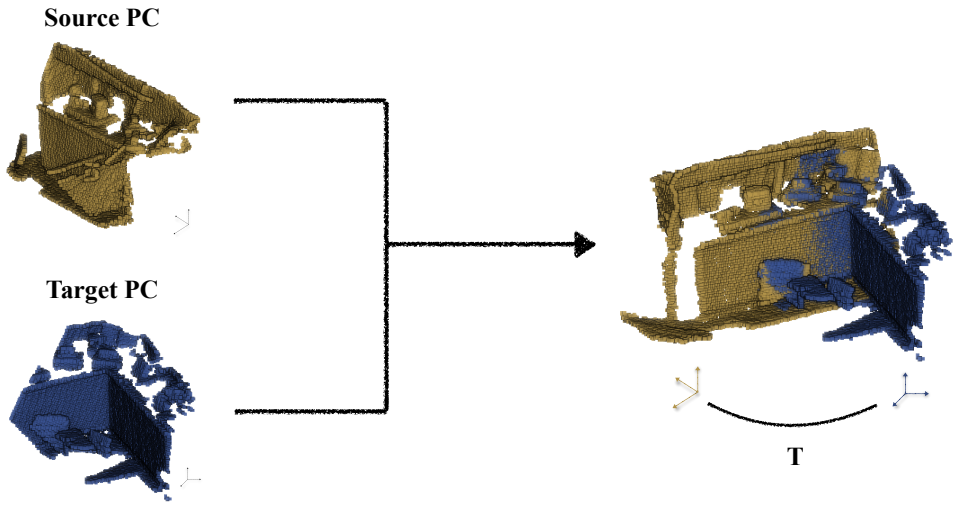
Q-REG: End-to-End Trainable Point Cloud Registration with Surface Curvature

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Q-REG: End-to-End Trainable Point Cloud Registration with Surface Curvature



Master Thesis CVG Lab
Fall Semester 2022

Shengze Jin

Advisors: Dr. Daniel Bela Barath, Dr. Iro Armeni
Professor: Prof. Marc Pollefeys

Abstract

Point cloud registration has seen recent success with several learning-based methods that focus on correspondence matching, and as such, optimize only for this objective. Following the learning step, they evaluate the estimated rigid transformation with a robust estimator, which alternates between the hypothesis set proposal and selection based on the sample consensus. While it is an indispensable component of these methods, it prevents the fully end-to-end training, leaving the objective to minimize the pose error non-served. I present a novel solution, *Q-REG*, which utilizes rich geometric information to estimate the rigid pose from a single correspondence. *Q-REG* allows to formalize the robust estimation as an exhaustive search, hence enabling end-to-end training that optimizes over both objectives of correspondence matching and final pose. I demonstrate in the experiments that *Q-REG* is agnostic to the correspondence matching method and provides consistent improvement both when used only in inference and in end-to-end training. It sets a new state-of-the-art on both the real, indoor point cloud datasets *3DMatch* and *3DLoMatch*, and the synthetic, object-centric datasets *ModelNet* and *ModelLoNet*. It yields an average increase in RR of 2.7% and 7.5% on the *3DMatch* and *3DLoMatch* benchmarks, respectively.

Acknowledgments

Taking this opportunity, I want to express my great thankfulness to all the people who selflessly and continuously gave me guidance, help and support during my master's thesis. First, I would like to thank Prof. Dr. Marc Pollefeys for giving me this precious chance to join his lab and work on the thesis. Secondly, I also want to sincerely thank my advisors, Dr. Daniel Bela Barath and Dr. Iro Armeni. Thank you for providing me with this interesting topic, and for your constant help and guidance. All of our discussions are very fruitful, from which I can benefit greatly. Especially, I want to thank Iro for her patience and generous assistance in helping me submit the CVPR paper. Her cheerful encouragement fuels me to pursue the best. I also want to thank Daniel for his great help in discussing mathematics questions. His optimistic personality and good sense of humor are also valuable to me, which always provide me with an easy and joyful atmosphere. The time we spent together is no wonder an unforgettable memory for me. Lastly, I would like to express my appreciation to my parents, to whom I can never say enough thanks. Their unconditional support and love give me the braveness to explore wherever I want without fear.

Contents

List of Figures

List of Tables

1 Introduction

Point cloud registration is the task of estimating the rigid transformation that aligns two partially overlapping point clouds. It is a key task in point cloud processing and has extensive applications in autonomous driving [1], motion estimation and 3D reconstruction [2], object detection and pose estimation [3, 4], robotic manipulation [5], simultaneous localization and mapping (SLAM) [6, 7], panorama stitching [8], virtual and augmented reality [9], and medical imaging [10]. It is commonly solved by establishing a set of tentative correspondences between the two point clouds, followed by estimating their rigid transformation from the correspondences. The field has seen substantial progress in recent years with methods that introduce a learning component to solve the task.

Most learning methods focus on solving the correspondence task [11, 12, 13, 14], where a feature extractor is trained to extract point correspondences between two input point clouds. Once the learning step is over, they use the estimated correspondences for computing the rigid pose. Due to the low inlier ratio in putative correspondences, these methods strongly rely on hypothesize-and-verify frameworks, *e.g.* RANSAC [15] to compute the pose in a robust manner. Recent methods [16, 17] employ advances in the field of transformers to improve the final estimated set of correspondences and remove the dependency on RANSAC, achieving close-to-RANSAC performance. However, in these methods too, the objective in the learning process remains to find the best and cleanest matches, ignoring the objective to estimate the pose. In addition, they do not achieve end-to-end differentiable training since they still employ robust estimation (*e.g.*, [14, 16]) combined with the Kabsch-Umeyama algorithm [17].

Other learning-based methods, such as [18, 19, 20], directly solve the registration problem by incorporating the pose estimation in their training pipeline. Since RANSAC is non-differentiable due to the random sampling, they choose to estimate the alignment using soft correspondences that are computed from local feature similarity scores. In contrast to these methods, I employ the aforementioned works on estimating hard correspondences and develop a robust solution to replace RANSAC, and allow for end-to-end differentiable training.

In general, RANSAC-like robust estimation is non-differentiable only due to the employed randomized sampling function. Such a sampler is essential to cope with the combinatorics of the problem via selecting random subsets of m correspondences (*e.g.*, $m = 3$ for rigid pose estimation). This allows to progressively explore the $\binom{n}{m}$ possible combinations, where n is the total number of matches. Actually testing all of them is unbearably expensive in practice, which is what methods like [16, 17] try to avoid. This computation bottleneck would be resolved if $m = 1$. Hence, I design a 1-point solution, *Q-REG*, that utilizes rich geometric cues extracted from local surface patches estimated from observed points (Figure 1.1). Specifically, it utilizes rich geometric information by fitting quadrics (*e.g.*, an ellipsoid) locally to the neighborhoods of an estimated correspondence. Moreover, such a solution allows quick outlier rejection by filtering degenerate surfaces and rigid poses inconsistent with motion priors (*e.g.*, unrealistically large scaling). *Q-REG* is designed to be deterministic, differentiable, and it replaces RANSAC for point cloud registration. It can be used together with any feature-matching or correspondence-matching method.

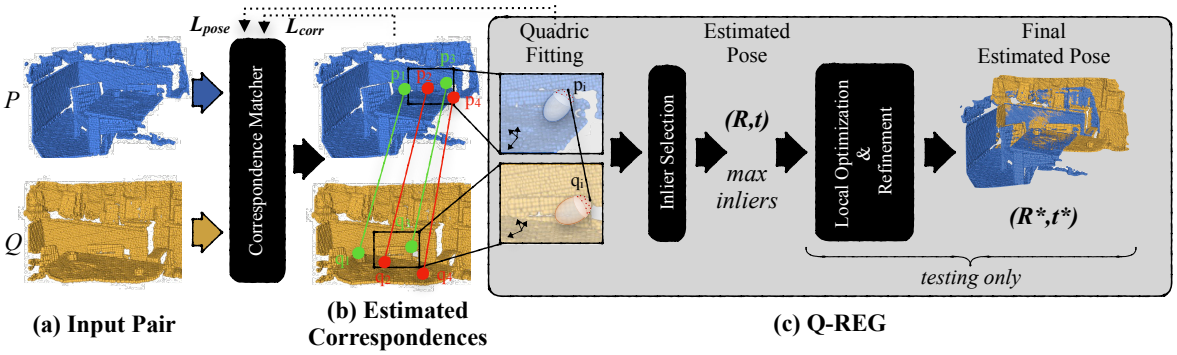


Figure 1.1: *Q-REG* solver. Given (a) two partially overlapping point clouds as input and (b) the estimated correspondences of a matching method, (c) *Q-REG* leverages the rich local geometry to estimate the rigid pose from a single correspondence, hence enabling end-to-end training of the matcher. *(Best viewed on screen.)*

Since *Q-REG* is fully differentiable, it achieves end-to-end training that optimizes both the correspondence matching and final pose objectives. As such, any learning-based matcher can be extended to being end-to-end trainable. In Chapter 5, I demonstrate how *Q-REG* consistently improves the performance of state-of-the-art matchers on the *3DMatch* [21] and *ModelNet* [22] datasets. It sets new state-of-the-art results on both benchmarks.

2 Related Work

2.1 Correspondence-based Registration Methods

The field of 3D point cloud registration is well-established and active. Approaches can be grouped into two main categories: *feature-based* and *end-to-end* registration. Feature-based methods comprise two steps: local feature extraction and pose estimation using robust estimators, like RANSAC [15]. An example pipeline is shown in Figure 2.1. Traditional methods use hand-crafted features [23, 24, 25, 26, 27] to capture local geometry and, while having good generalization abilities across scenes, they often lack robustness against occlusions. Learned local features have taken over in the past few years and, instead of using heuristics, they rely on deep models and metric learning [28, 29] to extract dataset-specific discriminative local descriptors. Depending on the input, these learned descriptors can be divided into patch-based and fully convolutional methods. Patch-based methods [30, 31] treat each point independently, while fully convolutional methods [13, 12, 14] can extract all local descriptors for the whole scene in a single forward pass. The following robust estimators they use are usually non-differentiable due to the employed randomized sampling function. Thus, the correspondence matching network cannot be optimized over the rigid transformation and there is a large gap between the correspondence matching and rigid pose estimation, which greatly degrades the final registration performance.

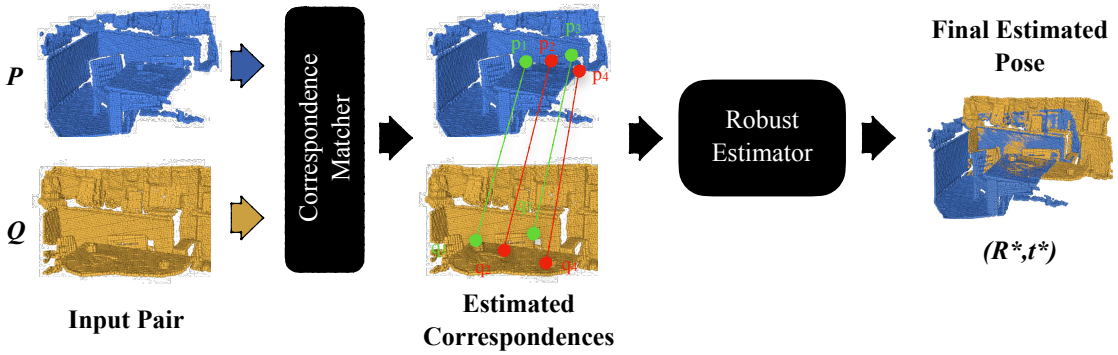


Figure 2.1: Correspondence-based Registration Methods.

