Correspondence Matrices are Underrated

Tejas Zodage *[1], Rahul Chakwate[2], Vinit Sarode[1], Rangaprasad Arun Srivatsan[1], and Howie Choset[1]

[1]Carnegie Mellon University
[2]Indian Institute of Technology, Madras
*tzodage@andrew.cmu.edu

Abstract

Point-cloud registration (PCR) is an important task in various applications such as robotic manipulation, augmented and virtual reality, SLAM, etc. PCR is an optimization problem involving minimization over two different types of interdependent variables: transformation parameters and point-to-point correspondences. Recent developments in deep-learning have produced computationally fast approaches for PCR. The loss functions that are optimized in these networks are based on the error in the transformation parameters. We hypothesize that these methods would perform significantly better if they calculated their loss function using correspondence error instead of only using error in transformation parameters. We define correspondence error as a metric based on incorrectly matched point pairs. We provide a fundamental explanation for why this is the case and test our hypothesis by modifying existing methods to use correspondence-based loss instead of transformation-based loss. These experiments show that the modified networks converge faster and register more accurately even at larger misalignment when compared to the original networks.

1. Introduction

Point cloud registration (PCR), the task of finding the alignment between pairs of point clouds is often encountered in several computer vision [15, 2] and robotic applications [1, 33, 19, 22]. A critical aspect of registration is determining a correspondance *i.e.* mapping between points of first cloud to the points of the second. Most approaches
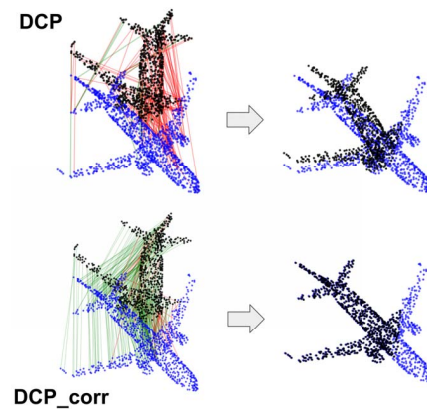
Figure 1. When learning to predict the pose parameters, deep-learning-based methods such as DCP [29] learn correspondence (mapping between the points of point clouds) implicitly. If the same networks are trained to explicitly learn correspondence (DCP_corr), the resulting registration is more accurate. Template point cloud is shown in blue, source point cloud in black. Green arrows show correct correspondence. Red arrows show incorrect correspondence.

to registration (e.g., [4]) use simple rules-of-thumb or implement a separate procedure to establish correspondances. While these approaches have been widely used, they do suffer from computational complexity impacting their performance to determine pose parameters in real-time. Recent developments in deep learning-based registration approaches have resulted in faster and, under some circumstances, more accurate results [2, 21, 29, 9].

Unlike conventional approaches, most deep learning approaches directly estimate the pose and often do not explicitly estimate point correspondences. Instead, they implicitly learn the correspondences while being trained. While exploring the relation between correspondence and registration, we observed that perturbing the correspondence produced only small changes in the final pose estimation when

compared to perturbations in the axis-angle representation of the rotation (see section 3.1 for more details).

Based on this observation, we hypothesize that higher registration accuracy can be achieved by training the networks to explicitly predict point correspondences instead of implicitly learning them. In order to test this hypothesis, we modify the loss function of existing registration approaches and compare the results. To develop a suitable loss function, we come up with a novel way of posing registration as a multi-class classification problem. Wherein, each point in one point cloud is classified as corresponding to a point in the other point cloud. These modified networks predict correspondences, from which pose parameters are then calculated using Horn's method [11]. We show that the performance of each network that we modified is substantially improved (see Fig. 1). Notably, these networks give more accurate registration results when faced with large initial misalignment and are more robust to partial point cloud data as compared to the original networks.

Even though recent learning-based methods such as DCP [29] and RPMNet [36] also calculate correspondence as an intermediate step in order to calculate pose parameters, the networks are not explicitly trained to learn them. We show that their networks can register more accurately when explicitly trained to learn the correspondence.

The key contributions of our work are listed as follows–

- We provide a fundamental reasoning of why explicitly predicting correspondence provides better accuracy and results in faster convergence and verify it through extensive experimentation.
- We introduce a new way of formulating point cloud registration as a multi-class classification problem and develop a suitable loss using point correspondence.

2. Related Work

2.1. Conventional registration methods

Iterative Closest Points (ICP) [4] is one of the most popular methods for point cloud registration. ICP iteratively computes nearest neighbor correspondences and updates transformation parameters by minimizing the least-squares error between the correspondences [11]. Over the years, several variants of ICP have been developed [18]. An important area of research in this space relates to efficient ways of finding correspondences, for example point-to-plane correspondences [6], probabilistic correspondences [22, 5, 13, 14], and feature-based correspondences [17, 27]. These methods are locally optimal and hence perform poorly in the case of large misalignment.

For large misalignment, stochastic optimization techniques have been developed such as genetic algorithms [23], particle swarm optimization [28], particle filtering [20, 25] etc. Another category of methods that deal with large mis-

alignment include globally optimal techniques. A popular approach is the globally optimal ICP (Go-ICP) [35] that uses a branch and bound algorithm to find the pose. Recently, mixed integer programming has been used to optimize a cost function over transformation parameters and correspondences simultaneously [26, 12]. These methods have theoretical guarantees to reach global optima. The fact that they explicitly optimize over correspondences, motivates our work.

2.2. Deep learning-based registration methods

Some of the recent deep-learning based PCR methods train a network to directly predict the transformation between the input point clouds. PointNetLK [2] aligns the point clouds by minimizing the difference between the PointNet [16] feature descriptors of two input point clouds. PCRNet [21], passes the concatenated global feature vectors through a set of fully connected layers to predict the pose parameters. These methods operate on global point cloud features and fail to capture the local geometrical intrinsics of the points.

