some traditional registration pipelines fail due to the low accuracy of feature matching. However, these end-to-end approaches have a few drawbacks. First, PointNet-based globally pooled features reduce the spatial resolution [2] and thus decrease the final registration accuracy. Second, strong assumptions on the distribution of points and correspondences [33] do not hold for registering 3D scans with partial overlap or visibility.

In this work, we resolve the drawbacks of the previous works by proposing three modules that are essential for robust and accurate global registration: a 6D convolutional neural network for correspondence confidence estimation; a Weighted Procrustes loss for backprop-able robust registration solvers; and a robust optimizer in SE(3) for fine-tuning the registration. We validate them on the tasks of real-world pairwise registration and large scale scene reconstruction.

The first component is a 6D convolutional neural network that estimates inlier probability and captures the geometry of the 3D correspondences. In 2D registration, Yi *et al.* [39] and Ranftl and Koltun [26] show that stacked (u, v) coordinates can be regarded as features to estimate the confidence of correspondences through multi-layer perceptrons. Similarly, Dias *et al.* [23] propose a network for 3D registration under an artificial setup where they use the ground truth pose to preprocess correspondences. However, on a realistic setup without groundtruth for the test, this network suffers from severe label imbalance and fails. Instead, we utilize a series of convolutions to

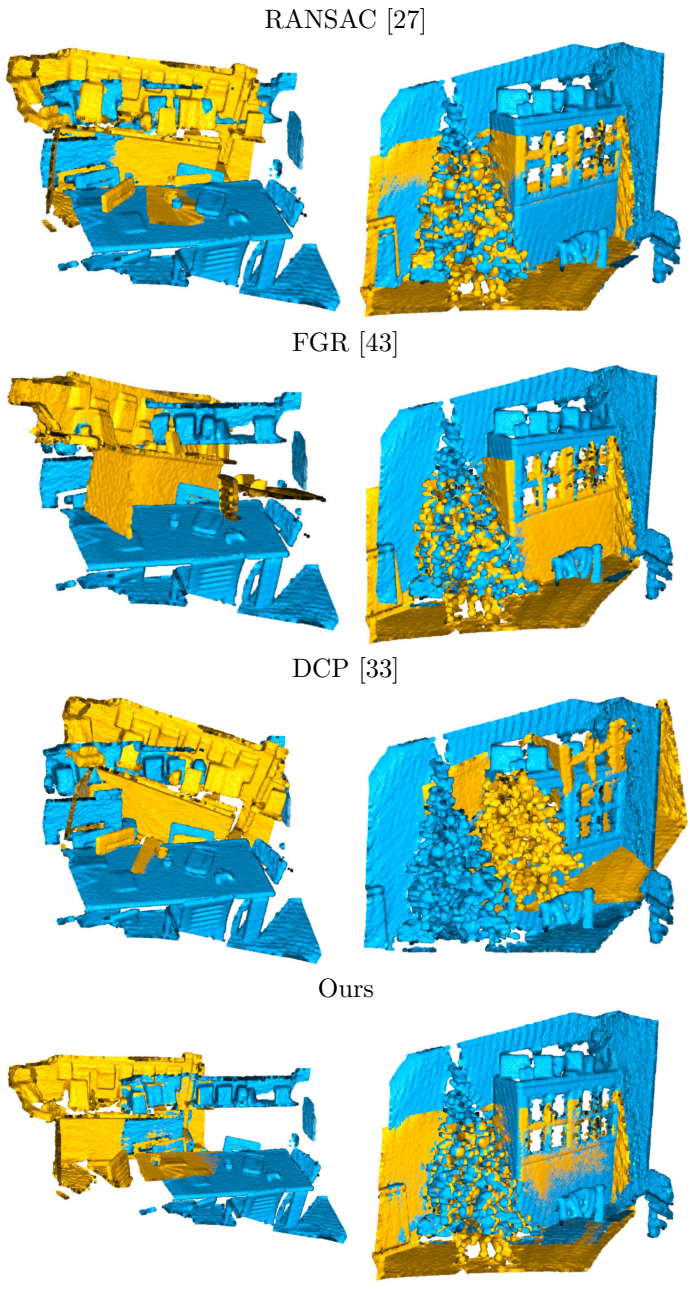

Figure 8.1: Registration results on the 3DMatch benchmark [40]. Two inputs are colored by blue and yellow respectively. Our method succeeds on challenging pairs while RANSAC [27], FGR [43], and DCP [33] all fail (*left*), and achieves finer alignment results on easier pairs (*right*).

process the 6D structure formed by correspondences for robust correspondence classification.

The second component is a differentiable Weighted Procrustes solver. Procrustes analysis [12]

minimizes the mean squared distance between correspondences. In particular, there exists a closed-form solution of Procrustes analysis between rigid objects in SE(3). Wang *et al.* [33] incorporate Procrustes analysis as a part of end-to-end training by predicting the corresponding 3D points in another 3D input. However, such training requires strong assumptions that fail in many real-world 3D scans. Instead, we approach this problem as a confidence prediction of correspondences and propose a differentiable Weighted Procrustes analysis. This change allows us to 1) reliably filter noisy correspondences; 2) handle partially observable scans with a small overlap like scans in Fig. 8.1; and 3) use the pipeline with non-differentiable features such as hand-crafted features [27, 32, 29] and deep features [9, 7].

Finally, we propose a robust optimization module to fine-tune the final registration prediction. The module can use any differentiable robust loss which we minimize with gradient descent on the continuous SE(3) representation space [45]. The final optimization is fast since it does not require a neighbor search per iteration [41].

Experimentally, we use real-world datasets [40, 6, 14, 24] to analyze the performance of our modules for pairwise and multi-way registration. We show that our modules are robust and accurate yet efficient compared with classical global registration algorithms [43, 27, 38] as well as recent end-to-end methods [23, 33, 2].

8.2 Related Work

We divide the related works of global registration into three categories: a) point-wise correspondence; b) outlier rejection; c) pose-optimization. Most studies work on one or more of these three categories to improve the global registration accuracy.

Geometric correspondence features. The first step in 3D registration is feature extraction that projects local/global geometry into a feature space that can be used for correspondence search, or global registration.

