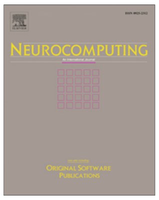
Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom



Reweighted Discriminative Optimization for least-squares problems with point cloud registration

Yan Zhao [a,b], Wen Tang [a,*], Jun Feng [b], TaoRuan Wan [c], Long Xi [a]

[a] Department of Creative Technology, Bournemouth University, Poole BH12 5BB, UK
[b] Department of Information Science and Technology, Northwest University, Xi'an 710127, China
[c] School of Informatics, University of Bradford, Bradford BD7 1DP, UK

A R T I C L E I N F O

Article history:
Received 8 February 2021
Revised 11 August 2021
Accepted 16 August 2021
Available online 20 August 2021
Communicated by Zidong Wang

Keywords:
Least squares
Optimization
Point cloud registration
Supervised learning

A B S T R A C T

Optimization plays a pivotal role in computer graphics and vision. Learning-based optimization algorithms have emerged as a powerful optimization technique for solving problems with robustness and accuracy because it learns gradients from data without calculating the Jacobian and Hessian matrices. The key aspect of the algorithms is the least-squares method, which formulates a general parametrized model of unconstrained optimizations and makes a residual vector approach to zeros to approximate a solution. The method may suffer from undesirable local optima for many applications, especially for point cloud registration, where each element of transformation vectors has a different impact on registration. In this paper, Reweighted Discriminative Optimization (RDO) method is proposed. By assigning different weights to components of the parameter vector, RDO explores the impact of each component and the asymmetrical contributions of the components on fitting results. The weights of parameter vectors are adjusted according to the characteristics of the mean square error of fitting results over the parameter vector space at per iteration. Theoretical analysis for the convergence of RDO is provided, and the benefits of RDO are demonstrated with tasks of 3D point cloud registrations and multi-views stitching. The experimental results show that RDO outperforms state-of-the-art registration methods in terms of accuracy and robustness to perturbations and achieves further improvement than non-weighting learning-based optimization.

1. Introduction

Mathematical optimization plays an essential role in solving many computer graphics and vision problems [1–3]. Gradient-based algorithms are widely used for solving various optimization tasks, such as gradient descent for multi-view reconstruction [4] and object tracking [5], Gauss–Newton for face alignment [6] and surface fitting [7], Levenberg–Marquardt for structure from motion [8], and Conjugate Gradient for surface reconstruction [9]. Newton's algorithm [10], one of the gradient-based optimization algorithms, is a very powerful technique due to its quadratic convergence. Nevertheless, Newton's algorithm requires cost functions to be twice differentiable and the Hessian matrix needs to be positive definite. The computational cost for obtaining the second-order gradient information for the Hessian matrix makes the method not feasible in many cases. Therefore, in order to achieve fast computation time for large and complicated problems, many researchers have been using Quasi-Newton methods [11,12] to generate an estimation of the inverse Hession matrix for finding the local maxima or minima of functions. However, the lack of precision in the Hessian estimation may lead to slow convergence. Another potential disadvantage of the Quasi-Newton method (such as Broyden–Fletcher–Goldfarb–Shanno algorithm (BFGS)) is that storing the inverse Hessian approximation takes a large memory space, which could be detrimental for solving large complex tasks. Limited-memory BFGS (LBFGS) as the variant of BFGS method, only stores a set of vectors and calculates a reduced rank approximation to the Hessian approximation, which needs much less memory to operate. Nevertheless, the amount of storage required by LBFGS depends on the parameter setting which determines the number of BFGS corrections saved. Using traditional gradient-based methods for visualization tasks poses particular challenges because of a large number of parameters and the high complexity of Hessian matrix inversion, with the ample storage required for the inverse Hessian approximation. Learning-based algorithms are proven to

* Corresponding author.
E-mail addresses: zhaoy@bournemouth.ac.uk (Y. Zhao), wtang@bournemouth.ac.uk (W. Tang), zhaoy@bournemouth.ac.uk (J. Feng), t.wan@bradford.ac.uk (T. Wan), lxi@bournemouth.ac.uk (L. Xi).

https://doi.org/10.1016/j.neucom.2021.08.080
0925-2312/© 2021 The Authors. Published by Elsevier B.V.

be efficient in overcoming the limitations since the gradient directions can be learned without calculating the Jacobian matrix or the Hessian matrix. For example, the Supervised Descent Method (SDM) [13,14] learns a sequence of linear maps as gradient directions through minimizing nonlinear least-squares functions in a feature space, which avoids the expensive computation of the Jacobian and Hessian metrics. The Discriminative Optimization (DO) method [15] learns a sequence of maps that update a set of parameters to a stationary point from the training data and mimics gradient descent without the explicit modeling of cost functions. The critical insight of these learning-based algorithms is that the updating gradients are learned by approaching the currently estimated parameter vectors towards the ground truth. Although the key insight of learning-based algorithms avoids calculating the Jacobian and Hessian metrics, it ignores the different impact of each component of parameter vectors on fitting results. For point cloud registration, the element of parameter vectors plays a different role in the registration process, as shown in Fig. 1. Fig. 1 shows that the approximation of the norm of residual vectors to minimum does not represent the best fitting between the matching models, and the components of the parameter vector have different impacts on the final fitting results.

In this paper, we propose a new method-Reweighted Discriminative Optimization (RDO), to address the issue of asymmetrical contributions of the components of the parameter vector on fitting accuracy. Our new method assigns different weights to components of parameter vectors in each iteration to emphasize the impact of each component on the fitting results. It is worth noting that our method is different from the Iterative Reweighted Least Squares (IRLS) [16]. IRLS attains weights through an M-estimation criterion to emphasize specific components or the range of equations, where the weighting matrix of $\ell 2$ norm is an identity matrix. When IRLS is applied to point cloud registrations to reduce the influence of outliers [17], the M-estimator of IRLS acts on the coordinates of points directly. Our RDO approach, however, utilizes the characteristics of fitting errors to adjust the weights which are assigned to the components of parameter vectors.

More specifically, our aim is the accurate estimations of parameter vectors through reweighting the different components of parameter vectors in the learning process. We demonstrate the potential of RDO in handling challenging visualization tasks including 3D point cloud registration and multi-view stitching. Experimental results show that RDO outperforms the state-of-the-art algorithms in terms of robustness and accuracy on synthetic data sets and range-scan data sets.

Section 2 reviews the work on point cloud registration 2.1 and supervised sequential update (SSU) methods 2.2. Section 3 introduces our framework and theoretical analysis of RDO. Finally, we demonstrate the better performance of RDO through the experiments of 3D registration and multi-view stitching in Section 4.

2. Previous work

2.1. Point cloud registration

Point cloud registration is a process of associating two sets of data into the same standard coordinate system, a common task in computer vision and graphics (e.g., objects reconstruction, object tracking, etc.). Iterative closest point (ICP) algorithm [18] is the most commonly used method for registration, which finds the best transformation parameters of a group of 3D points through rigid transformation and continuous iteration to minimize the difference between the two point clouds. Due to their conceptual simplicity, high usability and good performance in practice, ICP and its variants are popular methods and have been successfully applied in numerous real-world tasks [19–22]. However, ICP-based methods are sensitive to outliers and need the initialization to be as close to the optimal solution as possible to avoid a bad local minimum. Iteratively Reweighted Least Squares (IRLS) [17] uses various cost functions to improve the robustness to outliers and avoid bad local minima. On the other hand, Normal Distribution Transformation (NDT) [23] applies a statistical model to match 3D point clouds. Coherent Point Drift (CPD) [24] achieves point clouds registration based on a Gaussian mixture model, which moves the Gaussian mixture model centroids coherently as a group to preserve the topological structure of the point sets. Bayesian

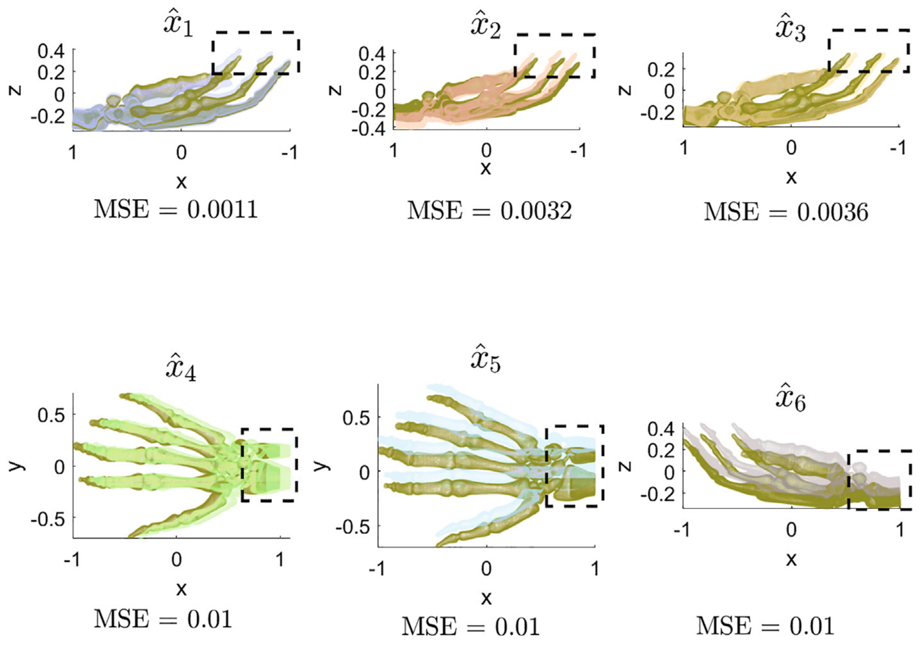


Fig. 1. Registration results with different transformation parameter vectors $\hat{\mathbf{x}}_k, k \in \{1, 2 \cdots 6\}$. $\hat{\mathbf{x}}_k$ is the k-th column of the diagonal matrix with diagonal elements of 0.1. The dark-green represents the original model (the parameter vector $\mathbf{x}_* = \mathbf{0}_6$) and the dotted rectangles show the obvious difference between the original model and the transformed models. It can be seen that each component of the parameter vector can have different impact on the final registration error.

Coherent Point Drift (BCPD) [25] formulates CPD in a bayesian setting and replaces the motion coherence theory in CPD with bayesian inference.

In principle, these algorithms achieve point cloud registration through casting the registration as an optimization problem, using the feature information of point clouds to devise objective functions, adding regularization terms or constraints to objective functions to improve the robustness. However, the specific designing of objective functions of the above algorithms limits the performance of algorithms on registration with a certain perturbation. For example, ICP and IRLS are sensitive to outliers, and the size of the subdivided cells constrains the performance of NDT. CPD and BCPD have low robustness to the registration with occlusion due to coherently moving. Besides, adding additional regularization terms will increase the complexity of optimization models, making it challenging to calculate the derivation of objective functions. Although gradient descent and its variants, such as Newton's method and Quasi-Newton method are commonly used to search optimal solutions of optimization models [26–28], there are a number of challenges in obtaining the optimal gradient information of objective functions, such as the learning rate of the gradient descent is not optimal, the gradient information is not readily available [29], the Hessian matrix may not be positive definite, or convergence rate is slow.

2.2. Supervised sequential update (SSU) methods

Addressing the challenges in calculating the gradient information of objective functions, learning-based optimization leverages supervised sequential update (SSU) methods to learn a sequence of regressors to mimic the gradient directions of objective functions. A cascaded pose regression was proposed in [30] to compute 2D object poses in images. Similarly, a cascaded regression approach [31] was derived from a Gauss–Newton solution to address a nonlinear least squares problem, where the descent direction is a sequence of averaged Jacobian and Hessian matrices learned from data. Cao [32] has developed an "Explicit Shape Regression" method for face alignment by learning a vectorial regression function to infer the whole facial shape from images which explicitly minimizes the alignment error over the training data. Tuzel [33] proposed a learning-based tracking algorithm combined with object detection, where the descent direction was represented by a linear regression function. SDM [13,14] learns a sequence of regression matrices to update shape parameters based on image features at per iteration. DO [34,15] utilizes the least-squares method to attain a sequence of regression matrices that are mapped to the feature of data sets to get the final parameter vectors. The aforementioned works learn a fixed regression or a sequence of regressions based on data features by making the estimated parameter vectors approach to ground truths at per iteration, which avoids calculating the Jacobian and Hessian matrices and improves the robustness of algorithms with the presence of various perturbations. However, that the $\ell 2$ norm of the residual vector approaches to 0 does not represent the best fitting between the estimated and real models because different components of parameter vectors have various impacts on fitting results, as shown in Fig. 1.