2.2 Direct Registration Methods

Recently, end-to-end registration methods have appeared that replace RANSAC with a differentiable optimization algorithm that targets to incorporate direct supervision from ground truth poses. The majority of these methods [18, 20, 19] use a weighted Kabsch solver [32] for final pose estimation. You can find an example pipeline in Figure 2.2. Deep Closest

Point (DCP) [18] iteratively computes soft correspondences based on features extracted by a dynamic graph convolutional neural network [33], which are then used in the Kabsch algorithm to estimate the transformation parameters. To handle partially overlapping point clouds, methods relax the one-to-one correspondence constraint with keypoint detection [20] or optimal transport layers [19, 34]. Li et al. [35] use analytical Jacobians to incorporate information from both the feature and Euclidean space into the pairwise point-matching process. Cao et al. [36] identifies reliable correspondences by leveraging both local geometry and global context through the multiplication of the cross-attention layers. Yuan et al. [37] follow a probabilistic registration approach and learn to compute point-to-distribution correspondences. Another line of work replaces local with global feature vectors that are used to regress the pose. PointNetLK [38] registers two point clouds by minimizing the distance of their feature vectors in the latent space, in an iterative fashion that resembles the Lucas-Kanade algorithm [39]. In [40], an approach is proposed for rejecting non-overlapping regions via masking on the global feature vector. However, due to the weak feature extractors, there is still a large performance gap compared to hard matching methods. These direct registration methods mostly work on single synthetic shape datasets [22] and often fail in large-scale scenes [14]. *Q-REG* uses *hard* correspondences while still being differentiable, via introducing an additional loss component that minimizes the relative pose error. In addition, as demonstrated in Chapter 5, it works for both real-world large-scale scene point clouds [21] and the synthetic shape datasets [22] and sets a new state-of-the-art.

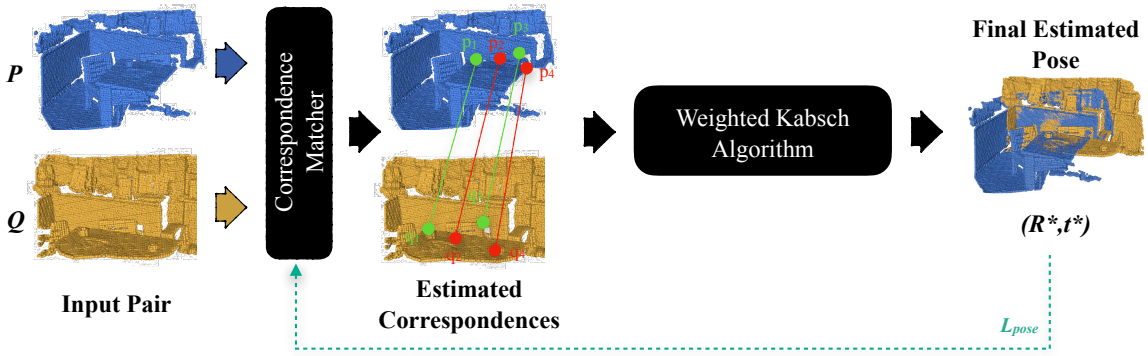


Figure 2.2: Direct Registration Methods.

2.3 Learned Robust Estimators

To address the fact that RANSAC is non-differentiable, other methods either modify it [41] or learn to filter outliers followed by a hypothesize-and-verify framework [42] or a weighted Kabsch optimization [43, 44, 45]. In the latter case, outliers are filtered by a dedicated network, which infers the correspondence weights to be used in the weighted Kabsch algorithm. Similarly, I employ the correspondence confidence predicted by a feature extraction network (e.g, by [14, 16, 17]) as weights in the pose-induced loss. I will summarize the aforementioned research questions in the next chapter and present our solutions to them in Chapter 4.

3 Research Questions

The aim of this thesis is to develop a deterministic and differentiable method leveraging hard correspondences to replace traditional robust estimators for point cloud registration. There are three research questions that need to be answered:

1. How can we design a method estimating the rigid pose from correspondences without a randomized sampling function but using hard correspondences instead of soft ones?
2. How can we reduce the number of combinations without reducing the quality of each solution?
3. How can we design a differentiable pose estimator and a robust pose loss to guide the optimization?

4 Q-REG: Point Cloud Registration with Quadrics

In this chapter, I will first describe the definition of point cloud registration problem (see Figure 4.1). Then, I will introduce ways of extracting local surface patches that can be exploited for point cloud registration.

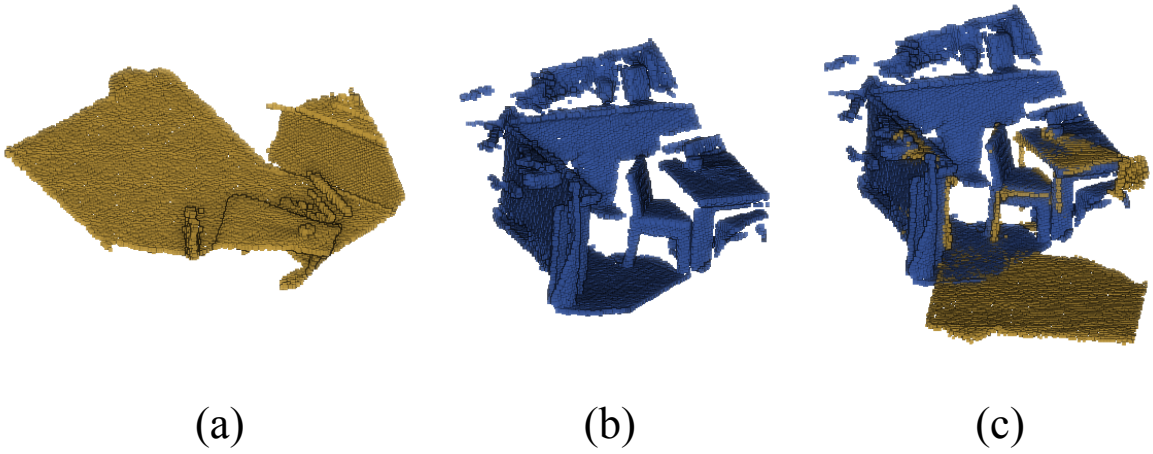


Figure 4.1: Points clouds (a) and (b) and their registration (c).

Suppose that we are given two 3D point clouds $\mathcal{P} = \{\mathbf{p}_i \in \mathbb{R}^3 \mid i = 1, ..., N\}$ and $\mathcal{Q} = \{\mathbf{q}_i \in \mathbb{R}^3 \mid i = 1, ..., M\}$, and a set of 3D-3D point correspondences $\mathcal{C} = \{(\tilde{\mathbf{p}}_i, \tilde{\mathbf{q}}_i) \mid \tilde{\mathbf{p}}_i \in \mathcal{P}, \tilde{\mathbf{q}}_i \in \mathcal{Q},\ i \in [1, K]\}$ extracted, *e.g.*, by the state-of-the-art matchers [14, 16, 17]. The objective is to estimate rigid transformation $\mathbf{T} = \{\mathbf{R}, \mathbf{t}\}$ that aligns two point clouds as follows:

$$\min_{\mathbf{R},\mathbf{t}} \sum\nolimits_{(\mathbf{p}_x^*, \mathbf{q}_y^*) \in \mathcal{C}^*} \|\mathbf{R}\mathbf{p}_x^* + \mathbf{t} - \mathbf{q}_y^*\|_2^2, \tag{4.1}$$

where $\mathbf{R} \in \mathrm{SO}(3)$ is a 3D rotation and $\mathbf{t} \in \mathbb{R}^3$ is a translation vector, and $\mathcal{C}^*$ is the set of ground truth correspondences between $\mathcal{P}$ and $\mathcal{Q}$. In practice, we use the putative correspondences instead of the ground truth correspondences and the set of correspondences often contains a large number of incorrect matches, *i.e.*, outliers. Therefore, the objective is formulated as

$$\min_{\mathbf{R},\mathbf{t}} \sum\nolimits_{(\mathbf{p}_x, \mathbf{q}_y) \in \mathcal{C}} \rho(\|\mathbf{R}\mathbf{p}_x + \mathbf{t} - \mathbf{q}_y\|_2^2), \tag{4.2}$$

where $\rho : \mathbb{R} \rightarrow \mathbb{R}$ is a robust loss, *e.g.*, Huber loss. The problem is solved by a RANSAC-like [15] hypothesize-and-verify framework combined with the Kabsch-Umeyama algorithm [32]. I will present in the next sections that, when employing higher-order geometric information, RANSAC can be replaced by exhaustive search, with the improvement of both the performance and run-time. Figure 4.2 illustrates the developed approach, called *Q-REG*.

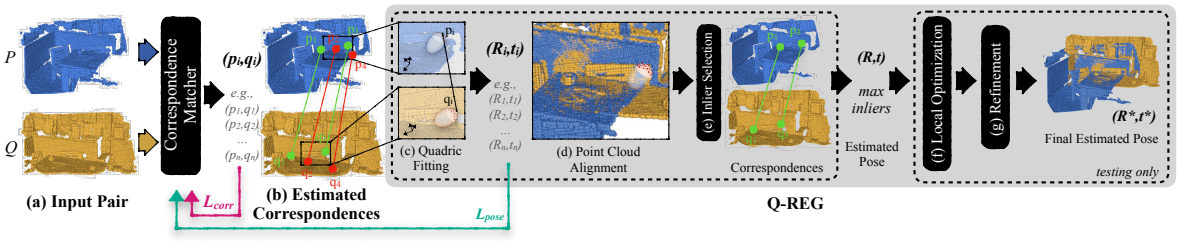


Figure 4.2: Overview of *Q-REG*. During inference, given (a) an input pair of partially overlapping point clouds and (b) the output of a correspondence matcher, I (c) perform quadric fitting for each estimated correspondence from which (d) I estimate the rigid pose and (e) compute the inliers given this pose. I iterate over all estimated correspondences, and choose the estimated pose that yields the most inliers. I further improve the result with (f) the local optimization and (g) the refinement step and output the final estimated pose. During training, I back-propagate the gradients to the correspondence matcher and, in addition to its standard loss formulation, I minimize the proposed loss (L_{pose}) based on the single-correspondence pose estimation. *(Best viewed on screen.)*

4.1 Local Surface Patches

The main goal in this section is to determine a pair of local coordinate systems $(\mathbf{R_p}, \mathbf{R_q})$ for each correspondence $(\mathbf{p}, \mathbf{q}) \in \mathcal{C}$, where $\mathbf{R_p}$, $\mathbf{R_q} \in \text{SO}(3)$. These coordinate systems will be then used to determine rotation $\mathbf{R}$ between the point clouds as $\mathbf{R} = \mathbf{R_q}\mathbf{R_p}^{\text{T}}$. I will present the method for calculating $\mathbf{R_p}$. It is the same for $\mathbf{R_q}$. Note that determining translation $\mathbf{t}$ is straightforward as $\mathbf{t} = \mathbf{q} - \mathbf{p}$.