The first family of 3D features includes hand-crafted point-wise 3D features that capture local geometry [16, 29, 32, 28, 27] with histograms of pairwise or high-order properties between points. Recently, deep features have replaced many of these hand-crafted features. These features can be mainly categorized into PointNet-based [10, 9, 42] and ConvNet-based [40, 11, 7].

In this work, we do not make a clear distinction on the specific choice of features as the modules we propose can be applied on top of all these features including non-differentiable hand-crafted features.

Outlier rejection. In many cases, the registration step takes a set of correspondences as an input. These correspondences are generated by matching either learned or hand-crafted features, and usually suffer from noise and outliers. Various methods have been proposed to handle the problem. One of the most widely used family of registration methods is random sample consensus

(RANSAC) [30, 1, 27, 22, 15], while another family utilizes branch-and-bound [38]. These methods are more accurate, but suffer from slow run-time since the computational cost increases rapidly as the signal-to-noise ratio decreases. The last family of methods adopts robust loss functions [43, 4] to reject outliers during optimization. While being faster than RANSAC-based methods, they still require proper filtering of the initial correspondence candidates, and thus are less stable on noisy data.

In this work, we use a convolutional neural network to find the 3D structure within the correspondences for outlier rejection or confidence estimation. This requires one feed-forward step and thus avoids iterative random sampling.

Pose optimization. Pose optimization is the final stage that minimizes loss over filtered correspondences. Iterative Closest Points (ICP) [3] and Fast Global Registration (FGR) [43] use a quadratic loss function with second-order optimization to refine poses. Maken *et al.* [21] propose to accelerate this process by stochastic gradient descent.

Recently, end-to-end frameworks are proposed combining feature learning and pose optimization for CAD model registration. Aoki et al. [2] combine PointNet global features with pose optimization [20]. Wang et al. [33, 34] connect the Procrustes method with graph net features.

In this work, we propose to finely decouple the correspondence search and registration in end-to-end registration into multiple stages: correspondence search, correspondence confidence prediction, and Weighted Procrustes for registration. With these changes, we can use arbitrary features for correspondence matching and handle noisy correspondences better. Thus, the proposed modules improve the overall performance of the model and can register challenging real-world scenes with small overlaps.

8.3 Deep Global Registration

Typical 3D reconstruction systems take a sequence of partial 3D scans as inputs and recover a complete 3D model of the scene. These scans are partial observations or fragments of a scene, as shown in Fig. 8.1. In order to reconstruct the scene, we have to align these fragments with each other accurately. Thus, pair-wise registration serves as a critical step of reconstruction.

Pair-wise registration starts with mapping the local geometry into point-wise features for correspondence search. We then find feature correspondences globally to avoid local minima. Next, we predict the inlier probability of a correspondence. For this, we propose a novel 6D convolutional network to capture the high-dimensional structure of correspondences and predict confidence. Finally, we propose a Weighted Procrustes method to align 3D scans given correspondences and associated confidence, and refine the result by optimizing a robust loss function.

Before we discuss the details, we first define notations that will be used throughout the paper. We define a query point cloud with N_x points as $X = [\mathbf{x}_1, ..., \mathbf{x}_{N_x}]$ and a reference point cloud with

N_y points as $Y = [\mathbf{y}_1, ..., \mathbf{y}_{N_y}]$ where $\mathbf{x}_i, \mathbf{y}_j \in \mathbb{R}^3$. Also, we denote a correspondence as $\mathbf{x}_i \leftrightarrow \mathbf{y}_j$ or more succinctly (i, j).

8.3.1 Feature Extraction

The first step in registration is the feature extraction stage that converts the local/global geometry to a metric feature vector space for the nearest neighbor search. Although our registration network can incorporate arbitrary features, we use the dense Fully Convolutional Geometric Features (FCGF) [7] to capture the geometry of the input more accurately. The dimension of the convolutional features can be as compact as 16 to 32, which also makes it ideal for fast nearest neighbor search in the feature space.

8.3.2 Correspondence Confidence Prediction

Given the features $\mathcal{F}_x$ and $\mathcal{F}_y$ of two 3D scans, we use the nearest neighbors in the feature space to generate correspondences. This procedure is deterministic and discrete, and therefore can be hand-crafted to filter noisy correspondences with ratio or reciprocity test. However, we learn this heuristic filtering process through a 6-dimensional convolutional network and find the underlying structure in 6D space concatenated by 3D correspondence candidates.

To understand this better, we first use a 1-dimensional analogy and later generalize it to the 3-dimensional space. Let A be a set of 1-dimensional points $A = \{0, 1, 2, 3, 4\}$ and B be another set of points $B = \{10, 11, 12, 13, 14\}$ that is a translation of A such that $B = \{a_i + 10 | a_i \in A\}$. If an algorithm returns a set of possible correspondences $\{(0, 10), (1, 11), (2, 12), (3, 13), (4, 14), (0, 14), (4, 10)\}$, then the set of correct correspondences (inliers) will form a line (first 5 pairs), whereas incorrect correspondences (outliers) will form random noise outside the line (last 2 pairs). If we extend this to 3D space, when we concatenate the coordinates of the correspondences $\mathbf{x}_i \leftrightarrow \mathbf{y}_j$ as $[\mathbf{x}_i^T, \mathbf{y}_j^T]^T \in \mathbb{R}^6$, the inliers will form a smooth 6D surface that follows the geometry of the input. Meanwhile, the outliers will be scattered and appear as random noise outside the surface. Thus, to capture the geometry of this 6D structure better, we use a 6D convolutional network to predict the confidence of each correspondence.

Note that the convolution is translation invariant, thus our 6D convolutional network will generate the same output regardless of the absolute position of inputs in SE(3). We use the similar network architecture proposed in [7] to create a 6D sparse convolutional network with skip connections within the same tensor stride across the network. During training, we use binary cross-entropy loss between the prediction and the ground truth correspondences to optimize the network parameters. The visualization of the network is in Fig. 8.2.

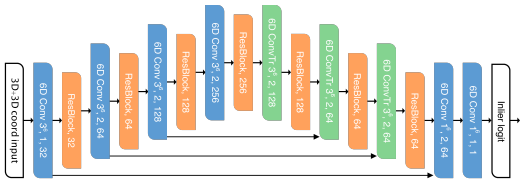

Figure 8.2: The 6-dimensional convolutional network architecture for inlier confidence prediction (Sec. 8.3.2). The network has a U-net structure with residual blocks between strided convolutions. Best viewed on the screen.