In this paper, we propose a Reweighted Discriminative Optimization (RDO) method, which assigns different weights to components of parameter vectors to ensure better fitting results. It is worth noting that the weights in our method are learned according to the characteristics of fitting errors, and the weights work directly on the parameter vectors. Our method is different from the IRLS [16], in which the weights are obtained via the M-estimation criterion and act directly on the model coordinates rather than the transformation parameter vectors.

3. Reweighted Discriminative Optimization

3.1. Motivation from SSU methods

Supervised sequential update (SSU) algorithms predict a set of parameters by sequentially refining previously estimated parameters. The refinement is performed by establishing a sequence of regressors to update the parameters. The specific refinement process can be cast as follows:

$$\mathbf{x}_{t+1} = C(\mathbf{x}_t, \mathbf{F}_{t+1} \circ \mathbf{h}(\mathbf{x}_t)) \tag{1}$$

where $\mathbf{x}_t \in \mathbf{R}^p$ is the estimated parameter vector at time t, $\mathbf{h} : \mathbf{R}^p \rightarrow \mathbf{R}^f$ extracts features from the input data, $\mathbf{F}_{t+1} : \mathbf{R}^f \rightarrow \mathbf{R}^d$ is a regressor that maps the feature $\mathbf{h}(\mathbf{x}_t)$ to an update vector, $C : \mathbf{R}^p \times \mathbf{R}^d \rightarrow \mathbf{R}^p$ is the operator working on the parameter $\mathbf{x}_t$ and the update map $\mathbf{F}_{t+1} \circ \mathbf{h}$.

The regressor $\mathbf{F}_{t+1}$ can be attained by minimizing the residual vector of the estimated vector $\mathbf{x}_{t+1}$ and the ground truth $\mathbf{x}_*$.

$$\mathbf{F}_{t+1} = \min_{\hat{\mathbf{F}}} \sum_{i=1}^{N} \left\| \mathbf{x}^i_{t+1} - \mathbf{x}^i_* \right\|_2^2 = \min_{\hat{\mathbf{F}}} \sum_{i=1}^{N} \left\| C\left(\mathbf{x}^i_t, \hat{\mathbf{F}} \circ \mathbf{h}(\mathbf{x}^i_t)\right) - \mathbf{x}^i_* \right\|_2^2 \tag{2}$$

where N is the number of samples.

Discriminative optimization (DO) algorithm as an advanced SSU algorithm has been applied in 3D point cloud registration. In DO, the refinement process mentioned in Eq. (1) is cast as follows:

$$C(\mathbf{x}_t, \mathbf{F}_{t+1} \circ \mathbf{h}(\mathbf{x}_t)) = \mathbf{x}_t - \mathbf{F}_{t+1} \times \mathbf{h}(\mathbf{x}_t)$$

Learning the regressor $\mathbf{F}_{t+1}$ is posed as the process of approaching the currently estimated vector $\mathbf{x}_{t+1}$ to the ground truth $\mathbf{x}_*$, as shown in Eq. (2). Our method RDO can be regarded as an extension to the DO framework to some extent.

3.2. Motivation from point cloud registration

Let $\mathbf{M}, \mathbf{S}$ be two point sets in a finite-dimensional real vector space $\mathbf{R}^3$, which contains N_m and N_s points respectively. Point cloud registration is to find a spatial transformation $\mathbf{T}$ to be applied to the scene set $\mathbf{S}$ such that the difference between $\mathbf{S}$ and the model set $\mathbf{M}$ is minimized. The rigid rotation matrix is not closed with respect to addition, which limits the direct derivation of the objective function to the rotation matrix. To overcome this challenge, Lie group $SE(3)$[1] is used to represent the transformations in 3D space [35] so that the 3D point cloud registration can be turned into an optimization problem over the Lie algebra $\mathfrak{se}(3)$,[2] where parameter vectors $\mathbf{x}_{t+1}, \mathbf{x}_* \in \mathbf{R}^6$.

Assume $\mathbf{x}_* = \mathbf{0}_6, \mathbf{x}_{t+1} = \hat{\mathbf{x}}_k, \hat{\mathbf{x}}_k$ is the k-th column of the diagonal matrix $\mathbf{A}$.

$$\mathbf{A} = \begin{bmatrix} 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1 \end{bmatrix} = [\hat{\mathbf{x}}_1, \quad \hat{\mathbf{x}}_2, \quad \hat{\mathbf{x}}_3, \quad \hat{\mathbf{x}}_4, \quad \hat{\mathbf{x}}_5, \quad \hat{\mathbf{x}}_6]$$

For any $\hat{\mathbf{x}}_k, k \in \{1, 2 \cdots 6\}$, the $\ell 2$ norm of the residual vector

[1] $SE(3) = \left\{ \mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\mathrm{T} & 1 \end{bmatrix} | \mathbf{R} \in SO(3), \mathbf{t} \in \mathbf{R}^3 \right\} \subset \mathbf{R}^{4\times4}$, where $SO(3)$ is the 3D rotation group.

[2] $\mathfrak{se}(3) = \left\{ \xi = [\boldsymbol{\phi} \boldsymbol{\rho}] | \hat{\phi} \in \mathfrak{so}(3), \boldsymbol{\rho} \in \mathbf{R}^3 \right\} \subset \mathbf{R}^{6\times1}$, where $\mathfrak{so}(3)$ is the corresponding Lie algebra of group $SO(3)$ and $\hat{\phi}$ is a rotational skew-symmetric matrix composed of the elements in vector $\boldsymbol{\phi}$.

$\|\mathbf{x}_* - \hat{\mathbf{x}}_k\|_2^2$ is 0.01. Each $\hat{\mathbf{x}}_k$ as the parameter vector is applied to register the hand model, and the registration results are shown in Fig. 1.

Fig. 1 shows that although the norms of residual vectors $\|\hat{\mathbf{x}}_k - \mathbf{x}_*\|_2^2$ reach the same value 0.01, the mean squared errors of the registration results are different, which means that the components of a parameter vector have different impacts on the registration results.

Fig. 2 illustrates the impact of each component of parameter vectors on transformation. $\hat{\mathbf{x}}_k, k \in \{1,2,3\}$ represent the rotations along with the x-axis, y-axis, and z-axis. $\hat{\mathbf{x}}_k, k \in \{4,5,6\}$ show the translation. It can be seen that each component plays a different role in transforming the dark-green plane. Combining with Fig. 1, we can find that although the components of parameter vectors have changed on the same scale 0.1, the registration error is different, and the $\hat{\mathbf{x}}_k, k \in \{1,2,3\}$ is more robust with regard to perturbations than translation. Therefore, the goal of RDO is to improve the robustness of translation to perturbations and maintain the robustness of rotation.

To reduce the registration error and improve robustness while the estimated parameter vector $\mathbf{x}_{t+1}$ is approaching the ground truth $\mathbf{x}_*$, RDO assigns different weights to the components of the parameter vectors to emphasize the influence of each component of the vectors on the final registration results.

3.3. Sequence of update maps

Let $\mathbf{h} : \mathbf{R}^p \to \mathbf{R}^f$ be a function that encodes features of a data set, and $\mathbf{D}_{t+1} \in \mathbf{R}^{p\times f}$ be a matrix that maps the feature $\mathbf{h}$ to a update vector. Given an initial parameter vector $\mathbf{x}_0 \in \mathbf{R}^p$, the iterative updating process can be defined as follows:

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \mathbf{D}_{t+1}\mathbf{h}(\mathbf{x}_t) \tag{3}$$

The update process ends until $\mathbf{x}_{t+1}$ converges to a stationary point. And the sequence of matrices $\mathbf{D}_{t+1}, t = 0, 1\cdots$ are learned through approximating estimated parameter vector $\mathbf{x}_{t+1}^i$ to the ground truth $\mathbf{x}_*^i$.

$$\begin{aligned}\mathbf{D}_{t+1} &= \min_{\hat{\mathbf{D}}} \frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}_{t+1}^i - \mathbf{x}_*^i\|_2^2 \\ &= \min_{\hat{\mathbf{D}}} \frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{x}_t^i - \hat{\mathbf{D}}\mathbf{h}(\mathbf{x}_t^i) - \mathbf{x}_*^i\right\|_2^2\end{aligned} \tag{4}$$

where N is the number of samples, $\mathbf{x}_t^i$ is the parameter vector of i-th sample at the t -th iteration. For simplicity, we denote $\mathbf{x}_t^i$ as $\mathbf{x}_t$ when the formula/function applies to all samples.

Considering that each component of the parameter vector $\mathbf{x}_t$ can have a different impact on the fitting error, RDO explores a weighting vector $\mathbf{w}_t \in \mathbf{R}^p$ to emphasize the impact of each component of the parameter vector on fitting results.

$$\mathbf{w}_t = [w_1,\ w_2,\ w_3,\ \cdots\ w_p]^{\mathrm{T}}$$

$$\begin{aligned}\mathbf{D}_{t+1} &= \min_{\hat{\mathbf{D}}} \frac{1}{N}\sum_{i=1}^{N}\|\mathbf{w}_t \odot (\mathbf{x}_{t+1}^i - \mathbf{x}_*^i)\|_2^2 \\ &= \min_{\hat{\mathbf{D}}} \frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{w}_t \odot \left(\mathbf{x}_t^i - \hat{\mathbf{D}}\mathbf{h}(\mathbf{x}_t^i) - \mathbf{x}_*^i\right)\right\|_2^2\end{aligned} \tag{5}$$

where $\odot$ represents Hadamard product. This weighting vector $\mathbf{w}_t$ can be transformed into a weighting diagonal matrix $\mathbf{W}_t \in \mathbf{R}^{p\times p}$.

$$\mathbf{W}_t = \begin{bmatrix} w_1 & 0\dots & 0 \\ 0 & w_2\dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & w_p \end{bmatrix}_{p\times p}$$

Then, we get

$$\begin{aligned}\mathbf{D}_{t+1} &= \min_{\hat{\mathbf{D}}} \frac{1}{N}\sum_{i=1}^{N}\|\mathbf{W}_t (\mathbf{x}_{t+1}^i - \mathbf{x}_*^i)\|_2^2 \\ &= \min_{\hat{\mathbf{D}}} \frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{W}_t \left(\mathbf{x}_t^i - \hat{\mathbf{D}}\mathbf{h}(\mathbf{x}_t^i) - \mathbf{x}_*^i\right)\right\|_2^2\end{aligned} \tag{6}$$

3.4. Design weighting matrix $\mathbf{W}_t$

Before designing weights for components of parameter vectors, we explored the function of weights on transformation, as shown in Fig. 3. It can be seen that the weights control the scale of transformation. The greater the weight, the larger the scale of the transformation.

We have known that the weights control transformation through acting on components, in order to determine which component is more important and should be allocated a larger weight, we introduce a 'detector' $\tilde{\mathbf{x}}_{t_k}$ to 'probe' the difference between the component of the current estimation $\mathbf{x}_t$ and that of $\mathbf{x}_*$.

$$\begin{aligned}\mathbf{C} &= \begin{bmatrix} c & 0\dots & 0 \\ 0 & c\dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & 0\dots & c \end{bmatrix} \\ &= [\mathbf{c}_1,\ \mathbf{c}_2,\ \cdots\ \mathbf{c}_p]_{p\times p}\end{aligned}$$

The element c of the diagonal matrix $\mathbf{C}$ is a constant. $\mathbf{c}_k \in \mathbf{R}^{p\times 1}$ is the k-th column of matrix $\mathbf{C}$.

$$\tilde{\mathbf{x}}_{t_k} = \mathbf{x}_t - \mathbf{c}_k,\ \ k \in \{1, 2, \cdots, p\} \tag{7}$$

where $\tilde{\mathbf{x}}_{t_k}$ is the changed parameter vector. Except for the k-th element, $\tilde{\mathbf{x}}_{t_k}$ is the same as $\mathbf{x}_t$.