Suppose that we are given a point $\mathbf{p} \in \mathcal{P}$ and its k-nearest-neighbors $\mathcal{N} \subseteq \mathcal{P}$ such that there exists a correspondence $(\mathbf{p}, \mathbf{q}) \in \mathcal{C}$, $k \in \mathbb{N}^+$. In practice, there are two widely used techniques for estimating local surfaces and determining a local coordinate system. First, one can fit a general quadratic surface to the given point and the points in $\mathcal{N}$ and calculate the normal curvature via the first and second-order derivatives at point $\mathbf{p}$. Since there are infinitely many directions that travel through a point on the surface, there are also infinitely many normal curvatures at a given point $\mathbf{p}$. In differential geometry, the minimum normal curvature c_1 and the maximum curvature c_2 at point $\mathbf{p}$ are defined as the principal curvatures, where $c_1 \leq c_2$, and the directions of the principal curvatures are the smoothest and steepest directions on the surface. These principal directions can give us a local coordinate system that is invariant to the translation and rotation of the local surface. Even though this algorithm is widely used in practice, it can suffer from degenerate cases and slow computation time. To address these limitations, I developed the following approach inspired by [46].

The approach is based on fitting a local quadric, *e.g.*ellipsoid, to the point $\mathbf{p}$ and the points in $\mathcal{N}$. See Figure 4.3 for some examples. The general constraint that a 3D quadric surface imposes on a 3D homogeneous point $\hat{\mathbf{p}} = (x, y, z, 1) \in \mathcal{N}$ lying on the quadric surface is

$$\hat{\mathbf{p}}^{\text{T}}\mathbf{Q}\hat{\mathbf{p}} = 0, \tag{4.3}$$

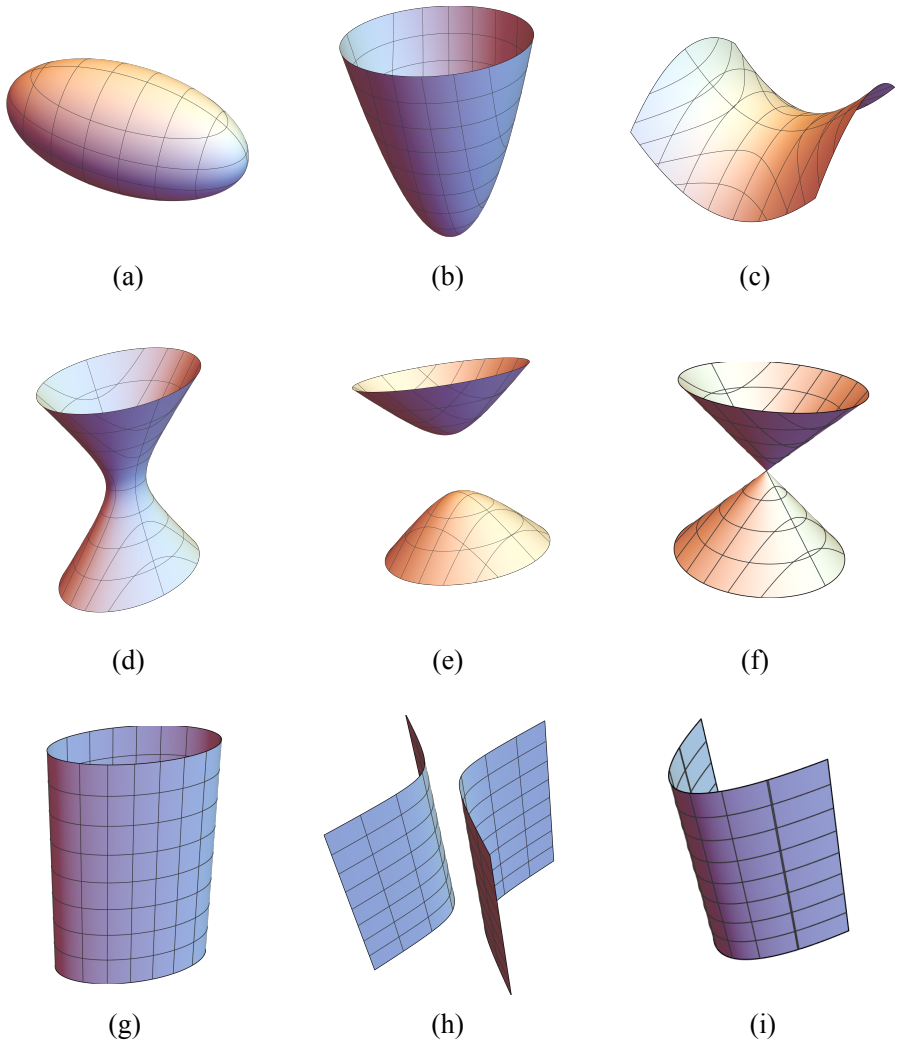


Figure 4.3: Examples of quadrics. (a) Ellipsoid; (b) Elliptic paraboloid; (c) Hyperbolic paraboloid; (d) Hyperbolic hyperboloid; (e) Elliptic hyperboloid; (f) Conical quadric; (g) Elliptic cylinder; (h) Hyperbolic cylinder; (i) Parabolic cylinder

where $\mathbf{Q}$ are the quadric parameters in the matrix form as follows:

$$\mathbf{Q} = \begin{pmatrix} A & D & E & G \\ D & B & F & H \\ E & F & C & I \\ G & H & I & J \end{pmatrix}. \tag{4.4}$$

We can rewrite constraint (4.3) into the form $\mathbf{k}^{\mathrm{T}}\mathbf{w} = d$, where

$$\begin{aligned} \mathbf{k}^{\mathrm{T}} &= (x^2+y^2-2z^2, x^2+z^2-2y^2, 2x^2y^2, \\ &\quad 2x^2z^2, 2y^2z^2, 2x^2, 2y^2, 2z^2, 1), \\ \mathbf{w}^{\mathrm{T}} &= (A', B', D, E, F, G, H, I, J), \\ d &= x^2+y^2+z^2, \\ A' &= \frac{2A+B}{3}, \\ B' &= \frac{A-B}{3}. \end{aligned}$$

By imposing constraints to all the points, we have

$$\sum_{i=1}^{l} \mathbf{k}_i {\mathbf{k}_i}^{\mathrm{T}} \mathbf{w} = \sum_{i=1}^{l} \mathbf{k}_i d_i. \tag{4.5}$$

By solving the above linear equation, we can get the coefficients of the quadric surface $\mathbf{Q}$.

As we are interested in finding $\mathbf{Q}$ such that the observed point $\mathbf{p}$ is ensured to lie on its surface, I substitute J with the formula of p by introducing the constraint

$$\mathbf{p}^{\mathrm{T}}\mathbf{Q}\mathbf{p} = 0. \tag{4.6}$$

In order to find a local coordinate system, I introduce the coefficient matrix

$$\mathbf{P} = \frac{1}{J}\begin{pmatrix} A & D & E \\ D & B & F \\ E & F & C \end{pmatrix}. \tag{4.7}$$

The matrix $\mathbf{P}$ can be decomposed into $\mathbf{I} = \mathbf{V}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}}$, where $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$, which projects the fitted points to the canonical coordinate, and $\mathbf{\Sigma} = \mathrm{diag}(l_1, l_2, l_3)$, which is the diagonal matrix with corresponding eigenvalues.

The matrix $\mathbf{V}$ contains the three main axes that project the quadric, fitted to point $\mathbf{p}$ and its local neighborhood $\mathcal{N}$, to canonical form. Due to its local nature, it is invariant to translation and rotation. Thus, it is a repeatable feature under different rotations and translations of the underlying 3D point cloud. $\mathbf{\Sigma}$ contains the three eigenvalues that are proportional to the reciprocals of the lengths of three axes squared.

4.2 Rigid Transformation from Surface Matches

Suppose that we are given sets of local coordinate systems $\mathcal{V}^{\mathcal{P}}$ and $\mathcal{V}^{\mathcal{Q}}$ associated with points on the found 3D-3D point correspondences. Given correspondence $(\mathbf{p}, \mathbf{q}) \in \mathcal{C}$, we know the

local coordinates systems $\mathbf{V}_{\mathbf{p}}^{\mathcal{P}} \in \mathcal{V}^{\mathcal{P}}$ and $\mathbf{V}_{\mathbf{q}}^{\mathcal{Q}} \in \mathcal{V}^{\mathcal{Q}}$ at, respectively, points $\mathbf{p}$ and $\mathbf{q}$. Due to the local surfaces being translation and rotation invariant, the coordinate systems must preserve the rigid transformation applied to the entire point cloud. Thus, $\mathbf{R} = \mathbf{V}_{\mathbf{q}}^{\mathcal{Q}}\mathbf{P}(\mathbf{V}_{\mathbf{p}}^{\mathcal{P}})^{\mathrm{T}} \in \mathrm{SO}(3)$ is the rotation between the point clouds, where $\mathbf{P}$ is an unknown permutation matrix assigning the axes in the first coordinate system to the axes in the second one.

There are three cases that have to be taken into account. Examples of ellipsoids are shown in Figure 4.4 for better illustration. Ideally, the lengths of the three axes $\mathbf{L}^a = (l_1^a, l_2^a, l_3^a)^{\mathrm{T}}$ have a distinct ordering such that $l_1^a > l_2^a > l_3^a$, $a \in \{\mathcal{P}, \mathcal{Q}\}$, as shown in Figure 4.4 (a). In this case, the permutation matrix can be determined such that it assigns the longest axis in $\mathbf{V}_{\mathbf{q}}^{\mathcal{Q}}$ to the longest one in $\mathbf{V}_{\mathbf{p}}^{\mathcal{P}}$, and so on. This procedure builds on the assumption that there is no or negligible anisotropic scaling in the point clouds and thus, the relative axes lengths remain unchanged. Also, having this assignment allows us to do the matching in a scale-invariant manner while enabling us to calculate a uniform scaling of the point clouds – or to early reject incorrect matches that imply unrealistic scaling. In this case, the problem is solved from a single correspondence.

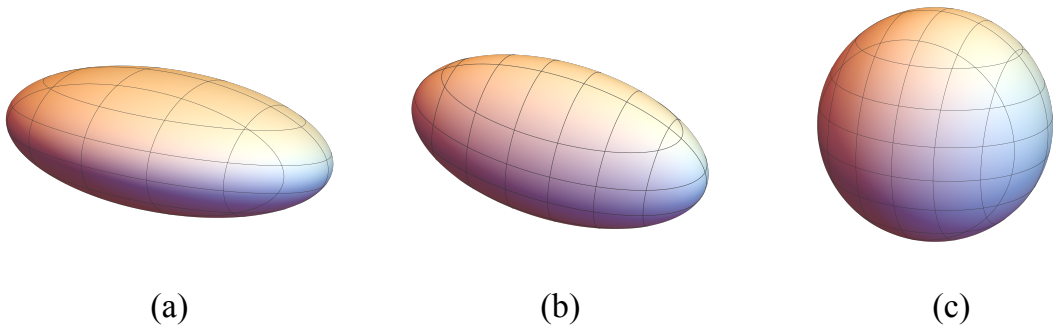


Figure 4.4: Examples of ellipsoids. (a) Ellipsoid; (b) Ellipsoid with two equal axes; (c) Sphere

The second case in Figure 4.4 (b) is when two axes have the same lengths, *e.g.*, $l_1^a \approx l_2^a$, and l_3^a is either shorter or longer than them. In this scenario, only l_3^a can be matched between the point clouds. This case is equivalent to having a corresponding oriented point pair. It gives us an additional constraint for estimating the rotation matrix. However, the rotation around axis l_3^a is unknown and has to be estimated from another point correspondence. While this is a useful solution to reduce the required number of points from three to two, it does not allow solving from a single correspondence.