8.3.3 Weighted Procrustes for SE(3)

The confidence prediction from the previous module can be used to weigh the importance of correspondences. Given binary predictions, we can use the Procrustes method to minimize the mean squared error between corresponding points. However, this procedure is not differentiable and is sensitive to noise. Instead, Wang *et al.* [33] propose to differentiate the weighted sum of the predicted location of points, which implicitly assumes that there exists at least one point that corresponds to a point within the convex hull of the input. This assumption does not hold for many real-world 3D scans due to self-occlusion, perspective changes, etc. Instead, we propose a Weighted Procrustes method to handle noisy correspondences and pass gradients through the weights of correspondences rather than the predicted positions.

Procrustes method [12] minimizes the mean squared error between corresponding points $\frac{1}{N}\sum \|\mathbf{x}_i - \mathbf{y}_j\|^2$. Similarly, a Weighted Procrustes analysis minimizes the weighted mean squared error

$$
\begin{aligned}
e^2 &= e^2(R, \mathbf{t}; \mathbf{w}, X, Y) && (8.1)\\
&= \sum_i \tilde{w_i}(\mathbf{y}_{\mathcal{I}_i} - (R\mathbf{x}_i + \mathbf{t}))^2 && (8.2)\\
&= \mathrm{Tr}\left((Y - RX - \mathbf{t}\mathbf{1}^T)W(Y - RX - \mathbf{t}\mathbf{1}^T)^T\right), && (8.3)
\end{aligned}
$$

where $\mathbf{1} = (1, ..., 1)^T$, $X = [\mathbf{x}_1, ..., \mathbf{x}_n]$, $Y = [\mathbf{y}_{I_1}, ..., \mathbf{y}_{\mathcal{I}_n}]$. $\mathcal{I}$ is the permutation that defines the correspondences $\mathbf{x}_i \leftrightarrow \mathbf{y}_{\mathcal{I}_i}$. $\mathbf{w} = [w_1, \cdots, w_n]$ is the input weight vector predicted by the previous module, $\tilde{\mathbf{w}} = [\tilde{w_1}, \cdots, \tilde{w_n}] \triangleq \frac{\phi(\mathbf{w})}{||\phi(\mathbf{w})||_1}$ denotes the normalized weight after a non-linear transformation ϕ that performs prefiltering, and $W = \mathrm{diag}(\tilde{\mathbf{w}})$ forms the diagonal weight matrix. We now prove that there exists a closed form solution of $\arg\min_{R,\mathbf{t}} e^2$.

Theorem 1 *: The R and $\boldsymbol{t}$ that minimize the squared error $e^2(R, \boldsymbol{t}) = \sum_i w_i(\boldsymbol{y}_j - R\boldsymbol{x}_i - \boldsymbol{t})^2$ are $\hat{\boldsymbol{t}} = (Y - RX)W\mathbf{1}$ and $\hat{R} = USV^T$ where $U\Sigma V^T = SVD(\Sigma_{xy})$, $\Sigma_{xy} = YKWKX^T$, $K = I - \sqrt{\tilde{\mathbf{w}}}\sqrt{\tilde{\mathbf{w}}}^T$, and $S = diag(1, \cdots, 1, det(U)det(V))$.*

Proof. First, we differentiate e^2 w.r.t. $\mathbf{t}$ and equates the partial derivative to 0.

$$\frac{\partial}{\partial \mathbf{t}} e^2 = \frac{\partial}{\partial \mathbf{t}} \sum_i \tilde{\mathbf{w}}_i (\mathbf{y}_i - R\mathbf{x}_i - \mathbf{t}) \tag{8.4}$$

$$= -2 \left(\sum_i \tilde{\mathbf{w}}_i \mathbf{y}_i - \sum_i \tilde{\mathbf{w}}_i R\mathbf{x}_i - \sum_i \tilde{\mathbf{w}}_i \mathbf{t} \right) = 0 \tag{8.5}$$

Thus, $\hat{\mathbf{t}} = (Y - RX)W\mathbf{1}$. Next, we substitute $X = KX + X\sqrt{\tilde{\mathbf{w}}}\sqrt{\tilde{\mathbf{w}}}^T$ on Eq. 8.3 and do the same for Y.

$$e^2 = \mathrm{Tr}\left((Y - RX - \mathbf{t}\mathbf{1}^T)W(Y - RX - \mathbf{t}\mathbf{1}^T)^T\right) \tag{8.6}$$

$$= \mathrm{Tr}\left((YK - RXK + A)W(YK - RXK + A)^T\right)$$

$$= \mathrm{Tr}((YK - RXK)W(YK - RXK)^T) \tag{8.7}$$

$$= \mathrm{Tr}(YKWK^TY^T) + \mathrm{Tr}(RXKWK^TX^TR^R) - 2\mathrm{Tr}(YKWK^TX^TR^T) \tag{8.8}$$

where $A = Y\sqrt{\tilde{\mathbf{w}}}\sqrt{\tilde{\mathbf{w}}}^T - RX\sqrt{\tilde{\mathbf{w}}}\sqrt{\tilde{\mathbf{w}}}^T - \mathbf{t}\mathbf{1}^T$. We used the fact that $W\mathbf{1}\mathbf{1}^T = \sqrt{\tilde{\mathbf{w}}}\sqrt{\tilde{\mathbf{w}}}^T$. To minimize the the weighted squared error, we have to maximize the last negative term.

$$\max_R \mathrm{Tr}(YKWK^TX^TR^T) = \sum_i \sigma_i(YKWK^TX^T) \tag{8.9}$$

where $\sigma_i(A)$ is the i-th largest singular value of the matrix A. Thus, the maximum of the above equation occurs when $R = USV^T$ where $U\Sigma V^T = \mathrm{SVD}(\Sigma_{xy})$, $\Sigma_{xy} = YKWKX^T$ and $S = \mathrm{diag}\,(1, \cdots, 1, \det(U)\det(V))$. The last $\det(U)\det(V)$ is either +1 or -1 depending on the direction of the orthonomal basis. □

We can easily extend the above theorem to incorporate a scaling factor $c \in \mathbb{R}^+$ for the tasks like scan to CAD registration, but in this paper we assume that real-world scans have the same scale, which is true for most 3D sensors.