The matching error between the 'detector' and ground-truth is the basis to assign weights. The greater error means the larger difference between the k_{th} component of the current estimation $\mathbf{x}_t$ and that of ground-truth $\mathbf{x}_*$. We need to assign a greater weight to reduce the difference. That is to say that the matching error determines the weights. And the weights act on the component of transformation vectors, then influence the transformation scale that determines the matching error. The relationship between weights and matching error is provided in Fig. 4. The principle to design the weights is that the weight monotonically increases as the matching error increases, which guarantees that larger weights corresponding to the greater matching error.

The weight w_k depends on err_k^i, which is the fitting error of the i-th 'detector' on the k-th dimension of the parameter vector space. err_k^i involves the parameter vector pair $\left\langle \tilde{\mathbf{x}}_{t_k}^i, \mathbf{x}_*^i \right\rangle$.

$$err_k^i = \frac{1}{N_{q_i}}\sum_{j=1}^{N_{q_i}}\left\|F\left(\tilde{\mathbf{x}}_{t_k}^i, \mathbf{q}_j^i\right) - F\left(\mathbf{x}_*^i, \mathbf{q}_j^i\right)\right\|_2^2 \tag{8}$$

where $\mathbf{q}_j^i$ represents the j-th elements of the i-th sample $\mathbf{Q}_i$, which can be the j-th point of the i-th point cloud $\mathbf{Q}_i$; F is a function that applies the parameter vector to $\mathbf{q}_j^i$.

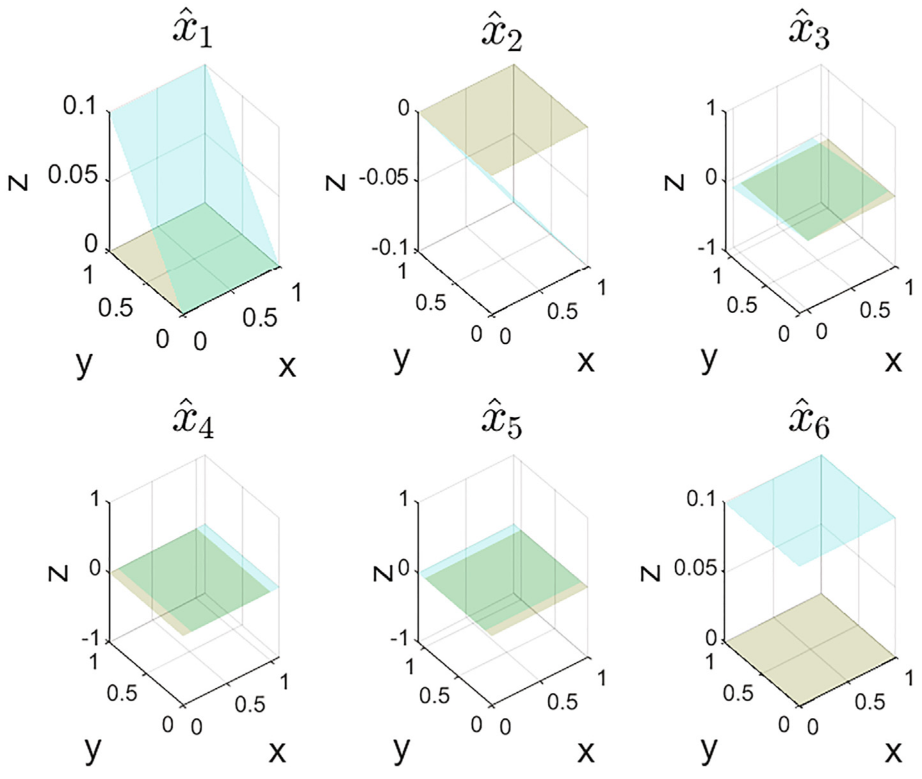


Fig. 2. The transformation with $\hat{\mathbf{x}}_k, k \in \{1, 2 \cdots 6\}$.

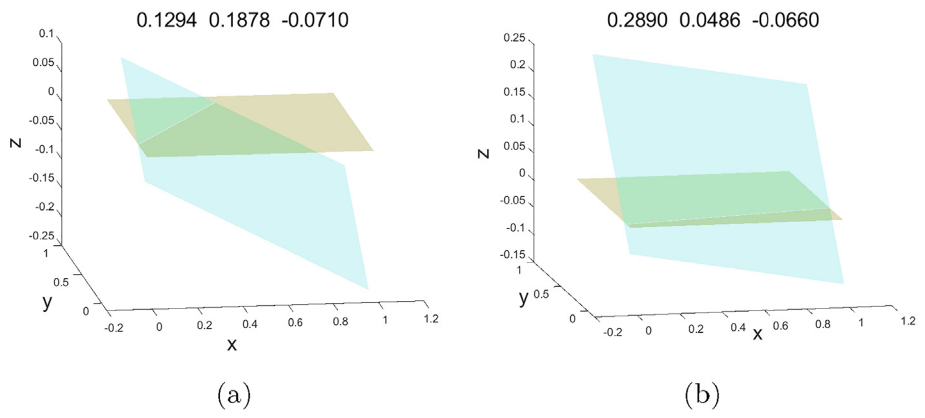


Fig. 3. The transformation with different weights. (a) shows the transformation process from the dark-green to green when $\mathbf{w}_t = [0.2, 0.6, 0.1, 1/30, 1/30, 1/30]^{\mathrm{T}}$. The element of the vector at the top of the sub-figure $[0.1294, 0.1878, -0.0710]$ corresponds to the rotation degree along with the X, Y, Z axes respectively. (b) illustrates the rotation information with the weights $[0.6, 0.2, 0.1, 1/30, 1/30, 1/30]$. We can find that under the influence of weights, the dark-green in (b) mainly rotates along with the X-axis rather than the y-axis that is the main axis for rotation in (a).

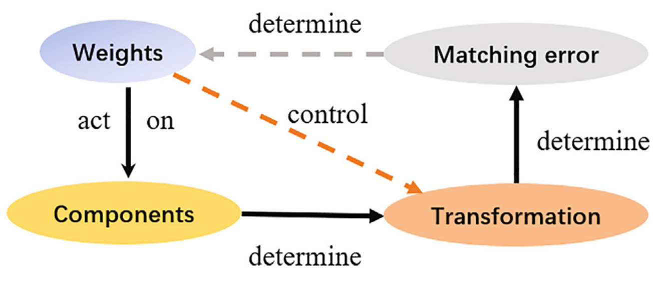


Fig. 4. The relationship between weights and matching error.

The fitting error of the $\mathbf{Q}_i$ on the whole parameter vector space is $\mathbf{err}^i = \left[err^i_1,\ err^i_2,\ \cdots\ err^i_p\right]^{\mathrm{T}}$.

$$Err^i_k = \frac{err^i_k - \min(\mathbf{err}^i)}{\max(\mathbf{err}^i) - \min(\mathbf{err}^i)} \tag{9}$$

$\mathbf{Err}^i = \left[Err^i_1,\ Err^i_2,\ \cdots\ Err^i_p\right]^{\mathrm{T}}$ refers to the normalization of vector $\mathbf{err}^i$. $\mathbf{E}$ is a $N \times p$ matrix made up of vector $\mathbf{Err}^i$. N is the number of samples. We express $[\mathbf{E}]_k$ as the k-th row of matrix $\mathbf{E}$, $[\mathbf{E}]_{:,k}$ as the k-th column of matrix $\mathbf{E}$.

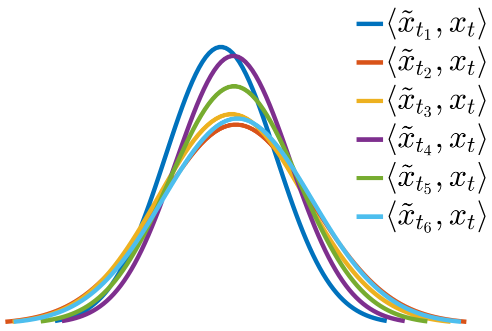


Fig. 5. Normal distributions of registration errors with parameter vectors pairs $\langle \tilde{\mathbf{x}}_{t_k}, \mathbf{x}_* \rangle$, $k \in \{1, 2, \cdots, p\}$. Curves with different colors represent different distributions.

We use statistical skills to get the normal distributions of the registration errors of point clouds on each dimension of the parameter vector space, as shown in Fig. 5. The normal distribution is plotted by fitting the histogram of the fitting errors $[\mathbf{E}]_{:,k}, k = 1, 2 \cdots p$. p is 6 for Lie-algebra of transformation matrices. It can be seen that the distributions corresponding to the different parameter vector pair $\langle \tilde{\mathbf{x}}_{t_k}, \mathbf{x}_* \rangle$ are different.

Weights w_k is calculated according to the characteristics of normal distributions of errors.

$$\begin{aligned} \mu_k &= \frac{\sum_{i=1}^{N} [\mathbf{E}]_{i,k}}{N} \\ \sigma_k^2 &= \frac{\sum_{i=1}^{N} \left([\mathbf{E}]_{i,k} - \mu_k\right)^2}{N} \end{aligned} \tag{10}$$

$$f(\mu_k) = \frac{1}{\sigma_l \sqrt{2\pi}} \exp(-\frac{1}{2\sigma_l^2} (\mu_k - \mu_l)^2) \tag{11}$$

where μ_k and σ_k^2 are the mean and variance of the fitting errors of all samples on the k-th dimension of the parameter vector.

Fig. 6 shows the criteria for designing weights, $\mu_1 < \mu_2 < \mu_3, \delta_3 < \delta_1 < \delta_2$. The orange circles are the $f(\mu_1)$ and $f(\mu_3)$, where the $\mu_l = \mu_2$. It can be seen that although $\mu_1 < \mu_3$, the $f(\mu_1) \approx f(\mu_3)$, which does not satisfy the principle of designing weights. The blue plus signs illustrate the $f(\mu_2)$ and $f(\mu_3)$, where the $\mu_l = \mu_1$. And the green squares represent the $f(\mu_1)$ and $f(\mu_2)$, where the $\mu_l = \mu_3$. The latter two cases ensure the monotonicity of weight assignment. The black lines show the difference between $f(\mu_2)$ and $f(\mu_3)$,$f(\mu_2)$ and $f(\mu_1)$. It can be seen that the difference between $f(\mu_2)$ and $f(\mu_1)$ is larger than the difference between $f(\mu_2)$ and $f(\mu_3)$.

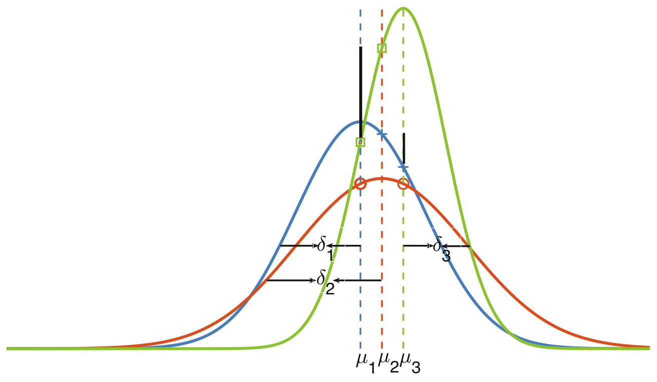


Fig. 6. The criteria for designing weights.

Assume $\mu_1 < \mu_2 < \mu_3 < \mu_4 < \mu_5 < \mu_6$, we choose μ_1 and μ_6 as the candidates for μ_l due to the monotonicity of weight assignment.

if $\delta_1 < \delta_6,\ \mu_l = \mu_1$

$$w_k = \exp(\frac{1}{2} (1 - f(\mu_k))^t) \tag{12}$$

if $\delta_1 > \delta_6,\ \mu_l = \mu_6$

$$w_k = \exp(\frac{1}{2} (f(\mu_k))^t) \tag{13}$$

Designing weights in this way guarantees monotonicity and makes sure that the difference between weights is sufficiently to reflect the difference between matching errors.