In the third case as shown in Figure 4.4 (c), when $l_1^a \approx l_2^a \approx l_3^a$, we are given a pair of corresponding spheres that provide no extra constraints on the unknown rotation.

In the proposed algorithm, I keep only those correspondences from $\mathcal{C}$ where the local surface patches are of the first type – i.e., they lead to enough constraints to estimate the rigid transformation from a single correspondence. In the next session, I will discuss how this approach can be used for training 3D-3D correspondence matching algorithms with robust estimation in an end-to-end manner.

4.3 End-to-End Training

Benefiting from the rich geometric information extracted via local surfaces (as described in the previous section), the presented solver is able to estimate the rigid pose from a single 3D quadric correspondence. This unlocks end-to-end differentiability, where the gradients of the feature matcher network can be propagated through the robust estimator to a loss directly measuring the pose error per correspondence. This enables using test-time evaluation metrics to optimize the end-to-end training.

Loss. In order to calculate a pose-induced loss from each correspondence, I first fit quadrics to local neighborhoods. This step has to be done only once, prior to the loss calculation, as the point clouds do not change. Suppose that we are given a set of correspondences $\mathcal{C} = \{(\mathbf{p}, \mathbf{q}, \mathbf{V}_\mathbf{p}^\mathcal{P}, \mathbf{V}_\mathbf{q}^\mathcal{Q}) \mid \mathbf{p} \in \mathcal{P},\ \mathbf{q} \in \mathcal{Q},\ \mathbf{V}_\mathbf{p}^\mathcal{P} \in \mathcal{V}^\mathcal{P},\ \mathbf{V}_\mathbf{q}^\mathcal{Q} \in \mathcal{V}^\mathcal{Q}\}$ equipped with their local quadrics and a solver $\phi : \mathcal{P} \times \mathcal{Q} \times \mathcal{V}^\mathcal{P} \times \mathcal{V}^\mathcal{Q} \rightarrow \mathrm{SE}(3)$, as described in Section 4.2, we can estimate the rigid transformation $\mathbf{T} = (\mathbf{R}, \mathbf{t}) \in \mathrm{SE}(3)$ from a single correspondence. Given a correspondence $(\mathbf{p}, \mathbf{q}, \mathbf{V}_\mathbf{p}^\mathcal{P}, \mathbf{V}_\mathbf{q}^\mathcal{Q})$ and the pose estimated from it $\mathbf{T}_{\mathbf{p},\mathbf{q}} = \phi(\mathbf{p}, \mathbf{q}, \mathbf{V}_\mathbf{p}^\mathcal{P}, \mathbf{V}_\mathbf{q}^\mathcal{Q})$, The error is formalized as follows:

$$\epsilon(\mathbf{T}_{\mathbf{p},\mathbf{q}}) = \sqrt{\frac{1}{|\mathcal{C}|} \sum_{(\mathbf{p}_i, \mathbf{q}_i, \ldots) \in \mathcal{C}} \|\mathbf{T}_{\mathbf{p},\mathbf{q}} \mathbf{p}_i - \mathbf{q}_i\|_2^2}, \tag{4.8}$$

where the RMSE of the pose is calculated by transforming the correspondences. The loss obtained by iterating through all correspondences is

$$L_{\text{pose}} = \sum_{(\mathbf{p},\mathbf{q},\mathbf{V}_\mathbf{p}^\mathcal{P},\mathbf{V}_\mathbf{q}^\mathcal{Q}) \in \mathcal{C}} \left(1 - \frac{\min(\epsilon(\mathbf{T}_{\mathbf{p},\mathbf{q}}), \gamma)}{\gamma} - s\right), \tag{4.9}$$

where $\gamma \in \mathbb{R}$ is a threshold and s is the score of the point correspondence predicted by the matching network. The proposed L_{pose} can be combined with any of the widely used loss functions, *e.g.*, registration loss. It bridges the gap between correspondence matching and registration and unlocks the end-to-end training.

4.4 Inference Time

While the proposed *Q-REG* is capable of propagating the gradients at training time, during inference, I equip it with components that ensure high accuracy but are non-differentiable. *Q-REG* iterates through the poses calculated from all tentative correspondences, by the proposed single-correspondence solver, in an exhaustive manner. For each match, the pose quality is calculated as the cardinality of its support, *i.e.*, the number of inliers. After the best model is found, I apply local optimization similar to [47], a local re-sampling and re-fitting of inlier correspondences based on their normals (coming from the fitted quadrics) and positions. Specifically, I select the correspondence set with the highest confidence based on the best rigid pose obtained and iterate through all the combinations with the designed two-point estimator. Theoretically, we need at least one point and two vector pairs to estimate the rigid transformation. We can achieve it with two points and their corresponding normals (at least one normal). Given two point correspondences $\{(\tilde{\mathbf{p}}_i, \tilde{\mathbf{q}}_i) \mid \tilde{\mathbf{p}}_i \in \mathcal{P}, \tilde{\mathbf{q}}_i \in \mathcal{Q},\ i \in [1, 2]\}$ and their

normal correspondences $\{(\tilde{\mathbf{n}}_i, \tilde{\mathbf{m}}_i) \mid \tilde{\mathbf{n}}_i \in \mathcal{N}, \tilde{\mathbf{m}}_i \in \mathcal{M},\ i \in [1,2]\}$, the objective function can be written as follows:

$$\min_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{2} w_i(\|\mathbf{R}\tilde{\mathbf{p}}_i + \mathbf{t} - \tilde{\mathbf{q}}_i\|_2^2 + \alpha_i^2\|\mathbf{R}\tilde{\mathbf{n}}_i - \tilde{\mathbf{m}}_i)\|_2^2), \tag{4.10}$$

where w_i is the weight for different correspondences and α_i is the weight for the normals. We can rewrite it as follows:

$$\min_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{2} w_i(\|\mathbf{R}\tilde{\mathbf{p}}'_i - \tilde{\mathbf{q}}'_i\|_2^2 + \alpha_i^2\|\mathbf{R}\tilde{\mathbf{n}}_i - \tilde{\mathbf{m}}_i\|_2^2 + \|\mathbf{R}\bar{\mathbf{p}} + \mathbf{t} - \bar{\mathbf{q}}\|_2^2), \tag{4.11}$$

where $\tilde{\mathbf{p}}'_i = \tilde{\mathbf{p}}_i - \bar{\mathbf{p}}$, $\tilde{\mathbf{q}}'_i = \tilde{\mathbf{q}}_i - \bar{\mathbf{q}}$, $\bar{\mathbf{p}} = \frac{1}{2}(\tilde{\mathbf{p}}_1 + \tilde{\mathbf{p}}_2)$ and $\bar{\mathbf{q}} = \frac{1}{2}(\tilde{\mathbf{q}}_1 + \tilde{\mathbf{q}}_2)$. Similar to [32], we define

$$\mathbf{H} = \sum_{i=1}^{2} w_i\ [\tilde{\mathbf{p}}'_i\tilde{\mathbf{q}}'^{\mathrm{T}}_i + (\alpha_i\tilde{\mathbf{n}}_i)(\alpha_i\tilde{\mathbf{m}}_i)^{\mathrm{T}}]. \tag{4.12}$$

We can decompose $\mathbf{H}$ as

$$\mathbf{H} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}} \tag{4.13}$$

using SVD decomposition. The estimated $\hat{\mathbf{R}}_j$ and $\hat{\mathbf{t}}_j$ can be represented as

$$\hat{\mathbf{R}} = \mathbf{V}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & det(\mathbf{V}\mathbf{U}^{\mathrm{T}}) \end{bmatrix}\mathbf{U}^{\mathrm{T}}, \tag{4.14}$$

$$\hat{\mathbf{t}} = \bar{\mathbf{q}} - \hat{\mathbf{R}}\bar{\mathbf{p}}. \tag{4.15}$$

The best model is selected according to the pose quality. The pose can be further refined with the inliers predicted from the previous step.

5 Experiments

I evaluate the *Q-REG* solver with three state-of-the-art matchers (Predator [14], RegTR [17], and GeoTr [16]) on the real, indoor point cloud datasets *3DMatch* [21] and *3DLoMatch* [14] (Section 5.1). I also evaluate *Q-REG* with GeoTr and RegTR on the synthetic, object-centric datasets *ModelNet* [22] and *ModelLoNet* [14] (Section 5.2). In all datasets, I evaluate the use of *Q-REG* for inference, as well as for end-to-end training. Furthermore, I evaluate the importance of different *Q-REG* components on the best-performing matcher on *3DMatch* and *3DLoMatch*, as well as run-time during inference (Section 5.3).

5.1 3DMatch & 3DLoMatch

The *3DMatch* [21] dataset contains 62 scenes in total, with 46 used for training, 8 for validation, and 8 for testing. I use the training data preprocessed by Huang et al. [14] and evaluate on both *3DMatch* and *3DLoMatch* [14] protocols. The point cloud pairs in *3DMatch* have more than 30% overlap, whereas those in *3DLoMatch* have a low overlap of 10% - 30%. Following prior work [16, 17], I evaluate the following metrics: (i) Registration Recall (RR), which measures the fraction of successfully registered pairs, defined as having a correspondence RMSE below 0.2 m; (ii) Relative Rotation Error (RRE); and (iii) Relative Translation Error (RTE). Both (ii) and (iii) measure the accuracy of successful registrations. Additionally, I report the mean RRE, RTE, and RMSE. In this setting, I evaluate over all valid pairs[1] instead of only those with an RMSE below 0.2 m, and I provide a simple average over all valid pairs instead of the median value of each scene followed by the average over all scenes. These metrics will show how consistently well (or not) a method performs in registering scenes.

I report several learned correspondence-based algorithms on the two datasets. For [11, 13, 12], I tabulate the results as reported in their original papers. For [14, 17, 16], I evaluate them with and without the *Q-REG* solver on all metrics. I also report methods that do not employ RANSAC [40, 43, 36] – results are taken from [17].

The results for *3DLoMatch* and *3DMatch* are tabulated in Tables 5.1 and 5.2 respectively. Note that, unless differently stated, hereafter the best values per group are in **bold** and the absolute best is **underlined**. Also, *Q-REG* means that the solver is used only in inference and ***Q-REG**** means it is used in both end-to-end training and inference. In the latter case, I train from scratch the correspondence matching network with the addition of the pose-induced loss. $1K$ or $50K$ refers to RANSAC iterations. Last, if nothing is added next to the method, the standard formulation is used.