Weighted Procrustes generates rotation $\hat{R}$ and translation $\hat{\mathbf{t}}$ as outputs that depend on the weight vector $\mathbf{w}$ as the input. In our current implementation, $\hat{R}$ and $\hat{\mathbf{t}}$ are directly sent to the robust registration module in Sec. 8.4 as an initial pose. However, we briefly demonstrate that they can also be embedded in an end-to-end registration pipeline, as Weighted Procrustes is differentiable. From a top-level loss function L of $\hat{R}$ and $\hat{\mathbf{t}}$, we can pass the gradient through the closed-form solver, and update parameters in downstream modules:

$$\frac{\partial}{\partial \mathbf{w}} L(\hat{R}, \hat{\mathbf{t}}) = \frac{\partial L(\hat{R}, \hat{\mathbf{t}})}{\partial \hat{R}} \frac{\partial \hat{R}}{\partial \hat{\mathbf{w}}} + \frac{\partial L(\hat{R}, \hat{\mathbf{t}})}{\partial \hat{\mathbf{t}}} \frac{\partial \hat{\mathbf{t}}(\hat{R}, \hat{\mathbf{w}})}{\partial \hat{\mathbf{w}}}, \tag{8.10}$$

where $L(\hat{R}, \hat{\mathbf{t}})$ can be defined as the combination of differentiable rotation error (RE) and translation error (TE) between predictions $\hat{R}, \hat{\mathbf{t}}$ and groundtruth $R^*, \mathbf{t}^*$:

$$L_{\text{rot}}(\hat{R}) = \arccos \frac{\text{Tr}(\hat{R}^T R^*) - 1}{2}, \tag{8.11}$$

$$L_{\text{trans}}(\hat{\mathbf{t}}) = ||\hat{\mathbf{t}} - \mathbf{t}^*||_2^2, \tag{8.12}$$

or the Forbenius norm of relative transformation matrices defined in [2, 33].

8.4 Robust Registration

In this section, we propose a registration fine-tuning step that minimizes a robust loss function of choice to improve the registration accuracy. We use a gradient-based method similar to [21], but we avoid finding the local correspondences. Instead, we rely on the correspondence confidence to define the loss which removes the need for an expensive local geometric neighbor search. Also, we use a continuous representation for rotations since gradient-based methods are sensitive to discontinuities.

8.4.1 SE(3) Representation and Initialization

We use the 6D-representation of 3D rotation proposed by Zhou *et al.* [45] rather than Euler angle or quaternions which have discontinuities. The representation requires 6 parameters $\mathbf{a}_1, \mathbf{a}_2 \in \mathbb{R}^3$ which can be converted into a 3×3 orthogonal matrix by

$$f\left(\begin{bmatrix} | & | \\ \mathbf{a}_1 & \mathbf{a}_2 \\ | & | \end{bmatrix}\right) = \begin{bmatrix} | & | & | \\ \mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 \\ | & | & | \end{bmatrix}, \tag{8.13}$$

where $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3 \in \mathbb{R}^3$ are $\mathbf{b}_1 = N(\mathbf{a}_1)$, $\mathbf{b}_2 = N(\mathbf{a}_2 - (\mathbf{b}_1 \cdot \mathbf{a}_2)\mathbf{b}_1)$, and $\mathbf{b}_3 = \mathbf{b}_1 \times \mathbf{b}_2$, and $N(\cdot)$ denotes the normalization operation. Thus, the final representation that we use is $\mathbf{a}_1, \mathbf{a}_2, \mathbf{t}$ which are equivalent to $R, \mathbf{t}$ using Eq. 8.13.

To initialize $\mathbf{a}_1, \mathbf{a}_2$, we simply use the first two columns of the rotation matrix R, i.e., $\mathbf{b}_1$, $\mathbf{b}_2$. For convenience, we define f^{-1} as $f^{-1}(f(R)) = R$ though this inverse is not unique as there are infinitely many choices of $\mathbf{a}_1, \mathbf{a}_2$ that map to the same R.

8.4.2 Energy Minimization

Traditional ICP [3] or FGR [43] minimize L2 distance, i.e., mean squared error, between registered points. However, squared distance is sensitive to outliers as the function assumes Gaussian noise. Instead, we use more robust loss functions to fine-tune the registration between predicted inlier

correspondences. The general form of the energy function is

$$E(R, \mathbf{t}) = \sum_{i=1}^{n} \tilde{w}_i L(\mathbf{y}_{\mathcal{I}_i}, R\mathbf{x}_i + \mathbf{t}), \tag{8.14}$$

where $\tilde{w}_i$ and $\mathcal{I}_i$ are defined as in Eq. 8.2. $L(\mathbf{x}, \mathbf{y})$ is a point-wise loss function between $\mathbf{x}$ and $\mathbf{y}$; we choose Huber loss in our implementation. The energy function is parameterized by R and $\mathbf{t}$ which in turn are represented as $\mathbf{a}_1, \mathbf{a}_2, \mathbf{t}$. We can use first-order optimization algorithms such as SGD, ADAM, Adagrad, etc. to minimize the energy function, but higher-order optimizers are also applicable since the number of parameters is small. Note though sharing a similar form, our formulation is distinct from FGR since we incorporate predicted differentiable weights while FGR depends on weights computed from point-wise loss. The complete algorithm of our pipeline is described in Alg. 8.