3.5. Learning a SUM

Suppose we are given a set of training data $\{(\mathbf{x}_0^i, \mathbf{x}_*^i, \mathbf{h}(\mathbf{x}_0^i), \mathbf{Q}_i)\}_{i=1}^N$, including N samples $\mathbf{Q}_i$, where $\mathbf{x}_0^i \in \mathbf{R}^p$ is the initial parameter vector of the i-th sample (*e.g.* the i-th points cloud), $\mathbf{x}_*^i \in \mathbf{R}^p$ is the ground truth (*e.g.* the transformation parameters), $\mathbf{h} : \mathbf{R}^p \rightarrow \mathbf{R}^f$ provides feature information of samples. For simplicity, we denote $\mathbf{h}(\mathbf{x}_t^i)$ as $\mathbf{h}_t^i$ to represent the feature of the i-th sample $\mathbf{Q}_i$ at the t-th iteration.

The goal of RDO is to learn a sequence of maps $\mathbf{D}_{t+1}$ that update $\mathbf{x}_0^i$to $\mathbf{x}_*^i$.

$$\mathbf{D}_{t+1} = \min_{\hat{\mathbf{D}}} \frac{1}{N} \sum_{i=1}^{N} \left\| \mathbf{W}_t \left(\mathbf{x}_*^i - \mathbf{x}_t^i + \hat{\mathbf{D}} \mathbf{h}_t^i \right) \right\|_2^2 \tag{14}$$

In practice, to prevent over fitting, ridge regression is used to learn the maps:

$$\mathbf{D}_{t+1} = \min_{\hat{\mathbf{D}}} \frac{1}{N} \sum_{i=1}^{N} \left\| \mathbf{W}_t \left(\mathbf{x}_*^i - \mathbf{x}_t^i + \hat{\mathbf{D}} \mathbf{h}_t^i \right) \right\|_2^2 + \lambda \left\| \hat{\mathbf{D}} \right\|_F^2 \tag{15}$$

where $\|\cdot\|_F$ is the Frobenius norm, and λ is a hyperparameter.

The initial training data $\left\{ (\mathbf{x}_0^i, \mathbf{x}_*^i, \mathbf{h}_0^i, \mathbf{Q}_i) \right\}_{i=1}^N$ is applied to (7)–(12) to get an initial weighting matrix $\mathbf{W}_0$ at first. Next, an initial update map $\mathbf{D}_1$ can be learned by applying the initial training data $\left\{ \left(\mathbf{x}_0^i, \mathbf{x}_*^i, \mathbf{h}_0^i, \mathbf{Q}_i \right) \right\}_{i=1}^N$ and the weighting matrix $\mathbf{W}_0$ to (15), then $\mathbf{D}_1$ will be applied to (3) to get the current estimation $\mathbf{x}_1$. At each step, a new parameter vector can be created by recursively applying the update rule (15) to (3). The learning process is repeated until some termination criteria are met, such as until the fitting error does not decrease much or the maximum number of iterations is reached. The pseudocode for training a sequence of update maps is shown in Algorithm 1.

Algorithm 1. Training a sequence of update maps

```
Input $\left\{ \left( \mathbf{x}_0^i, \mathbf{x}_*^i, \mathbf{h}_0^i, \mathbf{M}_i \right) \right\}_{i=1}^N, T, \lambda, c$
Output $\{\mathbf{D}_{t+1}\}_{t=0}^{T-1}$
1: for $t = 0$ to $T - 1$ do
2:     Compute $\mathbf{W}_t$ with (7)–(12)
3:     Compute $\mathbf{D}_{t+1}$ with (15)
4:     for $i = 1$ to $N$ do
5:         Update $\mathbf{x}_{t+1}^i := \mathbf{x}_t^i - \mathbf{D}_{t+1} \mathbf{h}_t^i$
6:     end for
7: end for
```

The detailed information about all notations is provided in supplemental materials.

3.6. Convergence of training error

Theorem 1 (*Convergence of RDO's training error*). *Given a training set* $\left\{\left(x_0^i, x_*^i, \mathbf{h}_0^i\right)\right\}_{i=1}^N$, *if there exists a linear map* $\hat{\mathbf{D}} \in \mathbf{R}^{p\times f}$ *where* $\hat{\mathbf{D}}\mathbf{h}_t^i$ *meets the condition* $\sum_{i=1}^N (\mathbf{x}_t^i - \mathbf{x}_*^i)^{\mathrm{T}}\hat{\mathbf{D}}\mathbf{h}_t^i > 0$*at* $\mathbf{x}_*^i$ *for all i where* $\left[\mathbf{h}_t^i\right]_{j,:} \in (0,1)$, *and if there exists an i where* $\mathbf{x}_t^i \neq \mathbf{x}_*^i$, *then the update rule:*

$$\mathbf{D}_{t+1} = \min_{\hat{\mathbf{D}}} \frac{1}{N}\sum_{i=1}^N \left\|\mathbf{W}_t\left(\mathbf{x}_*^i - \mathbf{x}_t^i + \hat{\mathbf{D}}\mathbf{h}_t^i\right)\right\|_2^2 + \lambda\left\|\hat{\mathbf{D}}\right\|_F^2 \tag{16}$$

where $[\mathbf{W}_t]_{j,j} > 1$

$$\mathbf{x}_{t+1}^i = \mathbf{x}_t^i - \mathbf{D}_{t+1}\mathbf{h}_t^i \tag{17}$$

guarantees that the training error strictly decreases in each iteration:

$$\sum_{i=1}^N \|\mathbf{x}_*^i - \mathbf{x}_{t+1}^i\|_2^2 < \sum_{i=1}^N \|\mathbf{x}_*^i - \mathbf{x}_t^i\|_2^2 \tag{18}$$

If $\hat{\mathbf{D}}\mathbf{h}_t^i$ *is strongly monotone at* $\mathbf{x}_*^i$, *and if there exist* $H > 0, M > 0$ *such that* $\left\|\hat{\mathbf{D}}\mathbf{h}_t^i\right\|_2^2 \leqslant H + M\|\mathbf{x}_*^i - \mathbf{x}_t^i\|_2^2$ *for all i, then the training error converges to zero.*

The proof of Theorem 1 is provided as the supplemental materials. Theorem 1 says that for all samples, if $\hat{\mathbf{D}}\mathbf{h}_t^i$ meets the condition $\sum_{i=1}^N (\mathbf{x}_t^i - \mathbf{x}_*^i)^{\mathrm{T}}\hat{\mathbf{D}}\mathbf{h}_t^i > 0$, then the average training error will decrease in each iteration; if $\hat{\mathbf{D}}\mathbf{h}_t^i$ is strongly monotone at $\mathbf{x}_*^i$, the average training error will converge to zero. [34] also presents a similar convergence result for an update rule, but it requires $\hat{\mathbf{D}}\mathbf{h}_t^i$to be strictly monotone at $\mathbf{x}_*^i$ for all i.

4. Experimentation

This section describes how to apply RDO to 3D point cloud registration (Synthetic data and Range-scan data) and stitching. We provide a comparative study of RDO with other classical registration methods on different data sets.

4.1. Experimental design

4.1.1. Data sets

We performed 3D points cloud registration experiments on synthetic data sets: Happy Buddha model [36], Skeleton Hand model [37], Dancing Children and Bimba Model http://visionair.ge.imati.cnr/ (as shown in $(a) \sim (d)$ of Fig. 7). And the registration experiment on Range-scan data is conducted on the UWA data set [38] (as shown in $(e), (f)$ of Fig. 7). Multi-view points clouds for the stitching experiment are as shown in Fig. 8. We also compare the performance of RDO and the advanced deep-learning-based registration method PointnetLK [39] on the ModelNet40 dataset [40], as shown in Fig. 9.

4.1.2. Design h

Let $\mathbf{M} \in \mathbf{R}^{3\times N_M}$ be a matrix containing the 3D coordinates of the original model and $\mathbf{S} \in \mathbf{R}^{3\times N_S}$ for the scene model. Our method is applied to register $\mathbf{S}$ and $\mathbf{M}$, where the transformation matrix is represented by Lie algebra $\mathfrak{se}(3)$ $\mathbf{x} \in \mathbf{R}^6$. We use the feature extraction method in [34] to extract the feature of the data sets, which designs $\mathbf{h}$ to be a histogram indicating the weights of scene points on the 'front' and the 'back' sides of each model point according to the coordinates in Fig. 10.

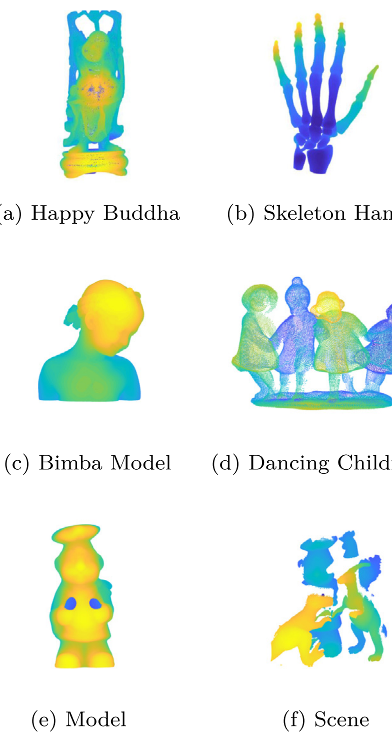


Fig. 7. 3D registration data sets. $(a) \sim (d)$ are the Synthetic data, and $(e), (f)$ are the Range-scan data.

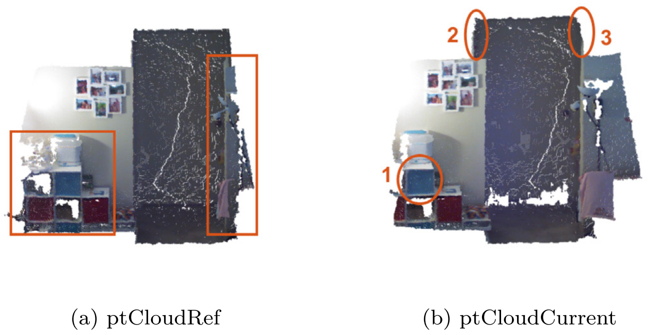


Fig. 8. Point Cloud stitching experiments data sets. The rectangles show the major differences between two different views. The ellipses marked by 1,2,3 are the compared parts in stitching experimental results.

$$\mathbf{S}_a^+ = \{\mathbf{s}_b : \mathbf{n}_a^T(F(\mathbf{s}_b;\mathbf{x}) - \mathbf{m}_a) > 0\} \tag{19}$$

$\mathbf{S}_a^+$ indicates the set of scene points on the 'front' of model point $\mathbf{m}_a$, and $\mathbf{S}_a^-$contains the remaining scene points; $\mathbf{n}_a \in \mathbf{R}^3$ is the normal vector of the model point $\mathbf{m}_a$; $F(\mathbf{s}_b;\mathbf{x})$ is the function that applies rigid transformation with parameter $\mathbf{x}$ to scene point $\mathbf{s}_b$. Then the features in different divided areas of the model point $\mathbf{m}_a$ can be calculated through the following formulas:

$$[\mathbf{h}(\mathbf{x};\mathbf{S})]_{a^+} = \frac{1}{z}\sum_{\mathbf{s}_b\in\mathbf{S}_a^+} \exp\left(\frac{-1}{\beta^2}\|F(\mathbf{s}_b;\mathbf{x}) - \mathbf{m}_a\|^2\right) \tag{20}$$

$$[\mathbf{h}(\mathbf{x};\mathbf{S})]_{a^-} = \frac{1}{z}\sum_{\mathbf{s}_b\in\mathbf{S}_a^-} \exp\left(\frac{-1}{\beta^2}\|F(\mathbf{s}_b;\mathbf{x}) - \mathbf{m}_a\|^2\right) \tag{21}$$

z normalizes $\mathbf{h}$ to sum to 1, and β controls the width of the exp function. $\mathbf{h}$ is specific to model $\mathbf{M}$, and it has a fixed size $2\mathbf{N}_M$.