In order to capture local geometry, Deep Closest Point [29] learns to assign embedding to the points in each point cloud based on its nearest neighbours and attention mechanism. Further based on the similarity between the features, a correspondence matrix is generated which calculates transformation parameters that are used to define the loss function. DCP network architecture is iteratively used by PRNet [30] to align partial point clouds. This idea of using a correspondence predictor iteratively is also used by RPMnet [36], where the network structure uses FGR [38] feature descriptor unlike DGCNN [31] used by DCP and PRNet.

Some other methods such as deep global registration (DGR) [7] and multi view registraiton (MVR) [10], follow a two step process – (1) they find a set of correspondence pairs between two sets of 3D points using fully convolution geometric features (FCGF) [8], and (2) these correspondence pairs are passed through a network which filters outliers. Note that DGR and MVR only find a subset of all possible correspondence pairs, and yet register more accurately than methods that directly predict pose parameters. This observation motivates us to study the effect of explicitly training a network to predict all point-point correspondences. A key difference between the approach taken by MVR and DGR from our approach is that, they take point-pairs and classify them as inlier/outliers, while our approach classifies each point in one set (source) as belonging to one of multiple available classes (points in the other set)

3. Mathematical Formulation

PCR is generally posed as an optimization problem. Consider two point clouds $X = [x_1, x_2, ..., x_{N_x}] \in \mathbb{R}^{3 \times N_x}$
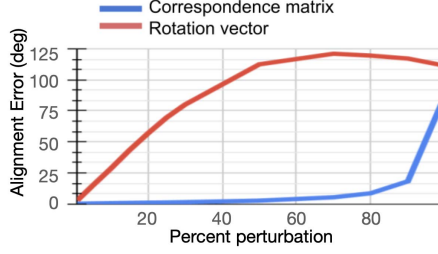
Figure 2. Graph showing percentage of perturbation to 'point-correspondence' and 'rotation vector' vs alignment error. The plot shows that the alignment error is low even with as high as 40% wrong correspondences. On the other hand the alignment error quickly increases with perturbation to the rotation parameters. Thus, we hypothesize that training the networks to learn correspondences would have better registration accuracy than learning pose parameters.

and $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_{N_y}] \in \mathbb{R}^{3 \times N_y}$ containining N_x and N_y points respectively, where $\boldsymbol{x}_i \in \mathbb{R}^3$ and $\boldsymbol{y}_i \in \mathbb{R}^3$ are the points in the respective point clouds and generally, $N_x \neq N_y$. The ground truth transformation, $\boldsymbol{R}^* \in SO(3)$ and $\boldsymbol{t}^* \in \mathbb{R}^3$, that aligns the two point clouds can be represented as

$$\boldsymbol{R}^*, \boldsymbol{t}^* = \underset{\boldsymbol{R}, \boldsymbol{t}}{argmin} \left(\sum_{i=1}^{N_x} ||\boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t} - \boldsymbol{y}_{\pi(\boldsymbol{x}_i)}||_2 \right), \quad (1)$$

where $|| \ldots ||_2$ is the L2 norm and, π denotes a function such that $\pi(\boldsymbol{x}_i)$ is the index of the point corresponding to $\boldsymbol{x}_i$ in $\boldsymbol{Y}$. This function can be represented by a binary matrix known as correspondence matrix $\boldsymbol{C} \in \mathbb{R}^{N_y \times N_x}$, where $C_{i,j} \in \{0, 1\} \ \forall \ i \in [1, \ldots, N_x]$ and $j \in [1, \ldots, N_y]$. $\boldsymbol{C}_{j,i} = 1$ implies that $\boldsymbol{y}_j$ corresponds to $\boldsymbol{x}_i$. Or, $\boldsymbol{y}_j = \boldsymbol{y}_{\pi(\boldsymbol{x}_i)} = \boldsymbol{Y}\boldsymbol{C}_{:,i}$. Here $\boldsymbol{C}_{:,i}$ represents the ith column of C. Since the correspondence is unknown, registration is restated as,

$$\boldsymbol{R}^*, \boldsymbol{t}^*, \boldsymbol{C}^* = \underset{\boldsymbol{R}, \boldsymbol{t}, \boldsymbol{C}}{argmin} \left(\sum_{i=1}^{N_x} ||\boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t} - \boldsymbol{Y}\boldsymbol{C}_{:,i}||_2 \right) \quad (2)$$

Where $\boldsymbol{C}^*$ is the ground truth correspondence matrix. Note that $\hat{\boldsymbol{Y}} = \boldsymbol{Y}\boldsymbol{C}$ denotes the rearranged $\boldsymbol{Y}$ such that i^{th} point of $\boldsymbol{X}$ corresponds to i^{th} point of $\hat{\boldsymbol{Y}}$.

3.1. Robustness of correspondences Vs robustness of transformations

To understand the effect of wrong correspondences on alignment, we perform the following experiment. We initially sample n points randomly from a unit 3D cube and denote it as point cloud $\boldsymbol{X}$. We then transform $\boldsymbol{X}$ with a random but known rotation $\boldsymbol{R}^*$ to create point cloud $\boldsymbol{X}' = \boldsymbol{R}^*\boldsymbol{X}$. For convenience, we convert $\boldsymbol{R}^*$ to a rotation

vector form $\boldsymbol{v}^* \in \mathbb{R}^3$ and add $p\%$ corruption to generate $\boldsymbol{v}^{pert} = \boldsymbol{v}^* + \boldsymbol{v}^{corrupt}$. We then calculate the error between $\boldsymbol{R}^{pert}$ and $\boldsymbol{R}^*$. This is noted as alignment error between corrupted rotation and ground truth rotation. We gradually increase the percentage corruption and calculate the corresponding alignment error (Figure 2). To observe the robustness of the correspondences, in another independent experiment, we randomly corrupt $p\%$ of the ground truth correspondence and calculate the resulting rotation matrix based on perturbed correspondences using Horn's method [3] [11]. The error between the ground truth rotation and perturbed rotation is then calculated. We gradually increase the percentage corruption and observe it's effect on the rotation error. We observe that even if 40% of the correspondences are wrong, the alignment error is $\approx 5°$.