Algorithm 8 Deep Global Registration

Require: $X \in \mathbb{R}^{n\times 3}, Y \in \mathbb{R}^{m\times 3}$
$\mathcal{F}_x \leftarrow \text{Feature}(X)$ // Sec. 8.3.1
$\mathcal{F}_y \leftarrow \text{Feature}(Y)$
$\mathcal{I}_{x\to y} \leftarrow \text{NearestNeighbor}(\mathcal{F}_x, \mathcal{F}_y)$ // Sec. 8.3.2
$\mathcal{C} \leftarrow \{X; Y[\mathcal{I}_{x\to y}]\ \}$
$\mathbf{w} \leftarrow \text{InlierProbability}(\mathcal{C})$
$\hat{R}, \hat{\mathbf{t}} \leftarrow \arg\min_{R,\mathbf{t}} e^2(R, \mathbf{t}; \mathbf{w}, X, Y)$ // Sec. 8.3.3
$\mathbf{a} \leftarrow f^{-1}(\hat{R}), \mathbf{t} \leftarrow \hat{\mathbf{t}}$ // Sec. 8.4
while not converging **do**
    $\ell \leftarrow \sum_c \tilde{w}_c L(Y_{\mathcal{I}_{x\to y}[c]}, f(\mathbf{a})X_c + \mathbf{t}), c \in \mathcal{C}$
    $\mathbf{a} \leftarrow \text{Update}(\mathbf{a}, \frac{\partial}{\partial \mathbf{a}}\ell(\mathbf{a}, \mathbf{t}))$
    $\mathbf{t} \leftarrow \text{Update}(\mathbf{t}, \frac{\partial}{\partial \mathbf{t}}\ell(\mathbf{a}, \mathbf{t}))$
end whilereturn $f(\mathbf{a}), \mathbf{t}$

8.5 Experiments

We train our network on the training split of the 3DMatch benchmark [40], which contains 3D point cloud pairs from various real-world scenes with ground truth transformations computed by RGB-D reconstruction pipelines [13, 8]. During training, we select pairs with at least 30% overlap and augment data by applying random rotations varying from -180 to 180 degrees around a random axis. Ground truth point-wise correspondences are found using the nearest neighbor search in 3D space.

We empirically downsample input point clouds by voxelizing input with a 5cm voxel size, and extract convolutional features [7] from the downsampled points for correspondence search. We train the 6-dimensional convolutional network on a single Titan XP with batch size 4. SGD is used with

an initial learning rate 10^{-1} and an exponential learning rate decay factor 0.99.

8.5.1 Pairwise Registration on 3DMatch

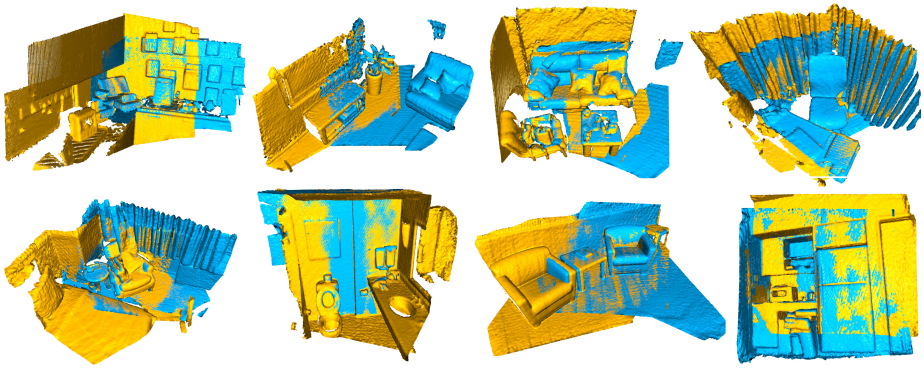

Figure 8.3: Global registration results of our method on all 8 different test scenes in 3DMatch [40]. Best viewed in color.

In this section, we test pairwise registration against baselines on the test set of the 3DMatch benchmark [40] which contains 8 different scenes as depicted in Fig. 8.3. We measure translation error (*TE*) defined in Eq. 8.11, rotation error (*RE*) defined in Eq. 8.12, and *recall*. Recall is the ratio of successful pair-wise registrations where RE and TE are smaller than predefined thresholds. Average TE and RE are computed only on these successfully registered pairs since failed registrations return random poses as predictions. In Table 8.1, we measure recall with the RE threshold 15 degrees and the TE threshold 30cm, which are practical thresholds for real-world global registrations. In Fig. 8.4, we plot the sensitivity of the recall on each threshold by changing one threshold and setting the other to infinity.

We compare our methods with various classical methods [43, 27, 38] and state-of-the-art learning

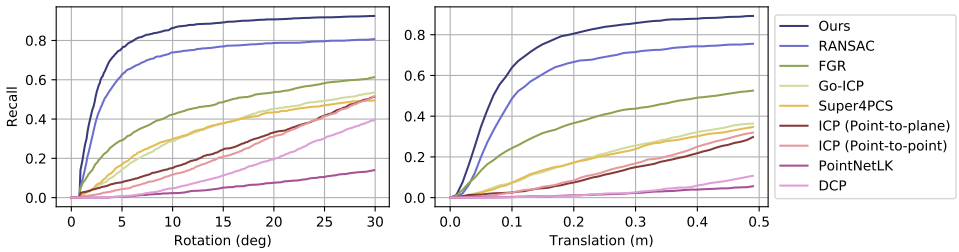

Figure 8.4: Overall precision recall curves for pairwise registrations on the 3DMatch benchmark with varying rotation and translation error thresholds. Our method achieves higher recall rate than all the baselines.

based methods [33, 2, 23]. All the experiments are evaluated on an Intel i7-7700 CPU and a GTX 1080Ti graphics card except for Go-ICP [38] tested on an Intel i7-5820K CPU. As an overview, Fig. 8.4 reports the overall precision-recall curves of our method and the baselines on the benchmark, while Fig. 8.5 includes detailed statistics on separate test scenes. Our system outperforms all the baselines in terms of RE, TE, and recall.