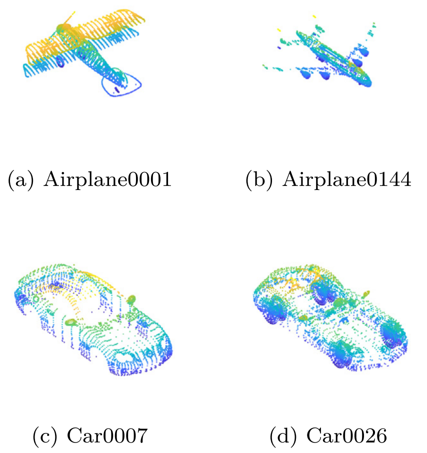


Fig. 9. 3D point clouds for comparing the PointnetLK and RDO registration methods.

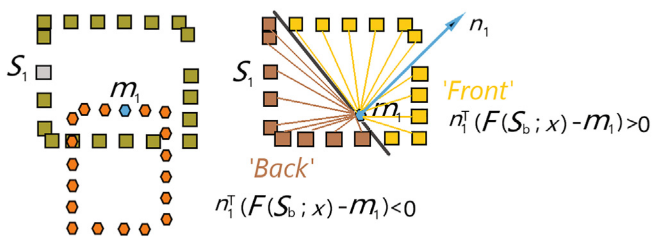


Fig. 10. The positional relationship between scene points (square) and model point (hexagon) $\mathbf{m}_1$.

4.2. RDO training

The parameters in the RDO training process are the same as those in the code provided in the Github of DO [34]. We normalize a given model shape $\mathbf{M}$ to $[-1,1]^3$ and uniformly sample from $\mathbf{M}$ with the replacement 400 to 700 points to generate a scene model. Then we apply the following perturbations to the scene model: *(i) Rotation and translation:* The rotation is within 60 degrees, and the translation is in $[-0.3,0.3]^3$, which represents ground truth $\mathbf{x}_*$; *(ii) Noise and Outliers:* Gaussian noise with the standard deviation 0.05 is added to the scene model. 0 to 300 points within $[-1.5,1.5]^3$ are added as the sparse outliers. Besides, a Gaussian ball of 0 to 200 points with the standard deviation of 0.1 to 0.25 is used to simulate the structured outliers; *(iii) incomplete shape:* We remove 40% to 90% points from the scene model to simulate occlusions, the detailed removing approach can be found in [34]. For all experiments, we generated 30000 training samples, set up iterations $T=30$ and set λ as 2×10^{-4}, β^2 as 0.03, and the initial transformation $\mathbf{x}_0$ is $\mathbf{0}^6$.

4.3. Experiments metrics

4.3.1. Baselines

We compared RDO with the advanced learning-based approach DO [34] and deep-learning-based method PointnetLK [39], two point-based approaches (ICP [18] and IRLS [17]) and three density-based approaches (CPD [24], BCPD [25] and NDT [23]). The codes for all methods were downloaded from the authors' websites, except for ICP, where we used MATLAB's implementation. For IRLS, the Huber cost function was used. The code of RDO was implemented in MATLAB.

4.3.2. Evaluation metrics

We use the registration success rate, the average MSE, and the computation time as performance metrics for comparing the per-

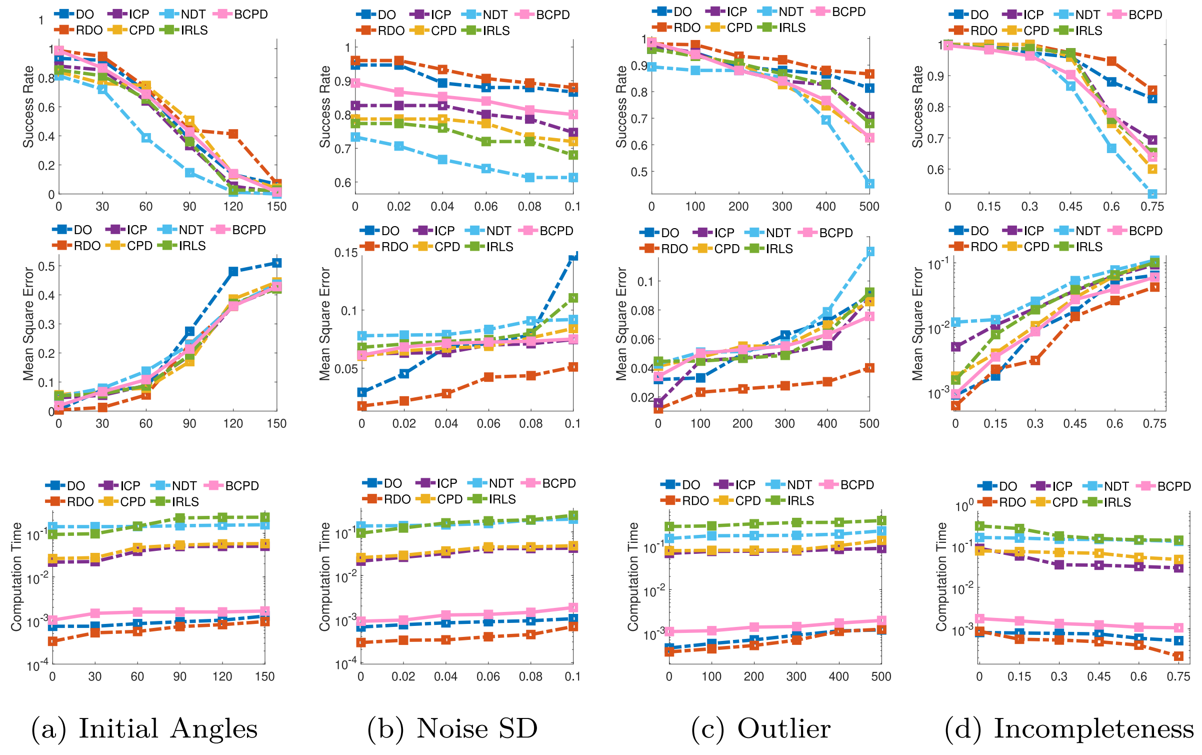


Fig. 11. Results of 3D registration with Happy model under different perturbations; (Top) Success Rate (SR). (Second row) Average MSE (AMSE). (Bottom) Computation Time.

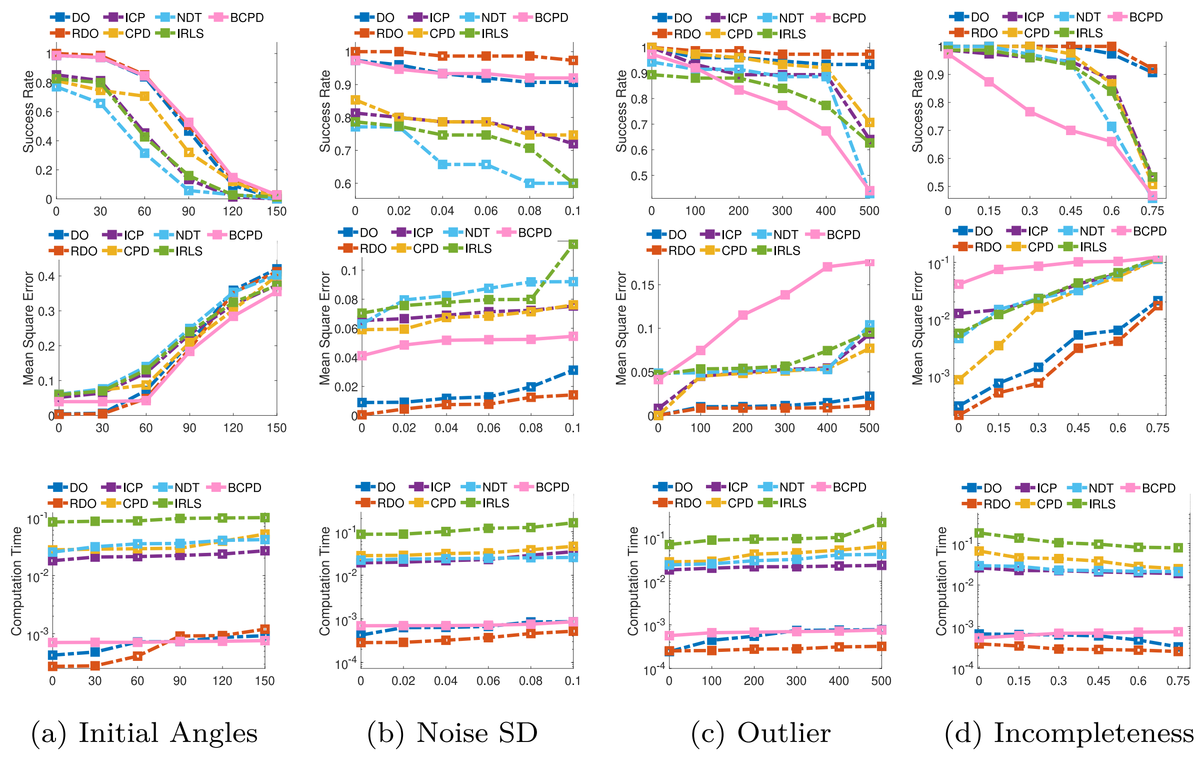


Fig. 12. Results of 3D registration with Hand model under different perturbations; (Top) Success Rate (SR). (Second row) Average MSE (AMSE). (Bottom) Computation Time.

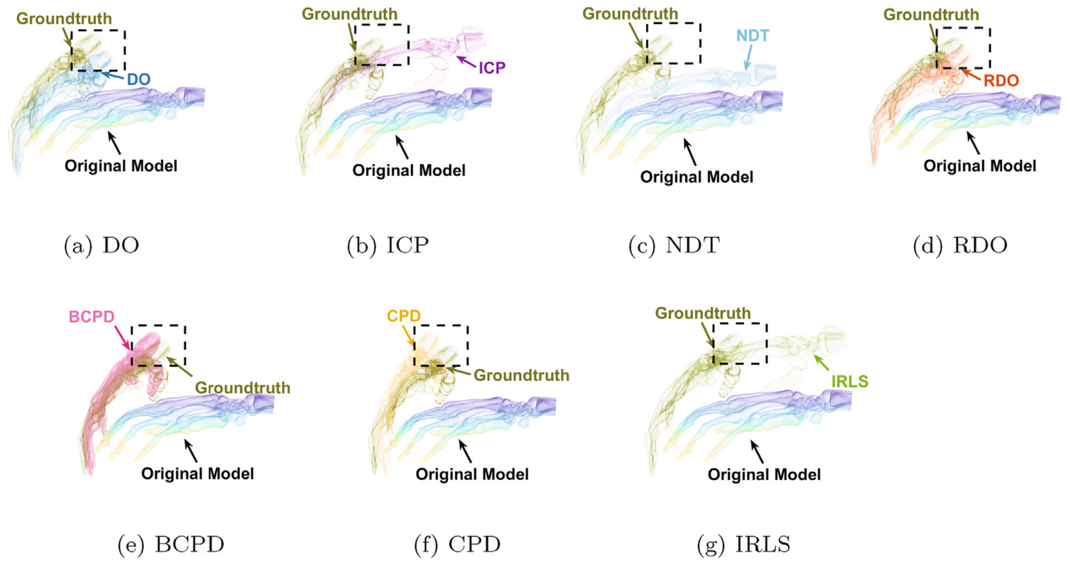


Fig. 13. Registration results of Hand model with 60° rotation; different colors show the registration results of different registration algorithms. The rectangles illustrate the obvious registration difference. Learning-based registration algorithms (DO, RDO) produce more accurate point cloud registration results than that of traditional registration algorithms (ICP, NDT, CPD, BCPD and IRLS).

formance of RDO with registration methods (DO, ICP, NDT, CPD, BCPD, and IRLS). And registration success rate, the MSE, and the log_{10} average MSE are used for RDO and PoinnetLK comparison.