Based on this observation we hypothesize that if a network is trained explicitly to predict correspondence, the network will align point clouds more accurately than a network with similar architecture but trained to predict pose.

4. PCR as multi-class classification

To test our hypothesis, we first develop a suitable loss function that can explicitly learn the correspondences. An obvious choice could be a mean square error or absolute error between predicted and ground truth correspondence but these loss functions do not provide any strong physical intuition about the correspondence.

We introduce a novel way of treating the task of correspondence assignment as a multi-class classification problem. We treat each point in $\boldsymbol{Y}$ to be a different class and each point in $\boldsymbol{X}$ belongs to one of the classes i.e. N_x examples and N_y classes. Note that each example needs to belong to at least one class but there can be classes with no corresponding example. This framework is particularly suitable to register partial point clouds where, $\forall \boldsymbol{x}_i, \exists \boldsymbol{y}_j$ but converse need not be true. Note that this is fundamentally different from MVR [10], where each correspondence pair is classified as a binary: inlier or outlier and the correspondence matrix constraints are not respected.

We consider a general framework that first generates per point features $\boldsymbol{F_X} = [\boldsymbol{f}_{x_1}, \boldsymbol{f}_{x_2}, ..., \boldsymbol{f}_{x_{N_x}}] \in \mathbb{R}^{N_e \times N_x}$ and $\boldsymbol{F_Y} = [\boldsymbol{f}_{y_1}, \boldsymbol{f}_{y_2}, ..., \boldsymbol{f}_{y_{N_y}}] \in \mathbb{R}^{N_e \times N_y}$ for input point clouds $\boldsymbol{X}$ and $\boldsymbol{Y}$ where $\boldsymbol{f}_j \in \mathbb{R}^{N_e \times 1}$ and N_e is the embedding space dimension. We generate a soft correspondence matrix based on a differentiable distance metric in the feature space [4]. The metric can be distance-based as introduced

---

[3]Note that Horn's method is just one of many closed form approaches to obtain transformation given corresponding pairs of point clouds. The results will be identical if Horn's method is replaced by weighted SVD, or Arun's method [3].

[4]The soft correspondence is similar to the matrix used in conventional registration approaches [14, 12, 26, 5], where every element of the correspondence matrix denotes the probability of matching.
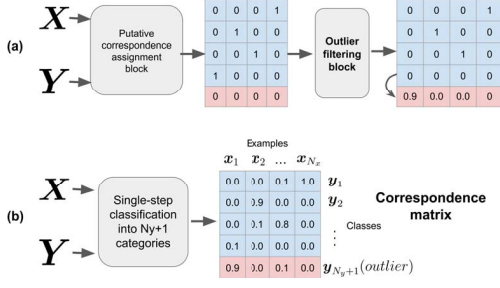
Figure 3. a) Deep Global Registration [7], b) Possible extension of our approach where two stage process of DGR can be merged into a single step thus resulting in a faster registration. Note that in our current work, networks output $C \in \mathbb{R}^{N_y \times N_x}$ we assume that the data can be partial but there are no outliers

by MVR [10] or projection-based as suggested in DCP [29]. Without any loss of generality, we choose DCP's approach to generate a soft correspondence matrix *i.e.* a matrix where each element denotes probability of a correspondence between point pairs as $C = softmax(F_Y^T F_X)$. We compare C with a ground truth correspondence matrix C^*. We define the ground-truth correspondence as nearest neighbor of a point x_i in Y when X and Y are aligned. It is worth noting that this is not a reversible mapping *i.e.* if $y_1 \in Y$ is the nearest neighbor of x_1, it is possible that the nearest neighbor of y_1 in X is x_j, where $j \neq 1$. This adds a constraint on the correspondence matrix that each the sum of the elements of each column should add up to one.

The multi-class classification framework allows us to use a cross-entropy loss, L_{CE}. This loss function implicitly applies the constraint that sum of the elements of a column should be one unlike binary cross entropy (BCE). For the sake of convenience of notation, we define $C' = F_Y^T F_X$

$$L_{CE}(C', C^*) = -\sum_{i=1}^{N_x} log \left(\frac{exp \left(\sum_{j=1}^{N_y} C'_{j,i} C^*_{j,i} \right)}{\sum_{j=1}^{N_y} exp(C'_{j,i})} \right)$$

While beyond the scope of this paper, it is worth noting that our framework can be further modified to add an additional class to classify an x_i as an outlier. If we incorporate an extra class for outliers in the correspondence matrix $C \in \mathbb{R}^{N_y+1 \times N_x}$, this becomes a single-step generalised version of the two step process used by DGR (see Fig. 3).

5. Experimental setup

We consider RPMNet [36], DCP [29], and PCRNet [21] to study the effects of training the network to learn correspondence vs training the network to learn pose parameters.

Note that these methods were originally developed to register point clouds with small ($\pm 45°$) initial misalignment. From here on we follow the notation that *method* is the network trained with loss function suggested in the original paper while *method_corr* is trained using our loss function (cross-entropy on correspondence matrix). We train and test all these *method*s and *method_corr*s on ModelNet40 [32] dataset.