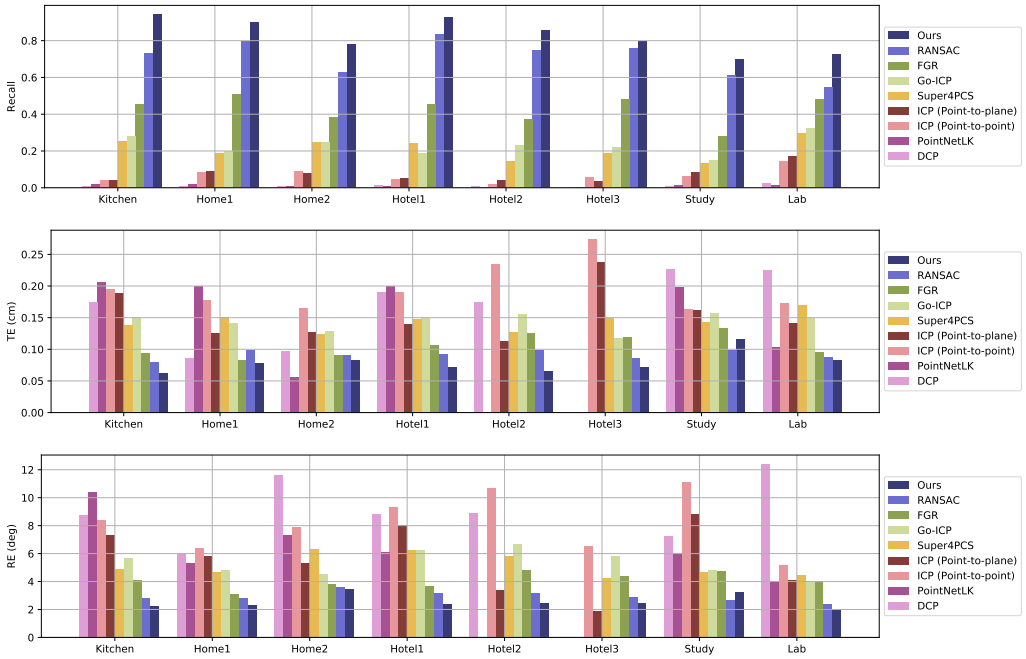

Figure 8.5: Analysis of registration results per scene. *Row 1*: recall rate, *the higher the better*. *Row 2-3*: TE and RE measured on successfully registered pairs, *the lower the better*. Our method is consistently better on all scenes unseen during training. Note: missing bars (e.g. PointNetLK) indicates no successful registration exists.

Classical methods. To compare with classical methods, we evaluate Point-to-point ICP, Point-to-plane ICP, RANSAC [27], and FGR [43], all implemented in Open3D [44]. In addition, we test the open source python binding of Go-ICP [38] and Super4PCS [22]. For RANSAC and FGR, we extract FPFH from voxelized point clouds. By default we set 4 million iterations and maximum 500 validations for RANSAC.

Since we perform global registration and assume unknown initial transformation, ICP variants mostly fail, as shown in Table 8.1. As a global registration method, Go-ICP has a higher recall rate than vanilla ICP, but the result is still less than 30% on average as it is sensitive to noise and small overlaps. It is also less efficient due to its expensive branch-and-bound search and the single-thread implementation. Super4PCS, as a sampling-based algorithm, performs similarly to Go-ICP. We found its default setting samples 200 points per iteration, leading to inaccurate registrations due to

Table 8.1: *Row 1-5*: registration results of our method and classical global registration methods on point clouds voxelized with 5cm voxel size. Our method outperforms RANSAC and FGR with comparable speed to FGR. *Row 6-9*: results of ICP variants, *Row 10 and 11*: results of learning-based methods. With relatively low recall rates, these methods generally fail on real-world scans.

	Recall	TE (cm)	RE (deg)	Time (s)
Ours	**85.2%**	**7.73**	**2.58**	0.70
FGR [43]	42.7%	10.6	4.08	**0.31**
RANSAC-2M [27]	66.1%	8.85	3.00	1.39
RANSAC-4M	70.7%	9.16	2.95	2.32
RANSAC-8M	74.9%	8.96	2.92	4.55
Go-ICP [37]	22.9%	14.7	5.38	771.0
Super4PCS [22]	21.6%	14.1	5.25	4.55
ICP (P2Point) [44]	6.04%	18.1	8.25	0.25
ICP (P2Plane) [44]	6.59%	15.2	6.61	0.27
DCP [33]	3.22%	21.4	8.42	0.07
PointNetLK [2]	1.61%	21.3	8.04	0.12

small overlaps in real-world scans. However, picking up more samples will drastically increase the runtime. We observe that on average a 20% recall from a 4-second runtime is a relatively balanced configuration.

Handcraft feature-based methods, FGR and RANSAC, perform better as shown in Table 8.1. When aligning point clouds voxelized with 5cm voxels, RANSAC achieves recall as high as over 70%, while FGR reaches 40%. Experimentally, in Table 8.1, increasing the number of RANSAC iterations by a factor of 2 only improves performance marginally.

In Fig. 8.4 and Table 8.1, our method outperforms all the classical methods on all metrics. Also, our method produces consistently lower TE and REs on all scenes. Note that our method only takes around half the time of RANSAC (default 4M iterations) to run, while achieving higher recall and registration accuracy.

Table 8.2: ATE (cm) error on the *Augmented ICL-NUIM* dataset with simulated depth noise. For InfiniTAM, loop closure module is disabled since it fails in all scenes. For BAD-SLAM, loop closure module only succeeds in *Livingroom 2*.

	ElasticFusion [36]	InfiniTAM [17]	BAD-SLAM [31]	Multi-way + FGR [43]	Multi-way + RANSAC [44]	Multi-way + Ours
Living room 1	66.61	46.07	fail	78.97	110.9	**21.06**
Living room 2	24.33	73.64	40.41	24.91	**19.33**	21.88
Office 1	**13.04**	113.8	18.53	14.96	14.42	15.76
Office 2	35.02	105.2	26.34	21.05	17.31	**11.56**
Avg. Rank	3	5	5	3.5	2.5	**2**

Learning-based methods. To compare with recent learning-based methods on the 3DMatch benchmark [40], we train off-the-shelf learning-based methods on the same 3DMatch training setup (Sec. 8.5) and evaluate them on the official test set. We use 3DRegNet [23], Deep Closest Point (DCP) [33], PRNet [34], and PointNetLK [2] as our baselines. We train all the baselines on 3DMatch with the same data augmentation as ours for fairness.

First, we train 3DRegNet [23] on the 3DMatch, which uses a multi-layer perceptron for inlier confidence prediction. We follow the setup outlined in [23], except that we do not manually filter outliers using groundtruth poses which are in practice not available during test time. Instead, we use raw FPFH correspondences for both training and testing phases, where the inlier ratio is around 10%. We find that while the inlier prediction accuracy can reach as high as 98% on the validation set, the registration loss of 3DRegNet does not converge w/ or w/o any data augmentation. On the test set, we find that the rotation errors and the translation errors are mostly above 30 degrees and 1m respectively.