Registration Success Rate. Registration is successful when the mean ℓ_2 is less than 0.05 of the model's largest dimension.

Average MSE. It is worth noting that the MSE is the mean ℓ_2 error between the model and the scene sets, and the Average MSE is the average for MSE for all test sets.

4.4. Parameters settings

The maximum number of iterations of all registration methods was set to 30. For *DO* and *RDO*, we set λ as 2×10^{-4}, β^2 as 0.03. The value of the tolerance of absolute difference between current estimation and ground truth in iterations is 1e-4; For *ICP*, the tolerance of absolute difference in translation and rotation is 0.01 and 0.5, respectively; For *IRLS*, we used *Huber criterion function* as the

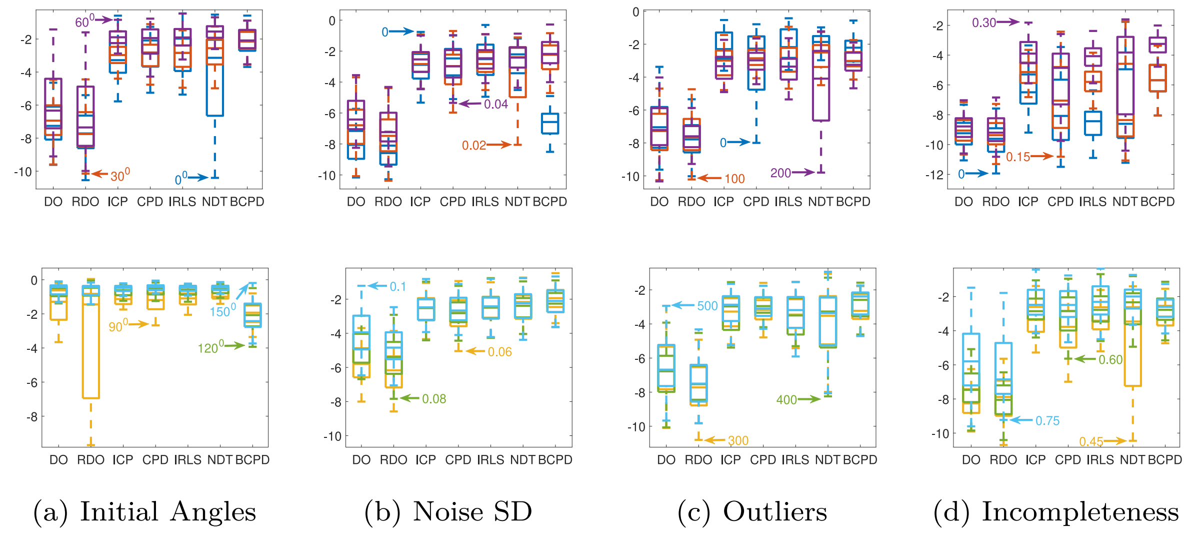


Fig. 14. Statistical results of 3D registration with the Bimba Model under different perturbations; each column shows the registration results in the presence of various perturbations with different degrees. The X-axis represents different registration algorithms; the y-axis shows the $\log_e$ value of the registration error of samples. The values with different colors represent different degrees of perturbations.

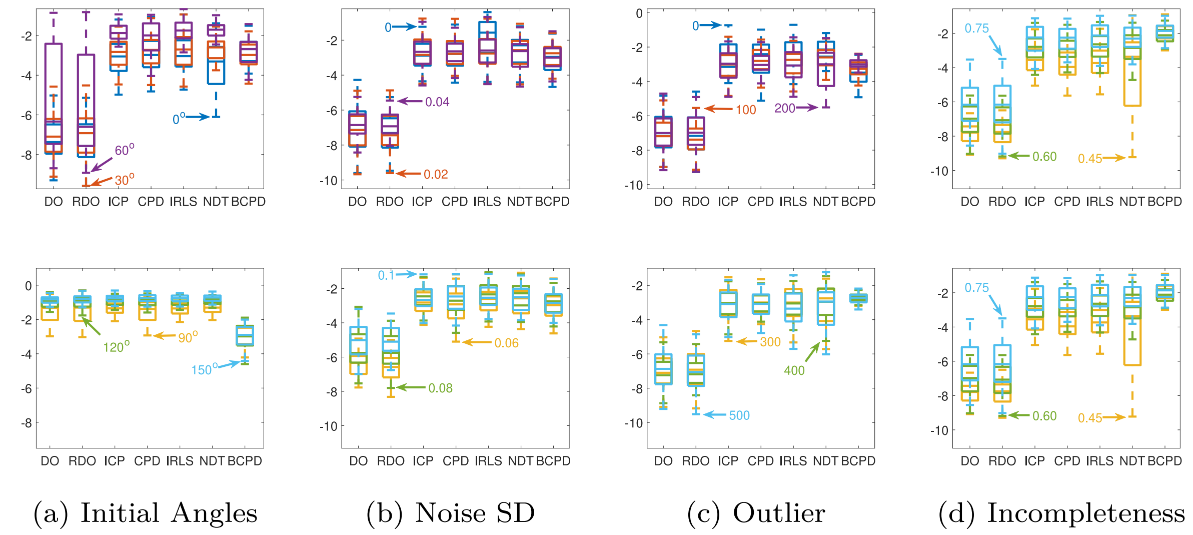


Fig. 15. Statistical results of 3D registration with Dancing Children model under different perturbations; The inter-quartile ranges in Box plots illustrate the high robustness of learning-based algorithms (DO, RDO); the minimum, maximum, quartiles, and the skewness of Box plots show that RDO has better performance than DO.

regression function, the remaining parameters were set as the same as the setting of *ICP*. For *CPD*, the type of transformation is set to *rigid*, and the expected percentage of outliers with respect to a normal distribution is 0.1; the tolerance value is the same as that in *DO*. For *NDT*, the value of the expected percentage of outliers is set to 0.55, and the tolerance value is set as the same as that in *ICP*. For *BCPD*, the expected percentage of outliers is set to 0.3, the parameter in the Gaussian kernel is 0.2, and the expected length of the displacement vector is 1e9. For *PointnetLK*, the PointnetLK network is trained on an Nvidia Geforce 2080Ti GPU with 12G memory. The kernel sizes of the PointnetLK are 64, 64, 64, 128, 1024. The maximum iteration for rotation and translation is set to 30. Adam optimizer with an initial learning rate of 0.001, 250 epochs and a batch size of 10 are used for the training process.

4.5. Registration experiments

4.5.1. Synthetic data

We perform registration experiments on Happy Buddha model, Skeleton Hand model, Dancing Children and Bimba Model in Fig. 7. The models are downsampled by selecting 477 points from the original model as the model set **M**. The performance of algorithms are evaluated by comparing the evaluation metrics in the case of various perturbations: *(1) rotation:* The initial angle is $0^\circ, 30^\circ, 60^\circ, 90^\circ, 120^\circ$ and 150° [default=0° to 60°]; *(2) noise:* The standard deviation of Gaussian noise is set to 0, 0.02, 0.04, 0.06, 0.08 and 0.1 [default = 0]; *(3) outlier:* We set the number of outliers to 0, 100, 200, 300, 400 and 500, respectively [default = 0]; *(4) incomplete ratio:* The ratio of incomplete scene shape is set to 0, 0.12, 0.243, 0.36, 0.48 and 0.6 [default = 0]. The random translation of all generated scenes is within $[-0.3, 0.3]^3$. It is worth noting that when we change one parameter, the values of other parameters are fixed to the default value. In addition, the scene points are sampled from the original model, not from **M**. We will test 750 test samples in each variable setting. The final results are shown in Figs. 11, 12, 14 and 15.

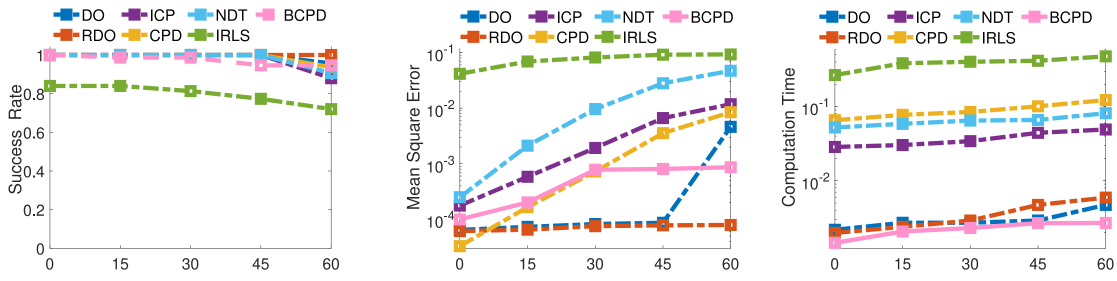
4.5.2. Range-scan data

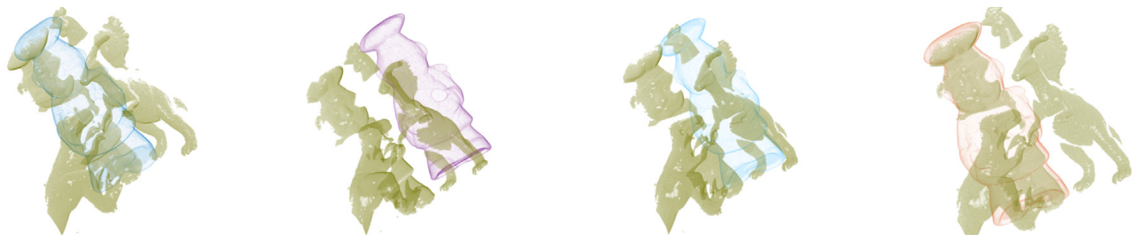
We perform a registration experiment on the UWA Dataset in Fig. 7. This dataset contains 50 cluttered scenes with five objects taken with the Minolta Vivid 910 scanner in various configurations. All objects are heavily occluded (60% to 90%). From the original model, ~400 points were sampled using *pcdownsample* and used as model **M**. We also downsampled the scene to ~1000 points. We initialized the model from 0 to 60 degrees from the ground truth orientation with random translation within $[-0.3, 0.3]^3$. We ran 75 initializations for each parameter setting, and we set the inlier ratio of ICP to 50% as an estimate for self-occlusion. The final result is shown in Fig. 16.

4.5.3. ModelNet40 Data set

We perform registration experiments on ModelNet40 Data set in Fig. 9. For *PointnetLK training*, We train the PointnetLK network on 20 categories of public ModelNet40 data set, and all 3D point clouds are downsampled to 1024 points during training. For *RDO training*, there are two training schemes: single-class training and multi-class training. The former is to train RDO on each point cloud (such as Airplane0001), and the latter is to train RDO on all data sets (four point clouds) in Fig. 9. The following perturbation setting is adaptable for RDO training (single-class and multi-class) and PointnetLK training. There are three kinds of perturbation setting modes. (i)$mode_1$: The rotation is within 45 degrees and the translations is in $[-0.3, 0.3]^3$; (ii) $mode_2$: The rotation is within 60 degrees and the translations is in $[-0.3, 0.3]^3$; (iii) $mode_3$: The rotation is within 45 degrees, the translations is in $[-0.3, 0.3]^3$ and Gaussian noise with the standard deviation 0.05 is also applied.