DCP [29] and RPMNet [36] generate an implicit correspondence based on the similarity between per-point features of the input point clouds (Fig. 4). This intermediate correspondence is used to find the transformation parameters between input point clouds using Horn's method [11] and weighted SVD method [36] respectively. For a network to implicitly learn correspondence, we define the loss as a function of the output transformation as suggested by the respective *method*. While to explicitly learn the correspondence, we define the loss as a function of intermediate correspondence as defined in Sec. 4.

We sample n number of points from a point cloud chosen from training data and denote this as point cloud X. We generate a copy of X and shuffle the order of points to generate Y'. To sample a rotation, we randomly choose a unit vector in $\mathbb{R}^3$ and an angle $\theta \in \mathcal{U}(-\theta_0, \theta_0)$, this axis and angle is used to generate a rotation vector which is further transformed into a ground-truth rotation matrix R^*. Here, θ_0 depends upon the specific experiment and $\mathcal{U}(a, b)$ denotes a uniform distribution in the range $[a, b]$. Further we generate a ground-truth translation vector $t^* \in [\mathcal{U}(-0.5, 0.5), \mathcal{U}(-0.5, 0.5), \mathcal{U}(-0.5, 0.5)]$. Now Y' is transformed with R^* and t^* to generate Y. To generate the ground-truth correspondence matrix C^*, we find the nearest neighbour of each $x_i \in X$ in Y'. If $y'_j \in Y$ is the nearest neighbour of x_i then $C^*(j, i)$ is set to 1 and other elements of ith column are set to 0.

To generate the partial point clouds, we randomly choose a plane passing through the centroid of the original point cloud of source (X). We then randomly choose either up or down directtion of the plane and remove a predetermined number of points from the source farthest from the plane.

5.1. DCP Vs DCP_corr

DCP uses DGCNN [31] features along with transformer network-based attention and co-attention mechanism to generate interrelated per point features of a point cloud. These features are used to generate probability distribution of source points on the target points matrix C. They further calculate an intermediate representation of target point cloud Y as $\hat{Y} = CY$. DCP uses Horn's method to estimate the rotation matrix R and translation t which minimizes the distance between corresponding points of $\hat{Y}$ and X. The loss for DCP is defined as

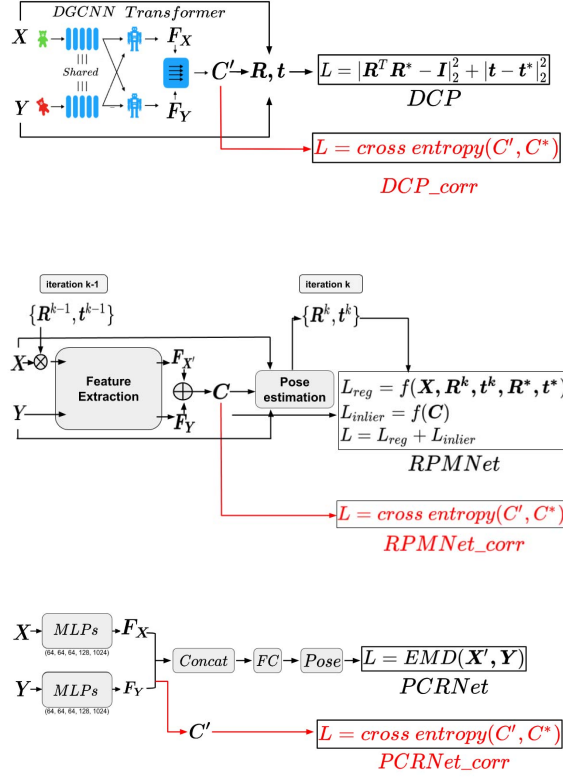$$L_{DCP} = ||R^T R^* - I||_2^2 + ||t - t^*||_2^2 \qquad (3)$$

Figure 4. DCP and RPMNet architctures internally calculate the correspondence matrix C. This correspondence matrix is further used along with X, Y to calculate R, t. In order to make these networks explicitly learn correspondence, we use C along with ground truth C^* to calculate cross entropy loss. Since PCRNet does not explicitly calculate C, we modify the network architecture and compare the PointNet's per-point features to generate the correspondence matrix.

For all the comparisons between DCP and DCP_corr, we use learning rate = 0.001 as recommended by DCP.

DCP_corr uses the correspondence matrix obtained in the intermediate step and compares it with ground truth correspondence using cross entropy

$$L_{DCP\text{-}corr} = cross\ entropy(C, C^*) \qquad (4)$$

5.2. RPMNet vs RPMNet_corr

RPMNet follows an iterative procedure. In each iteration, the point clouds X and Y and transformation from previous iterations are passed into the feature extraction network which computes point-wise features. The extracted features are then used to estimate the correspondence matrix which is further refined using Sinkhorn [24] normalization layer in an unsupervised manner. In order to estimate the transformation parameters, the target points Y are weighted with the correspondence matrix weights C to obtain putative source correspondences $\hat{Y} = YC$. RPMNet_corr uses

this correspondence matrix to define the cross entropy loss (see Fig. 4). RPMNet evaluates transformation parameters R, t based on $X, \hat{Y}$ and C. These transformation parameters are then used to define the primary loss function L_{reg},

$$L_{reg} = \frac{1}{N_x} \sum_{i=1}^{N_x} |(\boldsymbol{R}^*\boldsymbol{x}_i + \boldsymbol{t}^*) - (\boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t})|_1 \qquad (5)$$

RPMNet uses an additional unsupervised loss function L_{inlier} which forces the network to predict majority of the correspondences as inliers. These two loss functions together form $L_{RPMNet} = L_{reg} + L_{inlier}$.