[1] According to [21], a valid pair is a pair of non-consecutive frames.

Model	RR (%)↑	RRE (°)↓	RTE (cm)↓	*Mean* RRE ↓	*Mean* RTE ↓	*Mean* RMSE (cm)↓
3DSN [11]	33.0	3.53	10.3	-	-	-
FCGF [13]	40.1	**3.15**	10.0	-	-	-
D3Feat [12]	37.2	3.36	10.3	-	-	-
OMNet [40]	8.4	7.30	15.1	-	-	-
DGR [43]	48.7	3.95	11.3	-	-	-
PCAM [36]	**54.9**	3.53	**9.9**	-	-	-
Predator [14] + 1k	49.7	3.66	10.7	37.57	95.3	90.0
Predator [14] + 50k	60.4	3.21	9.5	30.07	76.2	72.8
Predator [14] + Q-REG	**66.6**	**2.70**	**8.1**	**28.44**	**71.9**	**68.8**
RegTR [17]	64.8	2.83	8.0	23.05	64.4	55.8
RegTR [17] + 1K	64.4	2.89	8.6	22.38	63.1	55.1
RegTR [17] + 50K	64.3	2.92	8.5	21.90	62.5	54.5
RegTR [17] + Q-REG	**65.3**	**2.81**	**7.8**	**21.43**	**60.9**	**53.3**
GeoTr [16]	74.1	2.59	7.3	23.15	58.3	57.8
GeoTr [16] + 1K	73.6	2.81	8.3	24.04	60.4	60.2
GeoTr [16] + 50K	75.0	2.54	7.7	22.69	57.8	57.3
GeoTr [16] + Q-REG	77.1	2.44	7.7	16.70	**46.0**	44.6
GeoTr [16] + ***Q-REG****	**78.3**	**2.38**	**7.2**	**15.65**	46.3	**42.5**

Table 5.1: Correspondence matching algorithms on the ***3DLoMatch*** [14] dataset. The three state-of-the-art matchers show improved performance on most metrics when combined with the *Q-REG* solver. The best values are **bold** in each group. The absolute best are **underlined**.

Model	RR (%)↑	RRE (°)↓	RTE (cm)↓	*Mean* RRE ↓	*Mean* RTE ↓	*Mean* RMSE (cm)↓
3DSN [11]	78.4	2.20	7.1	-	-	-
FCGF [13]	85.1	1.95	6.6	-	-	-
D3Feat [12]	81.6	2.16	6.7	-	-	-
OMNet [40]	35.9	4.17	10.5	-	-	-
DGR [43]	85.3	2.10	6.7	-	-	-
PCAM [36]	**85.5**	**1.81**	**5.9**	-	-	-
Predator [14] + 1k	86.7	2.11	6.6	9.97	30.3	24.8
Predator [14] + 50k	89.3	1.98	6.5	6.80	20.2	18.3
Predator [14] + Q-REG	**90.6**	**1.74**	**5.7**	**6.78**	**20.0**	**18.1**
RegTR [17]	92.0	**1.57**	**<u>4.9</u>**	5.31	17.0	13.8
RegTR [17] + 1K	91.4	1.76	5.7	5.21	17.1	14.5
RegTR [17] + 50K	91.3	1.72	5.9	5.26	17.5	14.7
RegTR [17] + Q-REG	**92.2**	**1.57**	**<u>4.9</u>**	**5.13**	**16.5**	**13.6**
GeoTr [16]	92.5	1.54	**5.1**	7.04	19.4	17.6
GeoTr [16] + 1K	91.9	1.73	5.6	6.75	18.4	17.0
GeoTr [16] + 50K	92.2	1.66	5.6	6.85	18.7	17.1
GeoTr [16] + Q-REG	93.8	1.57	5.3	4.74	15.0	12.8
GeoTr [16] + ***Q-REG****	**<u>95.2</u>**	**<u>1.53</u>**	5.3	**<u>3.70</u>**	**<u>12.5</u>**	**<u>10.7</u>**

Table 5.2: Correspondence matching algorithms on the ***3DMatch*** [21] dataset. Even on this saturated dataset, state-of-the-art matchers show improved performance when combined with the *Q-REG* solver. The best values are **bold** in each group. The absolute best are **<u>underlined</u>**.

In all three matchers, incorporating *Q-REG* in inference time yields an increase in RR that ranges from 1.0 to 16.9% in *3DLoMatch* and from 0.9 to 3.9% in *3DMatch*. The range difference between the two datasets is expected, since *3DMatch* is more saturated and the gap for improvement is small. Using *Q-REG* for inference achieves the second-best results overall (GeoTr + Q-REG). Even in the case of RegTR, where applying RANSAC ends in decreasing performance [17], *Q-REG* can still provide a boost in all metrics. When training end-to-end the best-performing matcher, GeoTr, I gain further boost and achieve the best results overall in both datasets, setting a new benchmark (GeoTr + ***Q-REG****). This behavior can be observed not only on the standard metrics (RR, RRE, RTE), but also at the Mean RRE, RTE, and RMSE. As expected, *Q-REG* results in smaller errors regardless of the matcher.

For a detailed comparison, I plot the cumulative distribution functions (CDF) of the following registration metrics: Relative Rotation Error (RRE) (Figure 5.1), Relative Translation Error (RTE) (Figure 5.2), and Root Mean Square Error (RMSE) (Figure 5.3), for the *3DLoMatch* [14] and *3DMatch* [21] datasets (Figure 5.4, 5.5 and 5.6). Being close to the top-left corner is interpreted as being accurate. As expected, when using *Q-REG*, state-of-the-art correspondence matching algorithms have improved performance with respect to their standard formulation. What can also be observed, is that when training the matcher in an end-to-end fashion with ***Q-REG****, I achieve new state-of-the-art results. The above stand for both *3DLoMatch* and *3DMatch* datasets.

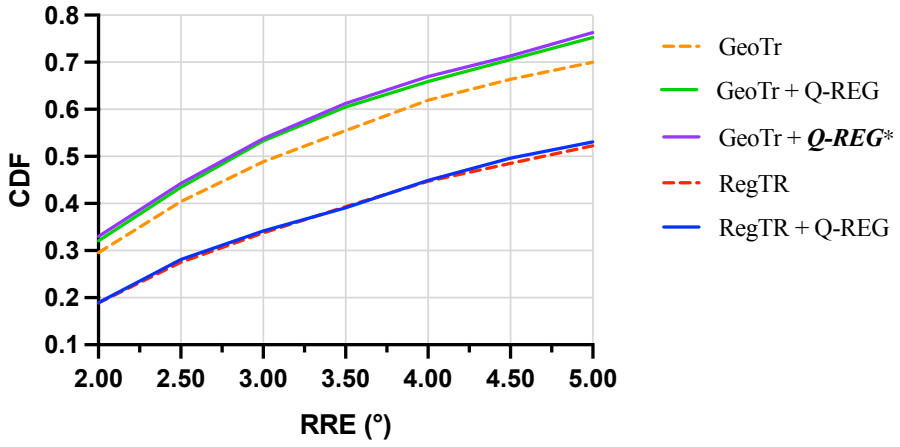


Figure 5.1: Cumulative distribution functions of RRE for the *3DLoMatch [14]* dataset. When using *Q-REG* the performance of state-of-the-art matchers is increased, with the best results achieved when *Q-REG* is used for end-to-end training. GeoTr [16] refers to the standard formulation, GeoTr+Q-REG to using the method during inference, and GeoTr+***Q-REG**** to using the method for end-to-end training. Similar for RegTr [17] and RegTr+Q-REG. Being close to the top-left corner is interpreted as being accurate.

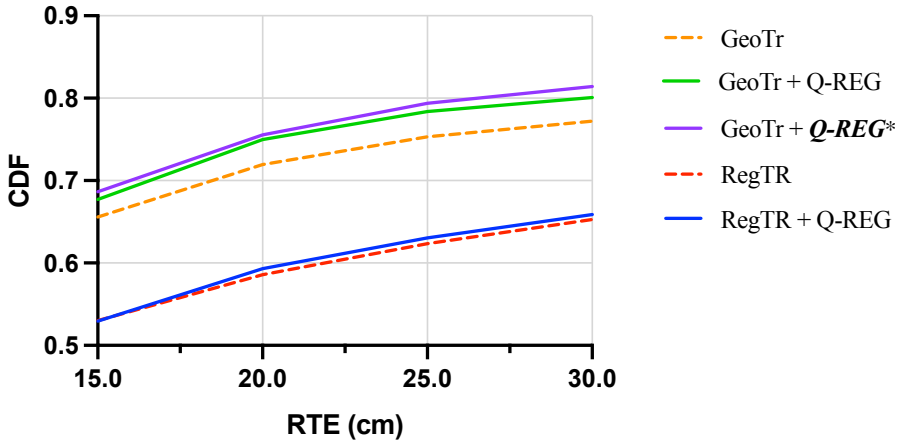


Figure 5.2: Cumulative distribution functions of RTE for the *3DLoMatch [14]* dataset.

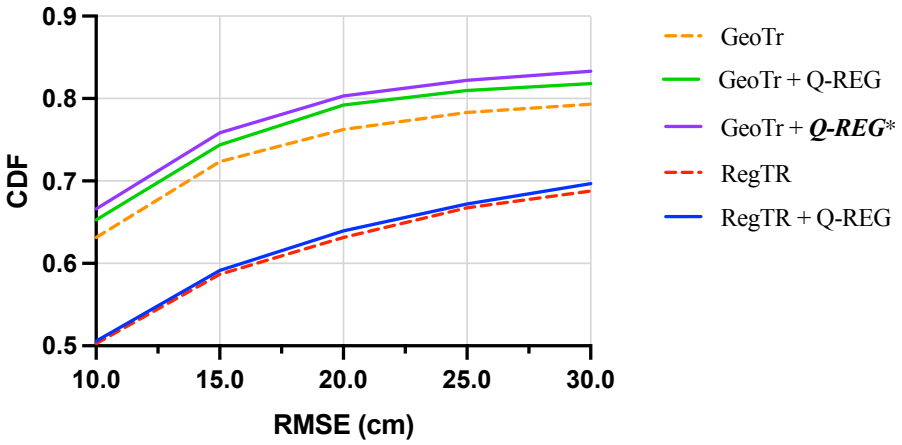


Figure 5.3: Cumulative distribution functions of RMSE for the *3DLoMatch [14]* dataset.