The performance of algorithms are evaluated by comparing the evaluation metrics in the case of various perturbations: for $mode_1$: the initial angle is $0^\circ, 15^\circ, 30^\circ, 45^\circ, 60^\circ$ and 75°; for $mode_2$: the initial angle is $0^\circ, 30^\circ, 60^\circ, 90^\circ, 120^\circ$ and 150°; for $mode_3$: the initial angle is $0^\circ, 15^\circ, 30^\circ, 45^\circ, 60^\circ$ and 75° [default=0° to 45°] and the standard deviation of Gaussian noise is set to 0, 0.02, 0.04, 0.06, 0.08 and 0.1 [default = 0]. It is worth noting that when we change one parameter, the values of other parameters are fixed to the default value. We will test 100 test samples in each variable setting. The registration results with the single-class training scheme are shown in Figs. 17, 19, 21 and 23. The registration results with




(a) DO (b) ICP (c) NDT (d) RDO

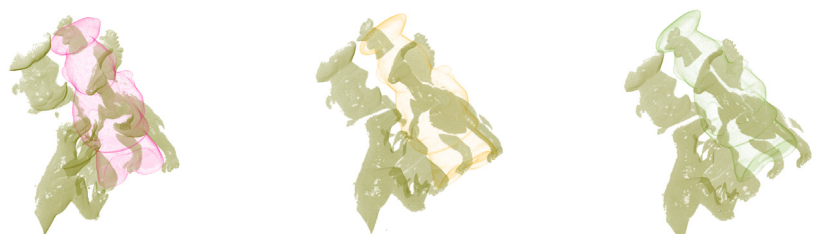
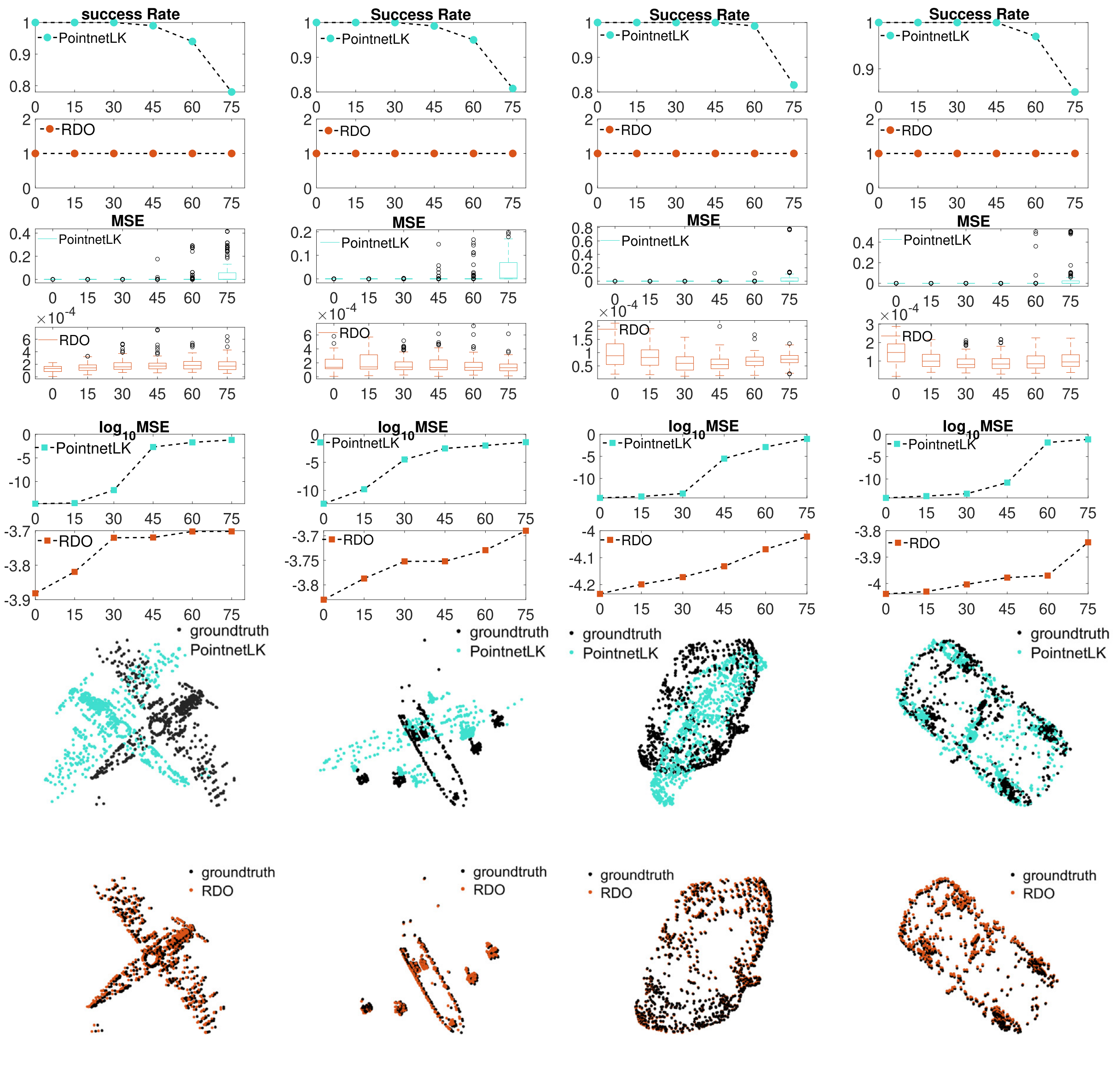

(e) BCPD (f) CPD (g) IRLS

Fig. 16. Results of the registration on the UWA dataset. The first row shows the evaluation results of the performance of various methods on the UWA dataset. And the second and the third rows display the registration results of the UWA dataset with 60° rotation; different colors show the registration results of different registration algorithms.



(a) Airplane0001 (b) Airplane0144 (c) Car0007 (d) Car0026

Fig. 17. The registration results of the single-class training scheme with perturbation setting *mode*$_1$. (Top) shows the Success Rate. (Second) shows the Mean Square Error (MSE). (Third) shows the *log*$_{10}$ average MSE. (Bottom) displays the registration results of ModelNet40 with 60° rotation, and the turquoise shows the registration result of PointnetLK, and the dark orange illustrates the registration results of RDO. It can be seen that RDO has better performance than PointnetLK.

the multi-class training scheme are shown in Figs. 18, 20, 22 and 24.

4.6. Stitching experiments

We perform a multi-view points cloud stitching experiment on the data set in Fig. 8, which stitches together a collection of point clouds that were captured with a Kinect to construct a larger 3D view of the scene. To align the two point clouds, we regard the first point cloud as the reference and apply the transformation parameters to the second point cloud. ∼400 points of the reference model were sampled using *pcdownsample* and used as the model **M**. We also downsampled the second point cloud to ∼ 1000 points. We initialize the reference model from 0 to 15 degrees with random translations within $[-0.1, 0.1]^3$. It is worth noting that after attaining the estimated parameters, we use it to transform the second point cloud to the reference coordinate system defined by the first point cloud. Then we construct the final scene, and the final result is shown in Fig. 25.

4.7. Experimental results and discussion

4.7.1. Registration results

Figs. 11 and 12 represent the 3D registration results on Happy Buddha model and Skeleton Hand model with various perturbations through Bar graphs. It can be seen that the learning-based registration algorithms (DO, RDO) have better performance than the traditional registration methods (ICP, CPD, BCPD, NDT, IRLS) on all evaluation metrics, i.e., Successful Registration Rate, Average MSE, and Computation Time. Specifically, in the presence of various perturbations, the Successful Registration Rate of RDO is higher and more stable compared with other algorithms. And the

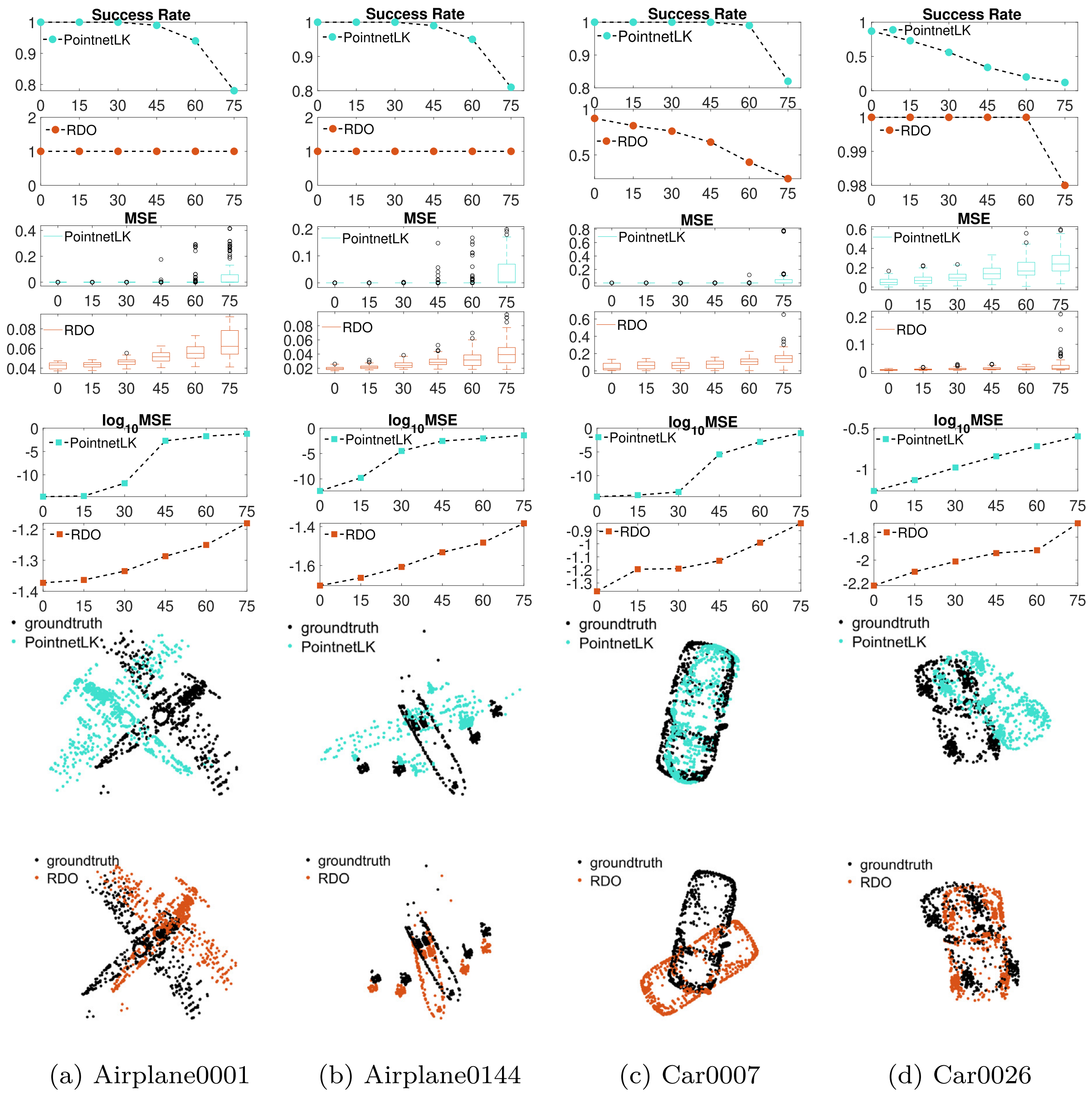


Fig. 18. The registration results of the multi-class training scheme with perturbation setting $mode_1$. (Top) shows the Success Rate. (Second) shows the Mean Square Error (MSE). (Third) shows the log_{10} average MSE. (Bottom) displays the registration results of ModelNet40 with 60° rotation, and the turquoise shows the registration result of PointnetLK, and the dark orange illustrates the registration results of RDO. It can be seen that RDO can keep higher stability than PointnetLK.

values of Average MSE illustrate that RDO can achieve more accurate and stable registration on different data sets (Happy Buddha and Skeleton Hand) than DO. While ICP required less computation time in all cases, it showed low success rates in high perturbations cases, and the performance of CPD was similar to that of ICP. As another statistic-based algorithm, NDT was more time-consuming and had higher registration errors and lower success rates than CPD. BCPD had better performance and took less time dealing with registration under all kinds of rotation and noise. However, when there were different degrees of outliers and occlusion, the performance of the algorithm was not ideal. IRLS required a high computation time and did not perform well when the model was highly incomplete. Fig. 13 illustrates that the better performance of RDO in dealing with the registration with 60° rotation compared with other algorithms. The rectangles illustrate the obvious difference between the registration results.