We also train Deep Closest Point (DCP) [33] with embedded DGCNN [35] on 3DMatch. We first use voxel downsampling to generate the same downsampled input as ours, and randomly pick up 1024 points following [33]. Since DCP is expensive to train, we use two Titan X's for these experiments. We follow the same hyperparameters described in [33], and 1) train 250 epochs from scratch; 2) finetune the pretrained weights for 150 epochs. Although the loss of DCP does converge for all configurations, the best recall we get on the 3DMatch benchmark is 3.22%. We suspect that the singly stochastic matrix used by DCP fails to capture the correspondences between point clouds as the surjective mapping assumption is violated in real-world data.

DCP's successor, PRNet [34], suffers from random runtime error in its open source version by the time of the submission[1]. From our 30 epochs' training before crashing, we observe that the loss curve behaves stochasticly and the total recall is less than 1%. We assume it is either not suitable for real-world scene scans or has fundamental implementation issues at the current stage. Due to the absence of valid registrations, we do not include the numbers.

Lastly, we finetune PointNetLK [2] on 3DMatch for 400 epochs. We use a similar procedure as DCP to generate input pairs with 1024 subsampled points. The loss does not converge, and the best recall rate is 1.61%. PointNetLK uses a feature that is globally pooled for each input and regresses the relative pose between objects. However, as inputs from 3DMatch contain multiple objects with small overlaps, we suspect that the globally pooled features do not meet the required resolution for registration and cannot correctly represent the input.

We report detailed results in Table 8.1, Fig. 8.4 and Fig. 8.5, where we demonstrate that these end-to-end methods return low recall in all scenes, and produce higher TE and RE on successful registrations. In conclusion, while working well on object-centric synthetic datasets, current end-to-end registration approaches fail on real-world data, even cannot achieve comparable performance to ICP. Unlike synthetic data, real 3D point cloud pairs contain multiple objects, partial observations, more noise, and cannot guarantee perfect overlap. We assume it could be non-trivial to transfer such networks to real-world registration tasks.

[1] "https://github.com/WangYueFt/prnet, commit ffceaf1.

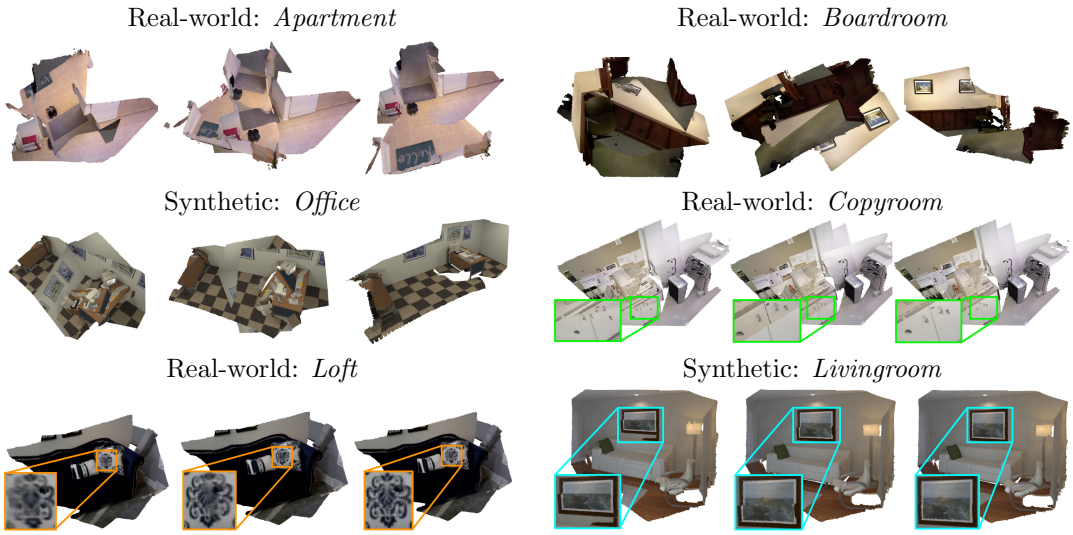

Figure 8.6: Fragment registrations on [6, 24]. From left to right: FGR [43], RANSAC [27], Ours. *Row 1-3*: our method succeeds on scenes with small overlaps or ambiguous geometry structures while other methods fail. *Row 4-6*: by combining Weighted Procrustes and gradient-based refinement, our method outputs more accurate registrations in one pass, leading to better aligned details.

8.5.2 Multi-way Registration

We implement multi-way registration for RGB-D scene reconstruction by modifying the state-of-the-art offline reconstruction pipeline provided by Open3D [44], which is an implementation of [6, 24]. In general, Open3D's pipeline first reconstructs fragments of scenes with a frontend containing RGB-D odometry and TSDF integration modules. It then registers these fragments with a backend where global registration is applied. After pairwise global registration, global pose per fragment is optimized by multi-way registration introduced in [6], where robust pose graph optimization [19] is performed. We reuse the frontend and replace the global registration module with our deep global registration in the backend. All the parameters of our model are trained on the 3DMatch training set and we do not finetune the weights on the test datasets.

We test the modified pipeline on the simulated Augmented ICL-NUIM dataset [6, 14] for quantitative trajectory results, and Indoor LiDAR RGB-D dataset [24] and Stanford RGB-D dataset [6] for qualitative registration visualizations. Note these scenes are unseen during training.

We measure the absolute trajectory error (ATE) on the Augmented ICL-NUIM dataset with simulated depth noise. As shown in Table 8.2, compared to state-of-the-art online SLAM methods [36, 17, 31] and offline reconstruction methods [43], our global registration module guarantees consistent low ATE and is robust to noise.

For qualitative results, we compare pairwise fragment registration on these scenes against FGR and RANSAC in Fig. 8.6. Full scene reconstruction results are in supplementary. Fig. 8.6 shows that

our method can overcome issues caused by small overlaps and ambiguity in structures, thus it can serve as a robust backend for large scale reconstruction. In addition, it shows that in cases when all methods are able to converge to the correct registration at a coarse scale, our method can produce more refined registration without performing additional ICP thanks to the robust pose estimation. This enhances the accuracy of the global pose graph optimization [6] and results in more accurate camera trajectories.