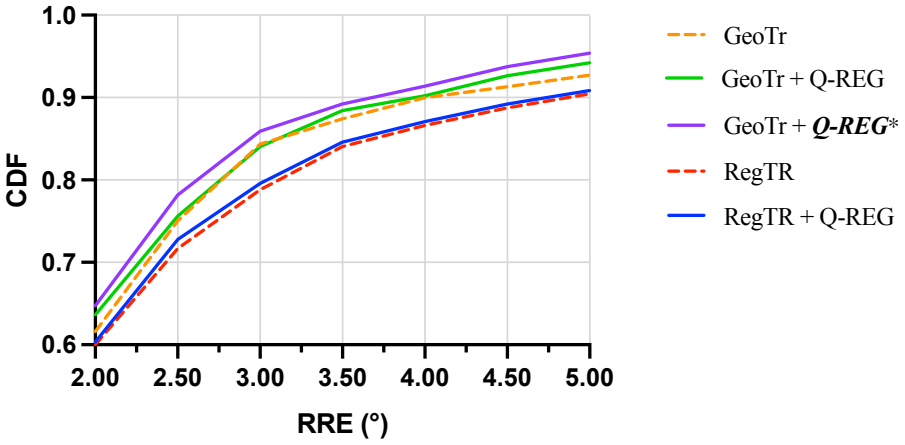


Figure 5.4: Cumulative distribution functions of RRE for the *3DMatch [21]* dataset. When using *Q-REG* the performance of state-of-the-art matchers is increased, with the best results achieved when *Q-REG* is used for end-to-end training. GeoTr [16] refers to the standard formulation, GeoTr+Q-REG to using the method during inference, and GeoTr+***Q-REG**** to using the method for end-to-end training. Similar for RegTr [17] and RegTr+Q-REG. Being close to the top-left corner is interpreted as being accurate.

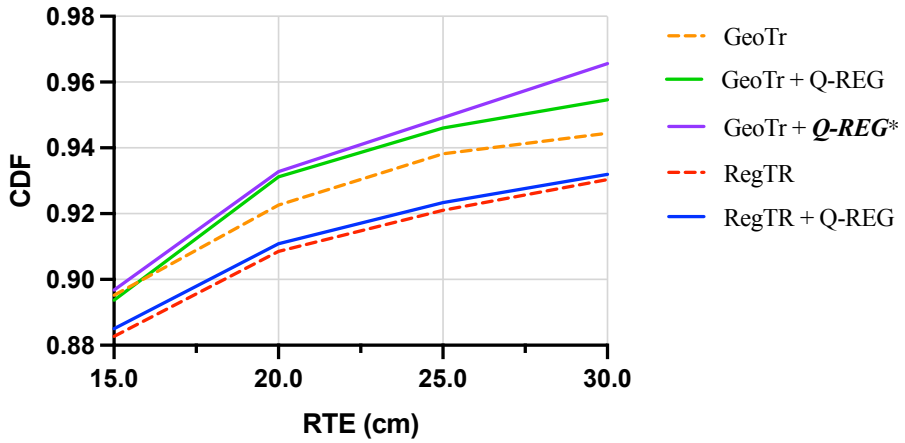


Figure 5.5: Cumulative distribution functions of RTE for the *3DMatch [21]* dataset.

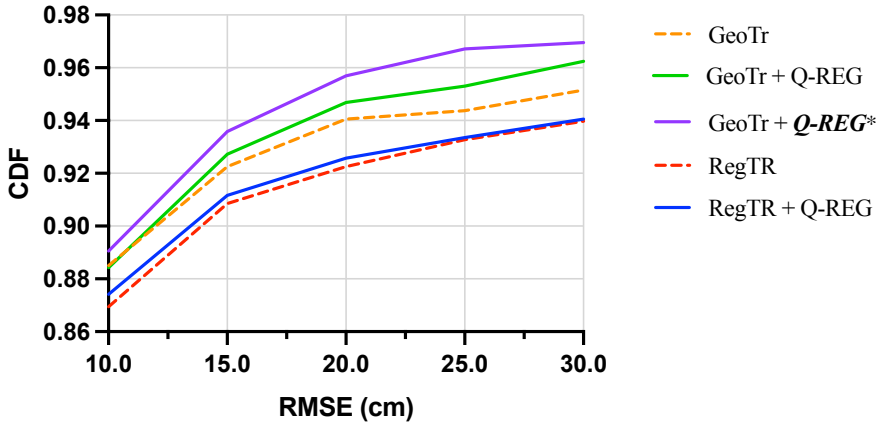


Figure 5.6: Cumulative distribution functions of RMSE for the *3DMatch [21]* dataset.

Qualitative results are illustrated in Figure 5.7. In the first row, GeoTr+***Q-REG**** achieves a good alignment of the point clouds when all other methods fail, which highlights the importance of end-to-end training. The shown examples in the second row demonstrate that it is hard for *Q-REG* to recover a good pose when the standard formulation of RegTR fails in the first place. This means that the estimated correspondences are not good, to the point that a robust estimator cannot identify a good subset. Similar to the above, end-to-end training could address this since the correspondences would be learned jointly with the pose minimization objective. When RegTR finds a correct pose, *Q-REG* can further optimize it, as shown in the fourth row. In the same example, although GeoTr fails to infer a good pose, both *Q-REG* and ***Q-REG**** are able to recover it.

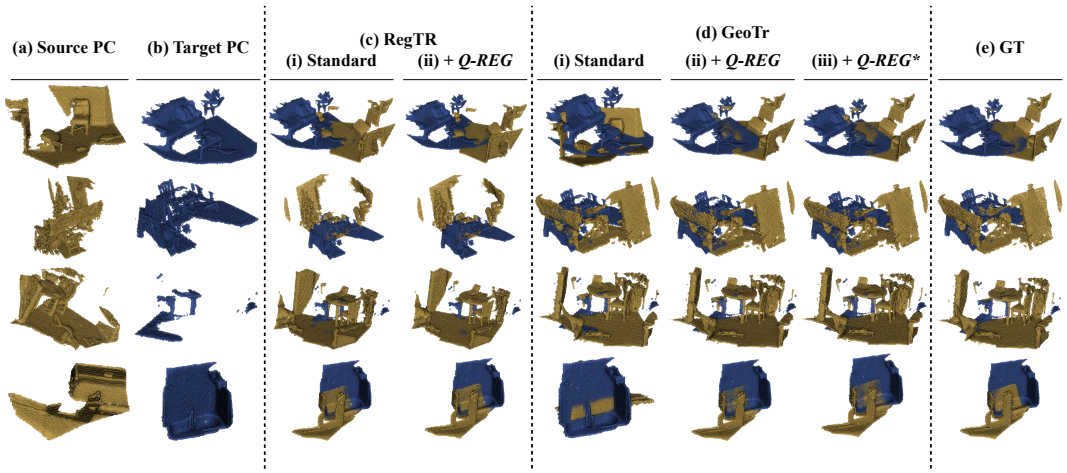


Figure 5.7: Qualitative Results. I showcase registration examples of RegTR [17] and GeoTr [16] with and without *Q-REG* for the *3DLoMatch* (first two rows) and *3DMatch* (last two rows) datasets. *(Best viewed on screen.)*

In Figures 5.8 and 5.9, I showcase additional qualitative registration results for the *3DLoMatch* and *3DMatch* datasets, respectively. I evaluate the state-of-the-art matchers GeoTr [16] and RegTr [17]. Please note that GeoTr Standard (d-i) refers to the standard formulation, GeoTr+Q-REG (d-ii) to using the presented method during inference, and GeoTr+***Q-REG**** (d-iii) to using the presented method for end-to-end training. Similar for RegTr Standard (c-i) and RegTr+Q-REG (c-ii).

In Figure 5.8, I illustrate qualitative examples of point cloud registration for the *3DLoMatch* dataset. Specifically, per row:

Row (1): In the case of GeoTr, the standard formulation (d-i) already produces well-aligned point clouds, and the addition of *Q-REG* (d-ii and d-iii) slightly refines this output. However, in the case of RegTr, we can see the most improvement. The standard formulation (c-i) fails to achieve a good alignment and *Q-REG* (c-ii) is able to recover a good pose. This means that *Q-REG* is able to identify robust correspondences and remove spurious ones.

Row (2): In this example the opposite behavior is observed, where Reg-Tr+Q-REG (c-ii) further refines the good results achieved by RegTr standard (c-i). GeoTr standard (d-i) fails to achieve a good registration, but the use of *Q-REG* (d-ii and d-iii) can recover the pose.

Row (3): Here, both RegTr standard (c-i) and GeoTr standard (d-i) fail to align the point clouds correctly. Despite this failure, in both matchers, the use of *Q-REG* (c-ii, d-ii, and d-iii) allows to recover the final pose.

Rows (4), (5), and (6): In these examples, the only method that allows to recover the final pose is GeoTr+***Q-REG**** (d-iii). This means that using *Q-REG* in end-to-end training can provide additional improvements in performance by learning how to better match correspondences together with the objective of rigid pose estimation, and not in isolation as it happens in all other cases (c-i, c-ii, d-i, and d-ii).

Row (7): In this final example, all methods perform reasonably well. However, in the case of GeoTr+Q-REG (d-ii) and GeoTr+***Q-REG**** (d-iii), the RMSE can be reduced from 16 cm in the standard formulation (d-i) to 5 cm, which is a substantial improvement in the estimated pose. It demonstrates the superiority of *Q-REG* in easy cases.

In Figure 5.9, I illustrate qualitative examples of point cloud registration for the *3DMatch* dataset. Specifically, per row:

Rows (1) and (2): In these examples, the addition of *Q-REG* (c-ii) to the standard RegTr formulation (c-i) is able to greatly correct the estimated pose, although it does not achieve perfect results. In the case of GeoTr+Q-REG (d-ii), it cannot correct the error of GeoTr standard (c-i), however, GeoTr+***Q-REG**** (d-iii) recovers the final pose. This points out, as mentioned beforehand, to the power of learning to choose (hard) correspondences with the inclusion of the pose error minimization objective.

Rows (3), (4), (5), and (6): Here, although there is no big improvement between RegTr standard (c-i) and RegTr+Q-REG (c-ii), this changes in the case of GeoTr. The standard formulation (d-i) fails to estimate a good alignment, but the use of *Q-REG* (d-ii) and ***Q-REG**** (d-iii) provide a close-to-GT pose estimation, with ***Q-REG**** being better.

Row (7): In this final example, all methods perform reasonably well. However, in the case of GeoTr+Q-REG (d-ii) and GeoTr+***Q-REG**** (d-iii) the presented method reduces the RMSE from 13 cm in the standard formulation (d-i) to 4 cm and 2 cm respectively.