Both RPMNet and RPMNet_corr are trained with the same hyper-parameters (as recommended in [36]), except for the learning rate. RPMNet_corr is trained with an initial learning rate of 0.01 which decays upto 0.0001 during training. We tried a higher learning for both the methods but training of RPMNet is unstable for higher learning rates.

5.3. PCRNet Vs PCRNet_Corr

PCRNet is a correspondence-free network that estimates registration parameters given a pair of input point clouds (X and Y). As shown in Fig. 4, PCRNet uses Point-Net [16] as a backbone to compute the point-wise features of each input point cloud arranged in a siamese architecture. In order to avoid input permutations, a symmetry function (max-pool) is operated on point-wise features to obtain a global feature vector ($\in \mathbb{R}^{1x1024}$). PCRNet concatenates the global feature vectors of both the inputs and uses a set of fully connected layers to regress the registration parameters. Rather than defining the loss function on the ground truth transformation, PCRNet uses chamfer distance (CD) as the loss function,

$$CD(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{N_x} \sum_{\boldsymbol{x}_i \in \boldsymbol{X}} \min_{\boldsymbol{y}_j \in \boldsymbol{Y}} \|\boldsymbol{x}_i - \boldsymbol{y}_j\|_2 +$$
$$\frac{1}{N_y} \sum_{\boldsymbol{y}_j \in \boldsymbol{Y}} \min_{\boldsymbol{x}_i \in \boldsymbol{X}} \|\boldsymbol{y}_j - \boldsymbol{x}_i\|_2 \qquad (6)$$

CD calculates the average closest distance between the template X and the point cloud obtained by applying predicted transformation on Y.

Even though PCRNet uses an unsupervised loss function, CD is a function of X, Y, R and t. In other words, the training of PCRNet again depends on the accuracy of R, t when compared to the ground truth.

6. Results

In this section, we present results of different existing approaches, referred to as $method$, and provide comparisons to versions of those approaches modified by training using our correspondence based loss – referred to as
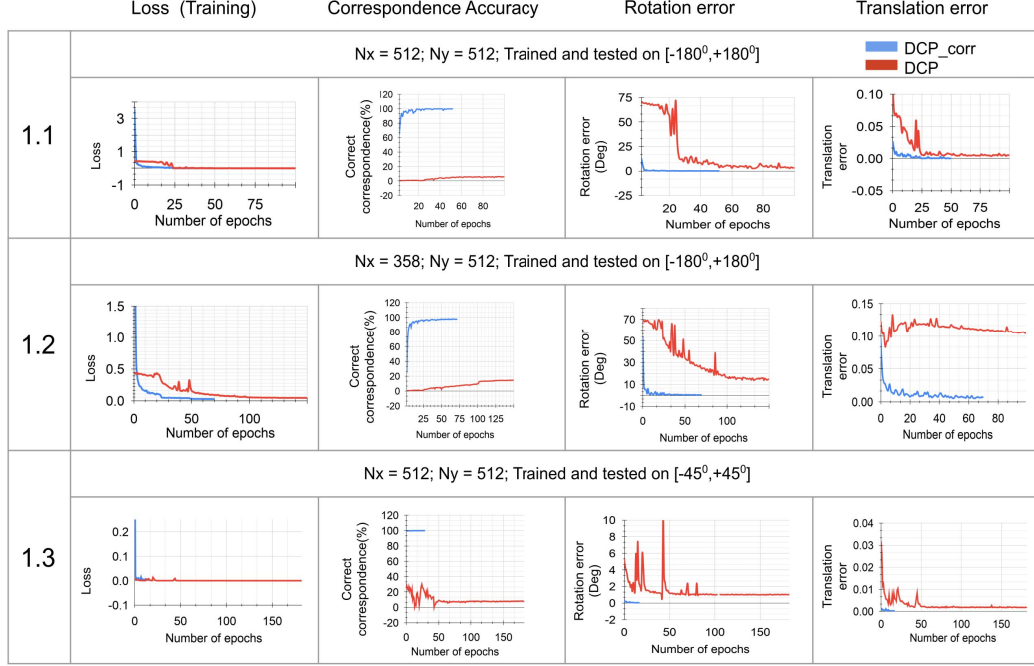
Figure 5. Results of experiments on DCP vs DCP_corr.

method_corr. We specifically highlight the improvement shown by *method_corr* compared to *method* to large initial misalignment errors as well as ability to register partial point-clouds.

Table 1. Effect of initial misalignment on registration accuracy

Rotation range (deg)	Rotation MAE (deg)		Correspondence (%)	
	DCP	DCP_corr	DCP	DCP_corr
0-30	0.99	0.005	8.90	99.99
30-60	1.55	0.008	6.12	99.97
60-90	1.69	0.010	5.78	99.96
90-120	1.56	0.010	5.69	99.96
120-150	1.62	0.010	5.66	99.95
150-180	1.64	0.010	5.60	99.96

6.1. DCP Vs DCP_corr

The authors of DCP, consider 1024 points in all of their experiments. Due to limited GPU space, we re-ran all the DCP experiments using 512 points with the same hyperparameters including learning rate for both. We sample rotations from $SO(3)$ with rotation vectors instead of Euler angles. This helped us to train DCP even for large misalignment. For different experimental settings Fig. 5 shows the comparison between DCP and DCP_corr. The first column shows that every training procedure converged. Second show the accuracy of correspondence estimation of both the methods. Third column shows rotation error as an RMSE over Euler angle error and fourth column denotes translation error.

Experiment 1.1 We have $N_x = 512$ points in the source and $N_y = 512$ points in the target. The initial misalignment between them is uniformly sampled from $SO(3)$ while the translation is bound in cube of unit size centered on origin. As observed in Fig. 5 we can see that DCP_corr converges faster than DCP and is more accurate.