8.6 Conclusion

We propose Deep Global Registration, a learning based framework that aligns real-wolrd 3D scans robustly and accurately. To achieve this, we propose a 6-D convolutional network for robust inlier detection, a differentiable Weighted Procrustes analysis for closed-form pose estimation, and a gradient-based optimizer for pose refinement. Experiments show that our method outperforms all the classical and learning-based registration methods, and can serve as a ready-to-use plugin to replace RANSAC or FGR in off-the-shelf scene reconstruction pipelines.

Bibliography

[1] Dror Aiger, Niloy J Mitra, and Daniel Cohen-Or. 4-points congruent sets for robust pairwise surface registration. In *TOG*, volume 27, page 85, 2008.

[2] Yasuhiro Aoki, Hunter Goforth, Rangaprasad Arun Srivatsan, and Simon Lucey. Pointnetlk: Robust & efficient point cloud registration using pointnet. In *CVPR*, pages 7163–7172, 2019.

[3] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *PAMI*, 14(2):239–256, Feb 1992.

[4] Sofien Bouaziz, Andrea Tagliasacchi, and Mark Pauly. Sparse iterative closest point. In *Eurographics*, pages 113–123. Eurographics Association, 2013.

[5] Qin Cai, David Gallup, Cha Zhang, and Zhengyou Zhang. 3d deformable face tracking with a commodity depth camera. In *ECCV*, pages 229–242. Springer, 2010.

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

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

[8] Angela Dai, Matthias Nießner, Michael Zollöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *TOG*, 2017.

[9] Haowen Deng, Tolga Birdal, and Slobodan Ilic. PPF-FoldNet: Unsupervised learning of rotation invariant 3d local descriptors. In *ECCV*, 2018.

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

[11] Zan Gojcic, Caifa Zhou, Jan Dirk Wegner, and Wieser Andreas. The perfect match: 3D point cloud matching with smoothed densities. In *CVPR*, 2019.

[12] John C Gower. Generalized procrustes analysis. *Psychometrika*, 40(1):33–51, 1975.

[13] Maciej Halber and Thomas Funkhouser. Fine-to-coarse global registration of rgb-d scans. 2017.

[14] A. Handa, T. Whelan, J.B. McDonald, and A.J. Davison. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *ICRA*, Hong Kong, China, May 2014.

[15] Dirk Holz, Alexandru E Ichim, Federico Tombari, Radu B Rusu, and Sven Behnke. Registration with the point cloud library: A modular framework for aligning in 3-d. *RA-L*, 22(4):110–124, 2015.

[16] Andrew E. Johnson and Martial Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *PAMI*, 21(5):433–449, 1999.

[17] Olaf Kähler, V. A. Prisacariu, and David W. Murray. Real-time large-scale dense 3D reconstruction with loop closure. In *ECCV*, pages 500–516, 2016.

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

[19] Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. g2o: A general framework for graph optimization. In *ICRA*, pages 3607–3613. IEEE, 2011.

[20] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *DARPA Image Understanding Workshop*. Vancouver, British Columbia, 1981.

[21] Fahira Afzal Maken, Fabio Ramos, and Lionel Ott. Speeding up iterative closest point using stochastic gradient descent. In *ICRA*, pages 6395–6401. IEEE, 2019.

[22] Nicolas Mellado, Dror Aiger, and Niloy J Mitra. Super 4pcs fast global pointcloud registration via smart indexing. In *Computer Graphics Forum*, volume 33, pages 205–215. Wiley Online Library, 2014.

[23] 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.

[24] J. Park, Q. Zhou, and V. Koltun. Colored point cloud registration revisited. In *ICCV*, pages 143–152, Oct. 2017.

[25] Chen Qian, Xiao Sun, Yichen Wei, Xiaoou Tang, and Jian Sun. Realtime and robust hand tracking from depth. In *CVPR*, pages 1106–1113, 2014.

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

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

[28] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz. Aligning point cloud views using persistent feature histograms. In *IROS*, 2008.

[29] S. Salti, F. Tombari, and L. di Stefano. SHOT: Unique signatures of histograms for surface and texture description. *CVIU*, 125, 2014.

[30] Ruwen Schnabel, Roland Wahl, and Reinhard Klein. Efficient ransac for point-cloud shape detection. In *Computer graphics forum*, volume 26, pages 214–226. Wiley Online Library, 2007.

[31] Thomas Schops, Torsten Sattler, and Marc Pollefeys. BAD SLAM: Bundle adjusted direct RGB-D SLAM. In *CVPR*, pages 134–144, June 2019.

[32] F. Tombari, S. Salti, and L. Di Stefano. Unique shape context for 3d data description. In *ACM Workshop on 3D Object Retrieval*, 2010.

[33] Yue Wang and Justin M. Solomon. Deep closest point: Learning representations for point cloud registration. In *ICCV*, October 2019.

[34] Yue Wang and Justin M. Solomon. Prnet: Self-supervised learning for partial-to-partial registration. In *Conference on Neural Information Processing Systems*, 2019.

[35] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *TOG*, 38(5):146, 2019.

[36] T. Whelan, S. Leutenegger, R. F. Salas-Moreno, B. Glocker, and A. J. Davison. ElasticFusion: Dense SLAM without a pose graph. In *RSS*, Rome, Italy, July 2015.

[37] Jiaolong Yang, Hongdong Li, Dylan Campbell, and Yunde Jia. Go-icp: A globally optimal solution to 3d icp point-set registration. *PAMI*, 38(11):2241–2254, 2015.

[38] Jiaolong Yang, Hongdong Li, and Yunde Jia. Go-icp: Solving 3d registration efficiently and globally optimally. In *ICCV*, pages 1457–1464, 2013.

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

[40] 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.

[41] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *IJCV*, 13(2):119–152, 1994.

[42] Chen Zhao, Zhiguo Cao, Chi Li, Xin Li, and Jiaqi Yang. Nm-net: Mining reliable neighbors for robust feature correspondences. In *CVPR*, 2019.

[43] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Fast global registration. In *European Conference on Computer Vision*, pages 766–782. Springer, 2016.

[44] Qian-Yi Zhou, Jaesik Park, and V. Koltun. Open3D: A modern library for 3d data processing. *CoRR*, abs/1801.09847, 2018.

[45] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *CVPR*, pages 5745–5753, 2019.