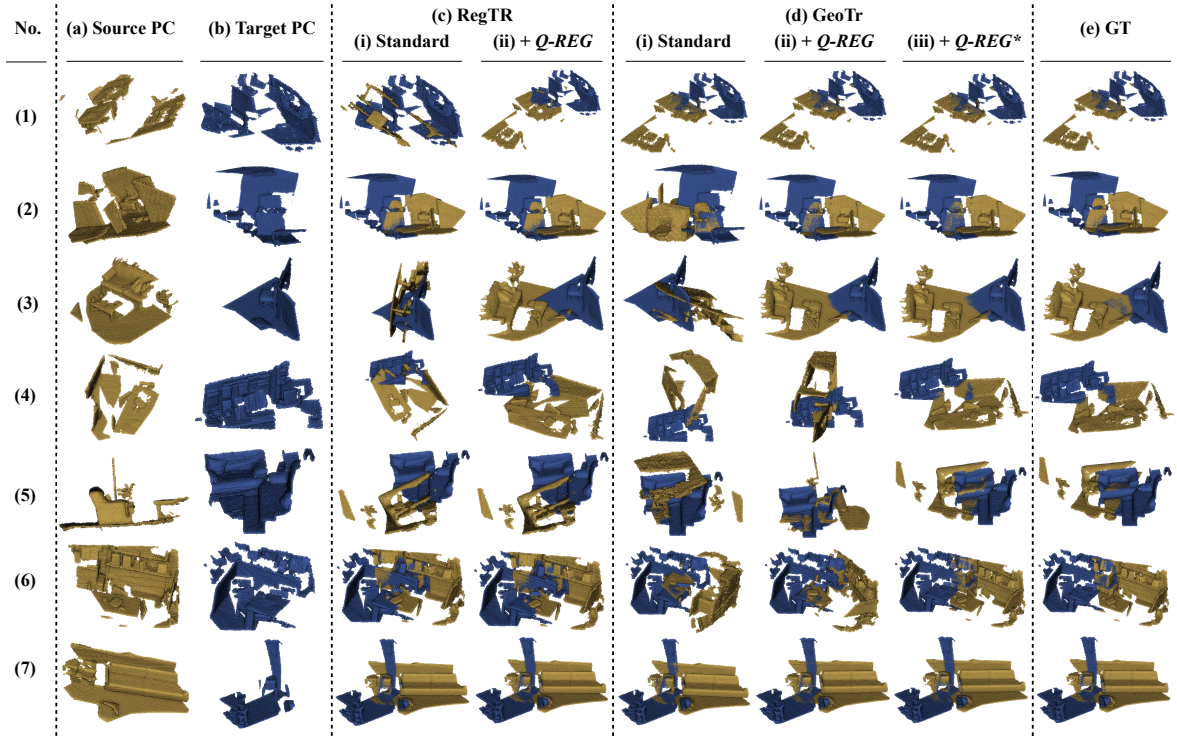


Figure 5.8: Qualitative Results for the *3DLoMatch [14]* dataset. See Section 5.1 for an explanation of the results.

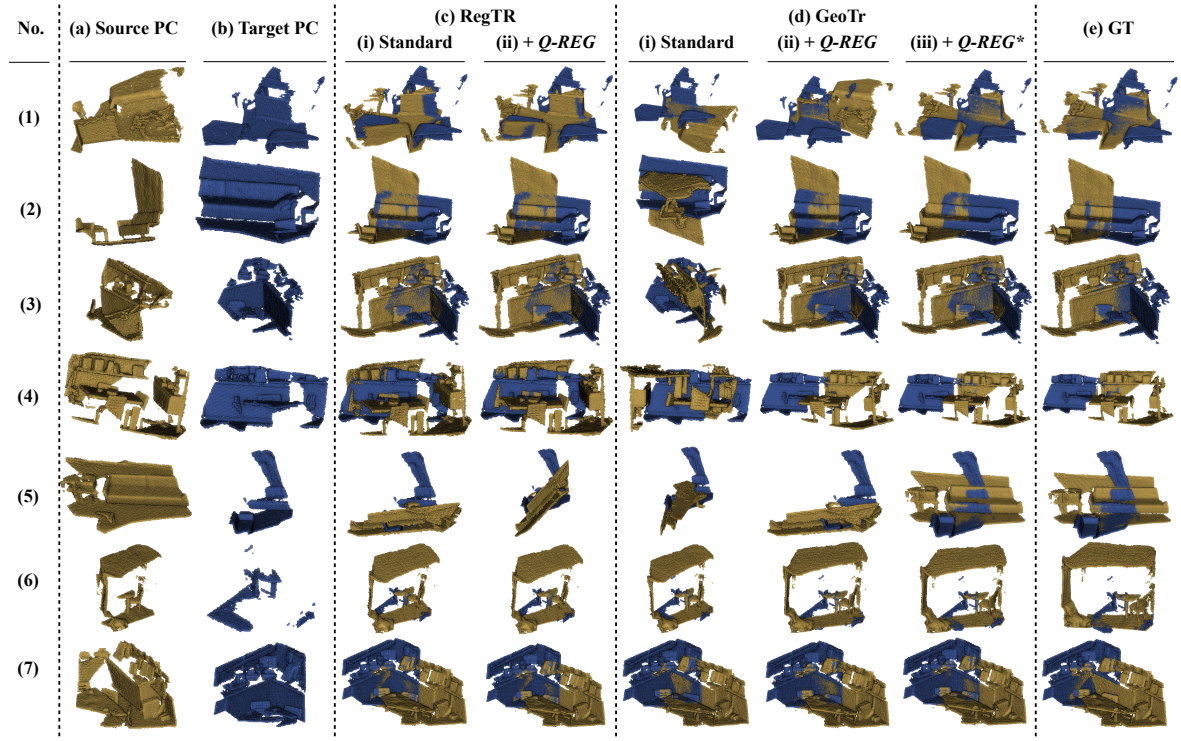


Figure 5.9: Qualitative Results for the *3DMatch [21]* dataset. See Section 5.1 for an explanation of the results.

5.2 ModelNet & ModelLoNet

The *ModelNet* [22] dataset contains in total of 12,311 CAD models of man-made objects from 40 categories, with 5,112 used for training, 1,202 for validation, and 1,266 for testing. I use the partial scans created by Yew et al. [19] for evaluating on *ModelNet* and those created by Huang et al. [14] for evaluating on *ModelLoNet*. The point cloud pairs in *ModelNet* have 73.5% overlap on average, whereas those in *ModelLoNet* have 53.6%. Following prior work [14, 17], I evaluate the following metrics: (i) Chamfer Distance (CD) between registered point clouds; (ii) Relative Rotation Error (RRE); and (iii) Relative Translation Error (RTE). I report several recent algorithms on the two datasets. For [38, 14], I tabulate the results as reported in their original papers. For [18, 19, 40], results are taken from [17]. For [17, 16], I evaluate them with and without *Q-REG*, similarly to the Section 5.1.

The results for *ModelNet* [22] and *ModelLoNet* [14] are tabulated in Tables 5.3 and 5.4, respectively. Here a similar trend can be observed in the results, with *Q-REG* boosting the performance of all matchers. RegTR with *Q-REG* achieves the best results overall on both datasets (RegTR + Q-REG). In addition, both when *Q-REG* is used for inference and end-to-end training, the results of GeoTr are also improved with respect to its standard formulation.

Method	ModelNet [22]		
	CD ↓	RRE (°)↓	RTE (cm)↓
PointNetLK [38]	0.02350	29.73	29.7
OMNet [40]	0.00150	2.95	3.2
DCP-v2 [18]	0.01170	11.98	17.1
RPM-Net [19]	**0.00085**	**1.71**	**1.8**
Predator [14]	0.00089	1.74	1.9
RegTR [17]	0.00078	1.47	1.4
RegTR [17] + 1K	0.00088	1.80	1.7
RegTR [17] + 50K	0.00091	1.82	1.8
RegTR [17] + Q-REG	**<u>0.00074</u>**	**<u>1.35</u>**	**<u>1.3</u>**
GeoTr [16]	0.00083	2.16	2.0
GeoTr [16] + 1K	0.00106	2.76	2.6
GeoTr [16] + 50K	0.00095	2.40	2.2
GeoTr [16] + Q-REG	0.00078	1.84	1.7
GeoTr [16] + ***Q-REG****	**0.00076**	**1.73**	**1.5**

Table 5.3: Evaluation of state-of-the-art matchers on the ***ModelNet*** [22] dataset. The best values are **bold** in each group. The absolute best are **<u>underlined</u>**.

Method	ModelLoNet [14]		
	CD ↓	RRE (°)↓	RTE (cm)↓
PointNetLK [38]	0.0367	48.57	50.7
OMNet [40]	0.0074	6.52	12.9
DCP-v2 [18]	0.0268	6.50	30.0
RPM-Net [19]	**0.0050**	7.34	**12.4**
Predator [14]	0.0083	**5.24**	13.2
RegTR [17]	0.0037	3.93	8.7
RegTR [17] + 1K	0.0039	4.20	9.1
RegTR [17] + 50K	0.0039	4.23	9.2
RegTR [17] + Q-REG	**<u>0.0034</u>**	**<u>3.65</u>**	**8.1**
GeoTr [16]	0.0050	4.49	7.6
GeoTr [16] + 1K	0.0051	4.99	8.4
GeoTr [16] + 50K	0.0050	4.27	8.0
GeoTr [16] + Q-REG	0.0044	3.87	7.0
GeoTr [16] + ***Q-REG****	**0.0040**	**3.73**	**<u>6.5</u>**

Table 5.4: Evaluation of state-of-the-art matchers on the ***ModelLoNet*** [14] dataset. The best values are **bold** in each group. The absolute best are **<u>underlined</u>**.

Qin et al. [16] do not provide any *ModelNet* dataset evaluation in their paper, however, they offer an evaluation on their GitHub page [48]. I follow their proposed protocol and dataset split for the standard setting. In a nutshell, (i) they use a portion of the *ModelNet* dataset that excludes the axis-symmetrical object categories and (ii) they train and test on all categories, instead of keeping certain ones unseen for testing. I trained GeoTr [16] from scratch on this setting and report the results on the same metrics in Table 5.5 (*best results marked in* ***bold***). It can be seen that the *Q-REG*, in both an only-testing setting and an end-to-end-training one, performs the best, and minimizes all three metrics with respect to the best-performing GeoTr without *Q-REG*.

Method	CD ↓	RRE (°)↓	RTE (cm)↓
GeoTr [16]	0.00093	1.61	1.9
GeoTr [16] + 1K	0.00124	2.31	2.7
GeoTr [16] + 50K	0.00112	1.94	2.3
GeoTr [16] + Q-REG	0.00088	1.43	1.7
GeoTr [16] + ***Q-REG****	**0.00085**	**1.26**	**1.5**

Table 5.5: Evaluation of GeoTr [16] on a different ***ModelNet*** setting: (i) using a portion of it that excludes axis-symmetrical categories and (ii) using all categories in both training and testing.

5.3 Ablation Studies

I perform ablation studies to evaluate the contribution of each component in the *Q-REG* solver to the best-performing matcher on the *3DMatch* and *3DLoMatch* datasets, the state-of-the-art GeoTr [16]. I evaluate the following self-baselines (i and ii are inference only): (i) **GeoTr + Q**: My quadric-fitting 1-point solver. (ii) **GeoTr + QL**: I extend the quadric fitting with local optimization as discussed in Section 4.3. (iii) **GeoTr + QT**: The quadric-fitting solver is used in end-to-end training – during inference I do not employ local optimization; and (iv) **GeoTr + QTL (*Q-REG**)**: The quadric-fitting 1-point solver is used in end-to-end training followed by inference using the local optimization.

The results for *3DLoMatch* and *3DMatch* are tabulated in Tables 5.6 and 5.7 respectively (*best results in* ***bold****, second best are* underlined). In both datasets, ***Q-REG**** performs the best in the majority of the metrics. Specifically, in *3DLoMatch*, there is a substantial increase in RR by 4.2%. Even in the saturated *3DMatch*, an increase of 2.7% is obtained. When the solver is used only during inference, a 3.0% and 1.3% increase can still be seen in RR per dataset. As expected, GeoTr + QT performs very closely to ***Q-REG****, since the main difference is the absence of the local optimization that refines the estimated pose. When considering the mean RRE, RTE, and RMSE, the self-baselines provide consistently more robust results over all valid pairs.