Experiment 1.2 The results of this section are visualized in Fig. 1. We have $N_x = 358$ points in the source and $N_y = 512$ points in the target. The source point cloud is made partial as described in Sec. 5 . We observe that even though DCP's loss function converges, the RMSE rotation error is $14.7°$ while the rotation error of DCP_corr is $0.51°$. This can be considered as an empirical evidence that multi-class classification approach can deal with partial data without any major modification to the network architecture.

Experiment 1.3 In this experiment, we compare DCP to DCP_corr for the specific task DCP was developed for, *i.e.* full-to-full point cloud registration for initial misalignment in the range of $[-45°, +45°]$. We observe that both the networks converge, and the rotation accuracy of DCP and DCP_corr are $1.036°$ and $0.034°$ respectively.

Experiment 1.4 In this experiment we observe the effect of initial misalignment on registration accuracy of DCP and DCP_corr trained for arbitrary initial misalignment (Table 1). For this experiment, we set the translation to zero and only allow a rotational misalignment between the input point clouds. We observe that DCP_corr always registers more accurately than DCP, which is attributed to the

remarkably high percentage of correct correspondence.

6.2. RPMNet Vs RPMNet_corr

We present the comparisons between RPMNet and RPMNet_corr in Fig. 6. Unlike the previous experiment with DCP, the rotation error metric used to evaluate these experiments is the mean absolute anisotropic rotation error also known as axis angle error. We chose this metric to be compliant with the choice of the authors of RPMNet [36]. Likewise, we present the Chamfer distance (CD) between registered point clouds, in the fourth column of Fig. 6, as suggested by the authors of RPMNet [36]. The initial misalignment in translation is sampled uniformly between $[-0.5, 0.5]$.

Experiment 2.1 In this experiment, both the point clouds have $N_x = N_y = 1024$ points. The misalignment between these clouds is uniformly sampled from $SO(3)$. It can be observed that the rotation error converges faster and to a lower value of $0.059°$ with RPMNet_corr as compared to an error of $0.56°$ for RPMNet.

Experiment 2.2 To test the ability of multi-class classification approach to handle partial point clouds, in this experiment we generate the partial source point cloud by retaining 70% of the points above a random plane such that $N_x = 717$ and $N_y = 1024$. We carry out this experiment with uniform sampling from $SO(3)$. Note that, even though one of the key features of RPMNet is the ability to deal with partial point clouds, RPMNet_corr has higher registration accuracy of $0.34°$ compared to $3.79°$ of RPMNet.

Experiment 2.3 RPMNet is specifically designed for $[-45°, +45°]$ initial misalignment. Even in this range, we observe that RPMNet_corr converges faster and registers more accurately. We also observe that eventually RPMNet reaches 96% correspondence accuracy. We believe that the RPMNet's Sinkhorn algorithm along with the unsupervised loss on correspondence (L_{inlier}), pushes the intermediate correspondence matrix towards the ground truth correspondence matrix in an unsupervised manner.

Experiment 2.4 In this experiment we study the effect of initial misalignment on the registration accuracy of RPMNet and RPMNet_corr. Both the networks are trained with arbitrary initial misalignment in the range of $[-180°, +180°]$ between the input point clouds. In this experiment, we set the translation to zero and only allow a rotational misalignment between the input point clouds. We calculate Mean Absolute Error (MAE) between predicted and ground truth rotation in Euler angles. We observe from Table 2 that the MAE for rotation is always lower for RPMNet_corr when compared to RPMNet.

6.3. PCRNet Vs PCRNet_corr

The results showing the comparison between PCRNet and PCRNet_corr are shown in 7. We only provide a ro-

Table 2. Effect of initial misalignment on registration accuracy

Rotation range (deg)	Rotation MAE (deg)		Correspondence (%)		Chamfer distance MSE× 1E-5	
	RPMNet	RPMNet_corr	RPMNet	RPMNet_corr	RPMNet	RPMNet_corr
0-30	0.52	0.011	26.96	98.28	8.51	0.56
30-60	0.58	0.013	26.97	98.28	8.48	0.56
60-90	0.69	0.015	26.96	98.28	8.51	0.55
90-120	0.89	0.26	26.96	98.18	8.49	0.68
120-150	1.01	0.47	26.97	98.15	9.36	0.12
150-180	0.96	0.66	26.97	98.07	8.68	0.79

tational misalignment between the input point clouds.

Experiment 3.1 We consider $N_x = N_y = 1024$ points for both the input point clouds. We train PCRNet with the hyper-parameters recommended in [21] and compare it with PCRNet_corr. Note that PCRNet_corr has fewer tunable parameters than PCRNet due to the removal of MLPs. We observe that both the approaches converge to $\approx 5°$ rotation accuracy. Based on the results of this experiment, we believe that PCRNet lacks depth or number of parameters to achieve higher accuracy. Another reason to believe this is, even after doing a thorough hyper-parameter search, we could not achieve correspondence accuracy of $\geq 70\%$.

Experiment 3.2 In this experiment, we have two point clouds with 100 points each. The initial misalignment between them is in the range of $[-45°, 45°]$. We observe that PCRNet converges to a rotation accuracy of $9.97°$ compared to $1.8°$ of PCRNet_corr.

Experiment 3.3 We repeat the previous experiment but use an initial misalignment that is uniformly sampled from $SO(3)$. We observe that PCRNet_corr outperforms PCRNet and is able to learn correspondences and the rotation accuracy reaches $21°$ at the end of 250 epochs.

7. Conclusion and Future Work

In this paper we demonstrate that higher registration accuracy can be achieved if a network is trained to explicitly learn correspondences instead of learning them implicitly by training on registration parameters. This paper adds to the ever-increasing body of work demonstrating how carefully selecting the desired output of a data-driven approach can lead to drastic improvements in performance. We observe faster convergence, higher registration accuracy and ability to register partial point clouds when networks are explicitly trained to learn correspondence instead of pose parameters. We also developed a new way to approach registration as a multi-class classification task.

While in this work we have limited ourselves to results on ModelNet40, we plan to extend it to real world datasets such as 3DMatch [37] and Sun3d [34]. In addition, future work will involve extended the multi-class classification approach to deal with outliers in the point clouds.
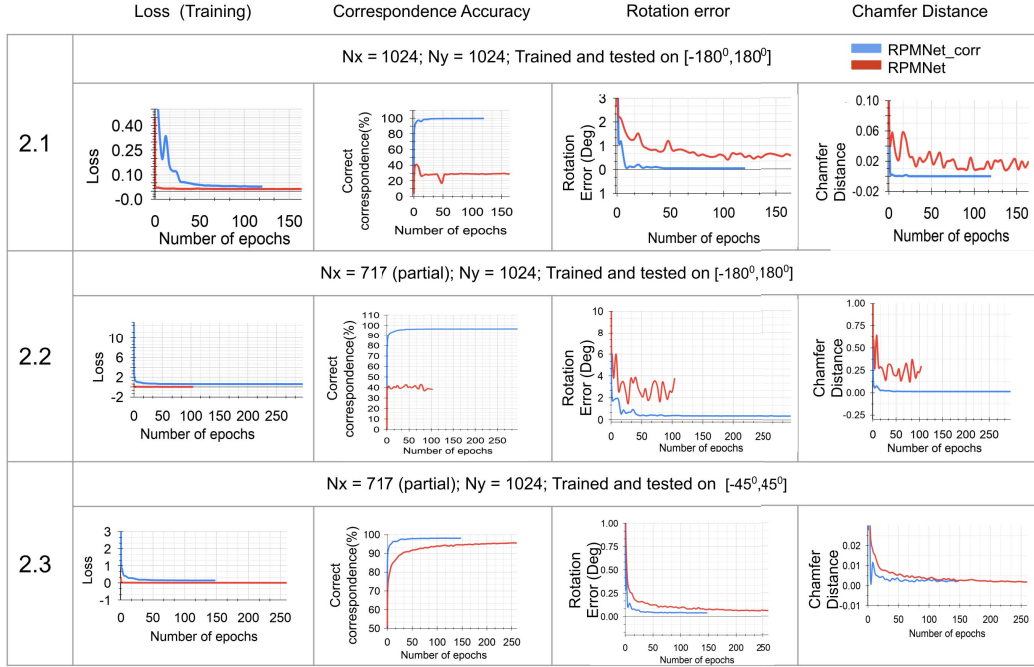
	Loss (Training)	Correspondence Accuracy	Rotation error	Chamfer Distance

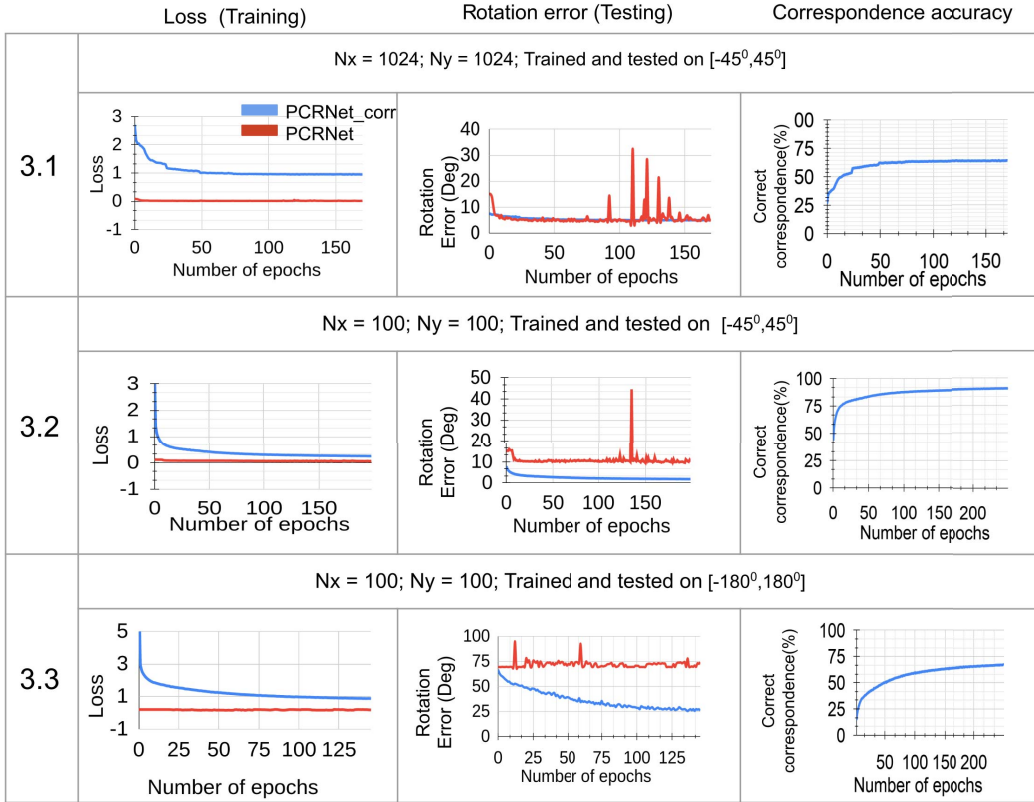Figure 6. Results of experiments on RPMNet vs RPMNet_corr