Model	RR (%)↑	RRE (°)↓	RTE (cm)↓	*Mean* RRE ↓	*Mean* RTE ↓	*Mean* RMSE ↓
GeoTr + LGR	74.1	2.59	<u>7.3</u>	23.15	58.3	57.8
GeoTr + 1K	73.6	2.81	8.3	24.04	60.4	60.2
GeoTr + 50K	75.0	2.54	7.7	22.69	57.8	57.3
i) GeoTr + Q	75.5	2.47	7.6	22.38	57.6	57.3
ii) GeoTr + QL (Q-REG)	77.1	2.44	7.7	<u>16.70</u>	**46.0**	<u>44.6</u>
iii) GeoTr + QT	<u>77.2</u>	**2.37**	7.5	17.32	50.3	47.4
iv) GeoTr + QTL (***Q-REG****)	**78.3**	<u>2.38</u>	**7.2**	**15.65**	<u>46.3</u>	**42.5**

Table 5.6: Ablation results on the ***3DLoMatch*** [14] dataset of GeoTr [16] with different aspects of the *Q-REG* solver. The best values are **bold** and the 2nd best are <u>underlined</u>.

Model	RR (%)↑	RRE (°)↓	RTE (cm)↓	*Mean* RRE ↓	*Mean* RTE ↓	*Mean* RMSE ↓
GeoTr + LGR	92.5	1.54	**5.1**	7.04	19.4	17.6
GeoTr + 1K	91.1	1.73	5.6	6.75	18.4	17.0
GeoTr + 50K	92.2	1.66	5.6	6.85	18.7	17.1
i) GeoTr + Q	92.6	1.55	5.3	6.26	17.4	15.8
ii) GeoTr + QL (Q-REG)	93.8	1.57	5.3	4.74	15.0	12.8
iii) GeoTr + QT	<u>94.3</u>	**1.51**	<u>5.2</u>	<u>3.78</u>	<u>12.8</u>	<u>10.9</u>
iv) GeoTr + QTL (***Q-REG****)	**95.2**	<u>1.53</u>	5.3	**3.70**	**12.5**	**10.7**

Table 5.7: Ablation results on the ***3DMatch*** [21] dataset of GeoTr [16] with different aspects of the *Q-REG* solver. The best values are **bold** and the 2nd best are <u>underlined</u>.

Run-time. I compute the average run-time in seconds for each component in Table 5.8 (evaluated with GeoTr on the *3DLoMatch* dataset). With respect to RANSAC 50K, which yields at least 2% lower RR, *QREG* provides better results while being an order of magnitude faster. All run-time experiments were run on 8 Intel Xeon Gold 6150 CPUs and an NVIDIA GeForce RTX 3090 GPU.

LGR	+1K	+50K	+Q	+QL (Q-REG)
0.016	0.053	1.809	0.085	0.166

Table 5.8: Run-time evaluation in seconds during inference using GeoTr [16] on the ***3DLoMatch*** dataset. Times shown for LGR, RANSAC running 1K and 50K iterations, Quadric solvers (Sec. 4.2), and with the entire *Q-REG* algorithm.

6 Discussion

I develop *Q-REG*, a robust solution for point cloud registration, estimating the pose from a single correspondence via leveraging local surface patches. It is agnostic to the correspondence matching method. *Q-REG* allows for quick outlier rejection by filtering degenerate solutions and assumption inconsistent motions (*e.g.*, related to scale). I extend the above formulation of *Q-REG* to a differentiable setting that allows for end-to-end training of correspondence matching methods with this presented solver. Thus, it optimize not only over the correspondence matching but also over the final pose.

The main bottleneck of the current solution is the criteria to select the best rigid pose, *i.e.*, the number of inliers. In most cases, it is a good measure of the pose quality. However, it fails to find a close-to-best solution especially when there are many ambiguous feature matchings or most of the putative correspondences are wrong. One possible solution is to design a network to learn the criteria, considering both the cardinality of its support and global spatial information. Another potential extension would be to extend *Q-REG* to estimate not only the rotation and translation but also the isotropic scale parameter.

7 Conclusion

I present a novel solution for point cloud registration, *Q-REG*, that utilizes rich geometric information to estimate the rigid pose from a single correspondence. With *Q-REG*, the number of possible combinations reduces from cubic to linear with respect to the number of correspondences. It allows us to formalize the robust estimation as an exhaustive search and enable us to iterate through all the combinations and select the best rigid pose among them. *Q-REG* utilizes rich geometric cues extracted from local surface patches estimated from observed points, which ensures the robustness of each estimated pose. It is differentiable by design, hence, together with the pose loss, enabling end-to-end training that optimizes over both objectives of correspondence matching and final pose. It performs quick outlier rejection by filtering degenerate solutions and assumption inconsistent motions (*e.g.*, related to scale). *Q-REG* is agnostic to matching methods and is consistently improving their performance on all reported datasets, setting new state-of-the-art on these benchmarks.

Bibliography

[1] J. Zhang and S. Singh, "Visual-lidar odometry and mapping: Low-drift, robust, and fast," in *ICRA*. IEEE, 2015, pp. 2174–2181.

[2] S. Choi, Q.-Y. Zhou, and V. Koltun, "Robust reconstruction of indoor scenes," in *CVPR*, 2015, pp. 5556–5565.

[3] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view rgb-d object dataset," in *ICRA*. IEEE, 2011, pp. 1817–1824.

[4] H. Yang and L. Carlone, "A polynomial-time solution for robust registration with extreme outlier rates," *arXiv*, 2019.

[5] B. Calli, A. Singh, J. Bruce, A. Walsman, K. Konolige, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Yale-cmu-berkeley dataset for robotic manipulation research," *IJRR*, vol. 36, no. 3, pp. 261–268, 2017.

[6] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[7] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[8] H. Yang and L. Carlone, "A quaternion-based certifiably optimal solution to the wahba problem with outliers," in *ICCV*, 2019, pp. 1665–1674.

[9] 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 *ISMAR*. Ieee, 2011, pp. 127–136.

[10] M. A. Audette, F. P. Ferrie, and T. M. Peters, "An algorithmic overview of surface registration techniques for medical imaging," *Medical image analysis*, vol. 4, no. 3, pp. 201–217, 2000.

[11] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3d point cloud matching with smoothed densities," in *CVPR*, 2019.

[12] X. Bai, Z. Luo, L. Zhou, H. Fu, L. Quan, and C.-L. Tai, "D3feat: Joint learning of dense detection and description of 3d local features," in *CVPR*, 2020.

[13] C. Choy, J. Park, and V. Koltun, "Fully convolutional geometric features," in *ICCV*, 2019.

[14] S. Huang, Z. Gojcic, M. Usvyatsov, A. Wieser, and K. Schindler, "Predator: Registration of 3d point clouds with low overlap," in *CVPR*, 2021.

[15] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communication of ACM*, 1981.

[16] Z. Qin, H. Yu, C. Wang, Y. Guo, Y. Peng, and K. Xu, "Geometric transformer for fast and robust point cloud registration," in *CVPR*, 2022.

[17] Z. J. Yew and G. H. Lee, "Regtr: End-to-end point cloud correspondences with transformers," in *CVPR*, 2022.

[18] Y. Wang and J. M. Solomon, "Deep closest point: Learning representations for point cloud registration," in *ICCV*, 2019.

[19] Z. J. Yew and G. H. Lee, "Rpm-net: Robust point matching using learned features," in *CVPR*, 2020.

[20] Y. Wang and J. M. Solomon, "Prnet: Self-supervised learning for partial-to-partial registration," *NeurIPS*, 2019.

[21] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, "3dmatch: Learning local geometric descriptors from rgb-d reconstructions," in *CVPR*, 2017.

[22] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *CVPR*, 2015.

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

[24] A. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3d scenes," *PAMI*, 1999.

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

[26] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *ICRA*, 2009.

[27] F. Tombari, S. Salti, and L. Di Stefano, "Unique signatures of histograms for local surface description," in *ECCV*, 2010.

[28] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," *arXiv preprint arXiv:1703.07737*, 2017.

[29] Y. Sun, C. Cheng, Y. Zhang, C. Zhang, L. Zheng, Z. Wang, and Y. Wei, "Circle loss: A unified perspective of pair similarity optimization," in *CVPR*, 2020.

[30] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3d point cloud matching with smoothed densities," in *CVPR*, 2019.

[31] S. Ao, Q. Hu, B. Yang, A. Markham, and Y. Guo, "Spinnet: Learning a general surface descriptor for 3d point cloud registration," in *CVPR*, 2021.

[32] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-d point sets," *IEEE TPAMI*, 1987.

[33] 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)*, vol. 38, no. 5, pp. 1–12, 2019.

[34] K. Fischer, M. Simon, F. Olsner, S. Milz, H.-M. Gross, and P. Mader, "Stickypillars: Robust and efficient feature matching on point clouds using graph neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 313–323.

[35] J. Li, C. Zhang, Z. Xu, H. Zhou, and C. Zhang, "Iterative distance-aware similarity matrix convolution with mutual-supervised point elimination for efficient point cloud registration," in *ECCV*. Springer, 2020, pp. 378–394.

[36] A.-Q. Cao, G. Puy, A. Boulch, and R. Marlet, "Pcam: Product of cross-attention matrices for rigid registration of point clouds," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 13 229–13 238.

[37] W. Yuan, B. Eckart, K. Kim, V. Jampani, D. Fox, and J. Kautz, "Deepgmr: Learning latent gaussian mixture models for registration," in *ECCV*. Springer, 2020, pp. 733–750.

[38] Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey, "Pointnetlk: Robust & efficient point cloud registration using pointnet," in *CVPR*, 2019.

[39] B. D. Lucas, T. Kanade, *et al.*, *An iterative image registration technique with an application to stereo vision*. Vancouver, 1981, vol. 81.

[40] H. Xu, S. Liu, G. Wang, G. Liu, and B. Zeng, "Omnet: Learning overlapping mask for partial-to-partial point cloud registration," in *ICCV*, 2021.

[41] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother, "Dsac-differentiable ransac for camera localization," in *CVPR*, 2017.

[42] X. Bai, Z. Luo, L. Zhou, H. Chen, L. Li, Z. Hu, H. Fu, and C.-L. Tai, "Pointdsc: Robust point cloud registration using deep spatial consistency," in *CVPR*, 2021.

[43] C. Choy, W. Dong, and V. Koltun, "Deep global registration," in *CVPR*, 2020.

[44] G. D. Pais, S. Ramalingam, V. M. Govindu, J. C. Nascimento, R. Chellappa, and P. Miraldo, "3dregnet: A deep neural network for 3d point registration," in *CVPR*, 2020.

[45] Z. Gojcic, C. Zhou, J. D. Wegner, L. J. Guibas, and T. Birdal, "Learning multiview 3d point cloud registration," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 1759–1769.

[46] Q. Li and J. G. Griffiths, "Least squares ellipsoid specific fitting," in *Geometric modeling and processing, 2004. proceedings.* IEEE, 2004, pp. 335–340.

[47] K. Lebeda, J. Matas, and O. Chum, "Fixing the locally optimized ransac–full experimental evaluation," in *British machine vision conference*, vol. 2. Citeseer, 2012.

[48] "GeoTransformer Github Page," https://github.com/qinzheng93/GeoTransformer, accessed: 2022-11-11.