Figure 7. Results of experiments on PCRNet Vs PCRNet_corr

References

[1] A. Aldoma, F. Tombari, R. B. Rusu, and M. Vincze. Our-cvfh–oriented, unique and repeatable clustered viewpoint feature histogram for object recognition and 6dof pose estimation. In *Joint DAGM (German Association for Pattern Recognition) and OAGM Symposium*, pages 113–122. Springer, 2012. 4321

[2] Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey. Pointnetlk: Robust & efficient point cloud registration using pointnet. *CoRR*, abs/1903.05711, 2019. 4321, 4322

[3] K. Arun, T. Huang, and S. Bolstein. Least-Squares Fitting of Two 3-D Point Sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):698–700, 1987. 4323

[4] P. Besl and N. D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, Feb 1992. 4321, 4322

[5] S. D. Billings, E. M. Boctor, and R. H. Taylor. Iterative Most-Likely Point Registration (IMLP): A Robust Algorithm for Computing Optimal Shape Alignment. *PLoS ONE*, 10, 2015. 4322, 4323

[6] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992. 4322

[7] C. Choy, W. Dong, and V. Koltun. Deep global registration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 4322, 4324

[8] C. Choy, J. Park, and V. Koltun. Fully convolutional geometric features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8958–8966, 2019. 4322

[9] Z. Gojcic, C. Zhou, J. D. Wegner, and W. Andreas. The perfect match: 3d point cloud matching with smoothed densities. In *International conference on computer vision and pattern recognition (CVPR)*, 2019. 4321

[10] Z. Gojcic, C. Zhou, J. D. Wegner, L. J. Guibas, and T. Birdal. Learning multiview 3d point cloud registration, 2020. 4322, 4323, 4324

[11] B. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, 4:629–642, 1987. 4322, 4323, 4324

[12] G. Izatt, H. Dai, and R. Tedrake. Globally optimal object pose estimation in point clouds with mixed-integer programming. In *Robotics Research*, pages 695–710. Springer, 2020. 4322, 4323

[13] L. Maier-Hein, T. R. dos Santos, A. M. Franz, and H.-P. Meinzer. Iterative closest point algorithm in the presence of anisotropic noise. *Bildverarbeitung für die Medizin*, 2010:231–235, 2010. 4322

[14] A. Myronenko and X. Song. Point Set Registration: Coherent Point Drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2262–2275, Dec 2010. 4322, 4323

[15] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011. 4321

[16] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 4322, 4325

[17] A. Rangarajan, H. Chui, and J. S. Duncan. Rigid point feature registration using mutual information. *Medical Image Analysis*, 3(4):425–440, 1999. 4322

[18] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *Proceedings third international conference on 3-D digital imaging and modeling*, pages 145–152. IEEE, 2001. 4322

[19] R. B. Rusu, A. Holzbach, M. Beetz, and G. Bradski. Detecting and segmenting objects for mobile manipulation. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pages 47–54. IEEE. 4321

[20] R. Sandhu, S. Dambreville, and A. Tannenbaum. Point set registration via particle filtering and stochastic dynamics. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1459–1473, 2009. 4322

[21] V. Sarode, X. Li, H. Goforth, Y. Aoki, R. A. Srivatsan, S. Lucey, and H. Choset. PCRNet: Point Cloud Registration Network using PointNet Encoding. *arXiv e-prints*, page arXiv:1908.07906, Aug. 2019. 4321, 4322, 4324, 4327

[22] A. Segal, D. Haehnel, and S. Thrun. Generalized-ICP. In *Robotics: Science and Systems*, volume 2, 2009. 4321, 4322

[23] L. Silva, O. R. P. Bellon, and K. L. Boyer. Precision range image registration using a robust surface interpenetration measure and enhanced genetic algorithms. *IEEE transactions on pattern analysis and machine intelligence*, 27(5):762–776, 2005. 4322

[24] R. Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The annals of mathematical statistics*, 35(2):876–879, 1964. 4325

[25] R. A. Srivatsan and H. Choset. Multiple start branch and prune filtering algorithm for nonconvex optimization. In *The 12th International Workshop on The Algorithmic Foundations of Robotics*. Springer, 2016. 4322

[26] R. A. Srivatsan, T. Zodage, and H. Choset. Globally optimal registration of noisy point clouds. *arXiv preprint arXiv:1908.08162*, 2019. 4322, 4323

[27] I. Stamos and M. Leordeanu. Automated feature-based range registration of urban scenes of large scale. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–Ii. IEEE, 2003. 4322

[28] M. P. Wachowiak, R. Smolíková, Y. Zheng, J. M. Zurada, and A. S. Elmaghraby. An approach to multimodal biomedical image registration utilizing particle swarm optimization. *IEEE Transactions on evolutionary computation*, 8(3):289–301, 2004. 4322

[29] Y. Wang and J. M. Solomon. Deep closest point: Learning representations for point cloud registration. *CoRR*, abs/1905.03304, 2019. 4321, 4322, 4324

[30] Y. Wang and J. M. Solomon. Prnet: Self-supervised learning for partial-to-partial registration. In *Advances in Neural*

Information Processing Systems, pages 8814–8826, 2019. 4322

[31] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 4322, 4324

[32] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 4324

[33] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018. 4321

[34] J. Xiao, A. Owens, and A. 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. 4327

[35] J. Yang, H. Li, and Y. Jia. Go-ICP: Solving 3D Registration Efficiently and Globally Optimally. In *2013 IEEE International Conference on Computer Vision (ICCV)*, pages 1457–1464, Dec 2013. 4322

[36] Z. J. Yew and G. H. Lee. Rpm-net: Robust point matching using learned features. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 4322, 4324, 4325, 4327

[37] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1802–1811, 2017. 4327

[38] Q.-Y. Zhou, J. Park, and V. Koltun. Fast global registration. In *European Conference on Computer Vision*, pages 766–782. Springer, 2016. 4322