Figs. 14 and 15 show the statistical registration results on the Bimba model and Dancing children model with various perturbations by Box plots. The $\log_e$ value of the registration error corresponding to the inter-quartile ranges in Box Plots illustrates that the learning-based algorithms (DO and RDO) are more robust than the traditional algorithms (ICP, CPD, BCPD, NDT, IRLS). ICP, IRLS, and BCPD did not perform well when the model was highly incomplete. And CPD and BCPD were less able to deal with registration with outliers, especially when the number of outliers was high. DO and RDO outperformed the baselines in almost all test scenarios. However, RDO achieved more accurate registration than DO as shown in the positions of minimum, maximum, quartiles, and the skewness in Box plots.

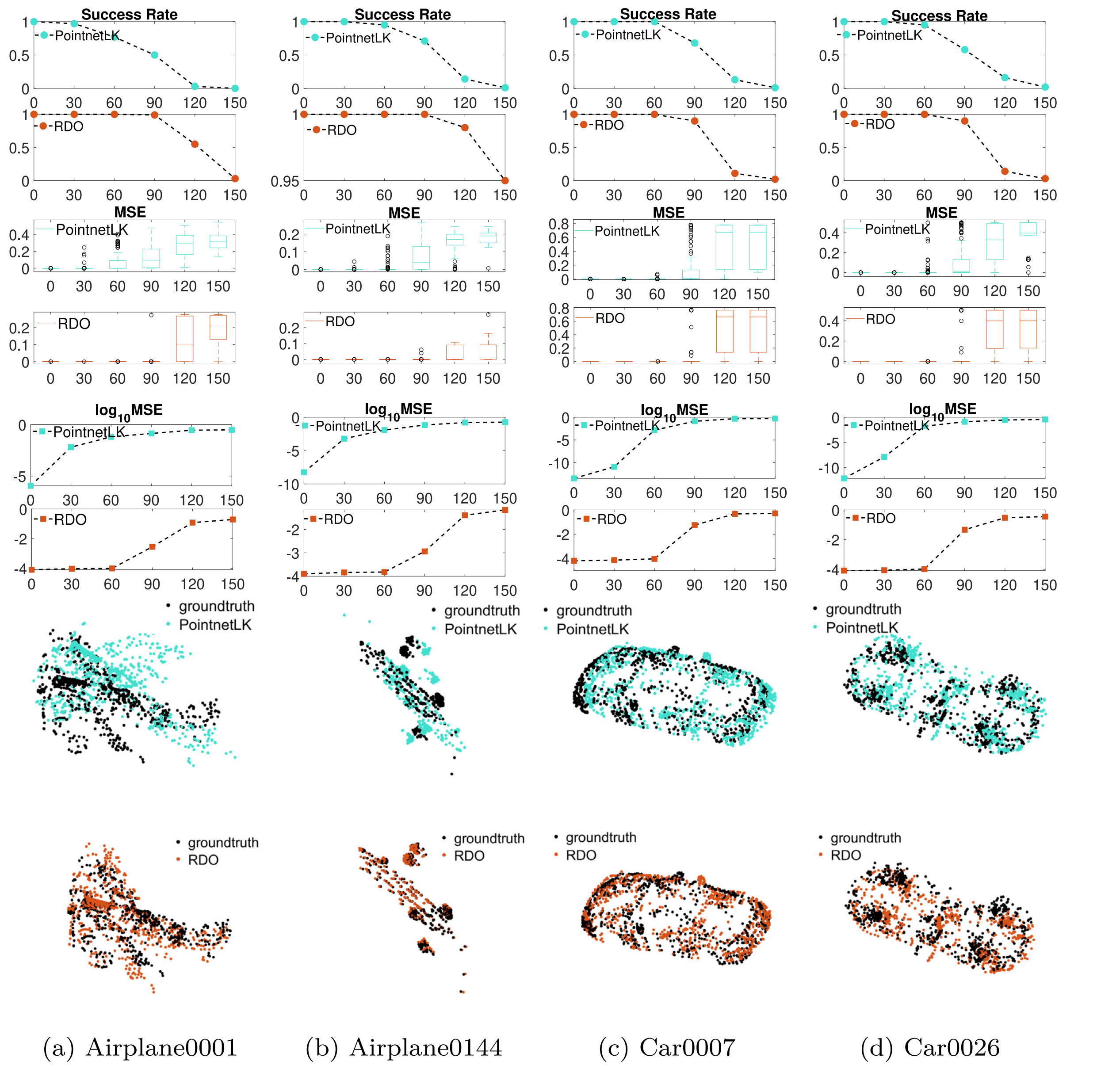
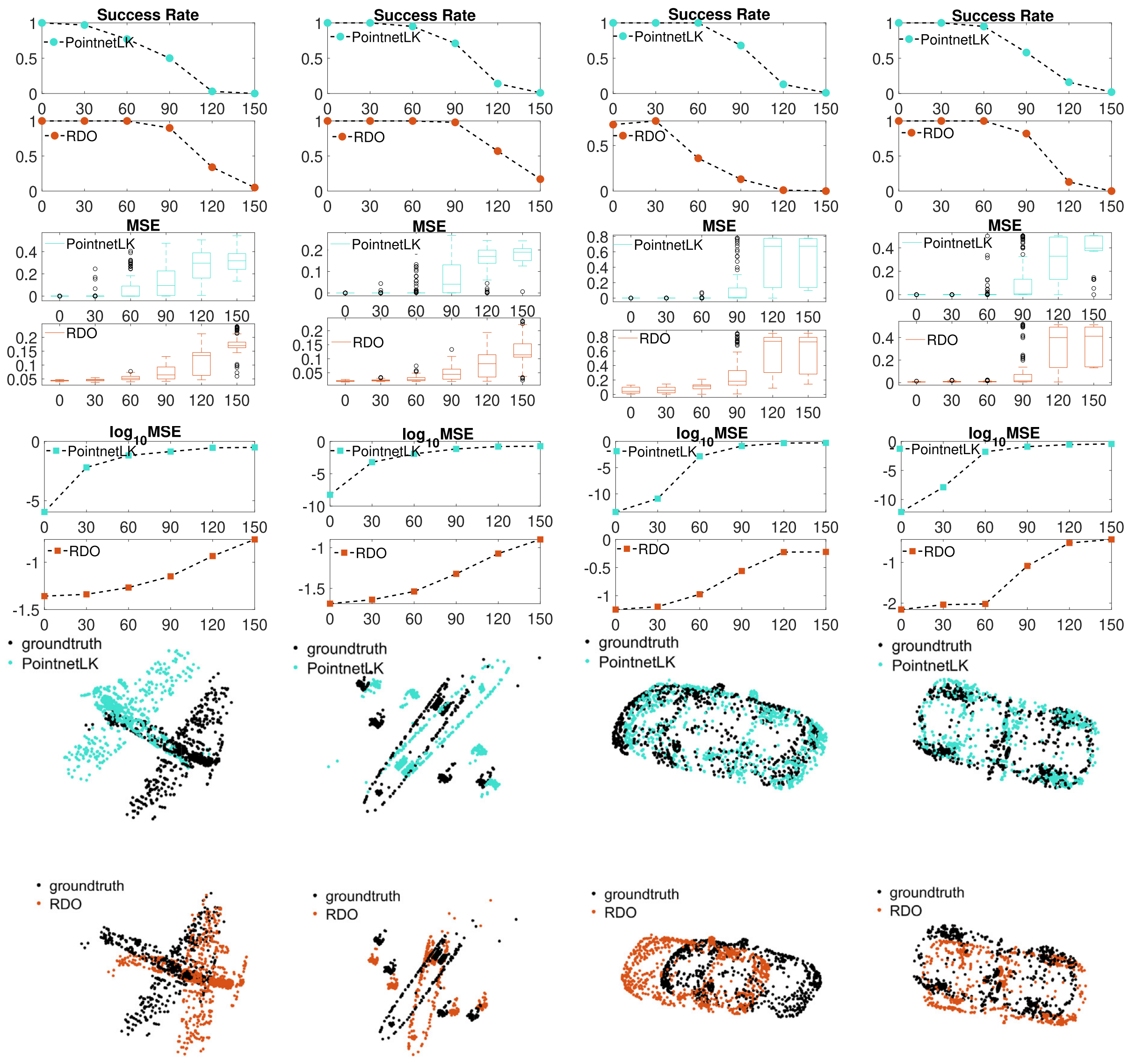


Fig. 19. The registration results of the single-class training scheme with perturbation setting $mode_2$. (Top) shows the Success Rate. (Second) shows the Mean Square Error (MSE). (Third) shows the log_{10} average MSE. (Bottom) displays the registration results of ModelNet40 with 120° rotation, and the turquoise shows the registration result of PointnetLK, and the dark orange illustrates the registration results of RDO. It can be seen that RDO has higher accuracy and robustness than PointnetLK when dealing with registration with large rotations.

Fig. 16 shows the results of the registration with the UWA data set. While DO and RDO have outperformed other traditional registration algorithms in terms of the Success Rate, Average MSE, and Computation Errors, RDO has shown improvements over DO and maintained low computation time. RDO produces more accurate results than DO when registering point clouds with large rotations.

Figs. 17, 19, 21 and 23 illustrate the registration results of RDO and PointnetLK on ModelNet40 with the single-class training scheme. The first two correspond to the evaluation for perturbation setting modes $mode_1$ and $mode_2$, and the last two show the registration results corresponding to the perturbation setting mode $mode_3$. Fig. 17 shows the registration results when rotation is within 45° during the training process. It can be seen that, compared with PointnetLK, RDO had a higher and more stable *Success Rate* and lower *Mean Square Error (MSE)*. The number of outliers in box-plots of *MSE* illustrates that the performance of PointnetLK is not stable when dealing with the registration with rotation degree over 45°, which also can be verified by the registration accuracy showed in *log_{10} average MSE*. Fig. 19 displays the registration results with the training rotation within 60°. It can be seen that PointnetLK did not perform well when registering the point cloud with a larger rotation, even it had low registration accuracy when dealing with registration with 60°. Figs. 21 and 23 illustrate the registration results when the rotation is within 45° and the standard deviation of Gaussian noise is 0.05 during the training process. Fig. 21 also exposes the low robustness and instability of the PointnetLK method when registering point clouds with rotation degrees over 45°. Fig. 23 shows that PointnetLK did not per-



(a) Airplane0001 (b) Airplane0144 (c) Car0007 (d) Car0026

Fig. 20. The registration results of the multi-class training scheme with perturbation setting $mode_2$. (Top) shows the Success Rate. (Second) shows the Mean Square Error (MSE). (Third) shows the log_{10} average MSE. (Bottom) displays the registration results of ModelNet40 with 120° rotation, and the turquoise shows the registration result of PointnetLK, and the dark orange illustrates the registration results of RDO. The accuracy of registration results of RDO is lower than that of PointnetLK.

form well when dealing with registration with various degrees of noise.

Figs. 18, 20, 22 and 24 illustrate the registration results of RDO and PointnetLK on ModelNet40 with the multi-class training scheme. The first two correspond to the evaluation for perturbation setting modes $mode_1$ and $mode_2$, and the last two show the registration results corresponding to the perturbation setting mode $mode_3$. Fig. 18 shows the registration results when rotation is within 45° during the training process. It can be seen that, compared with PointnetLK, the performance of RDO is stable but has lower accuracy, which also can be illustrated by Fig. 20. However, in the case of multi-class and multi-perturbation training, the stability of RDO and the accuracy of RDO are better, especially for registration when the perturbation scale exceeds the prescribed range setting in the training process, as shown in Figs. 22 and 24. Besides, Figs. 18 and 22 show that the stability of PointnetLK and the accuracy of PointnetLK are reduced in the case of multi-perturbations training. By contrast, the performance of RDO is more stable.

The comparative study demonstrates not only the ability of learning-based algorithms in dealing with complex point cloud registration tasks in the presence of noise and the incomplete data sets or occlusions but also shows the marked performance improvements of our proposed Reweighted Discriminative Optimization (RDO) over the learning-based algorithm Discriminative Optimization (DO) for registration tasks. And RDO maintains high accuracy, robustness, and stability when dealing with point cloud registration with large perturbations. Compared with the deep-learning-based method PointnetLK, RDO has better stability and robustness in dealing with registration problems with large and multiple perturbations.

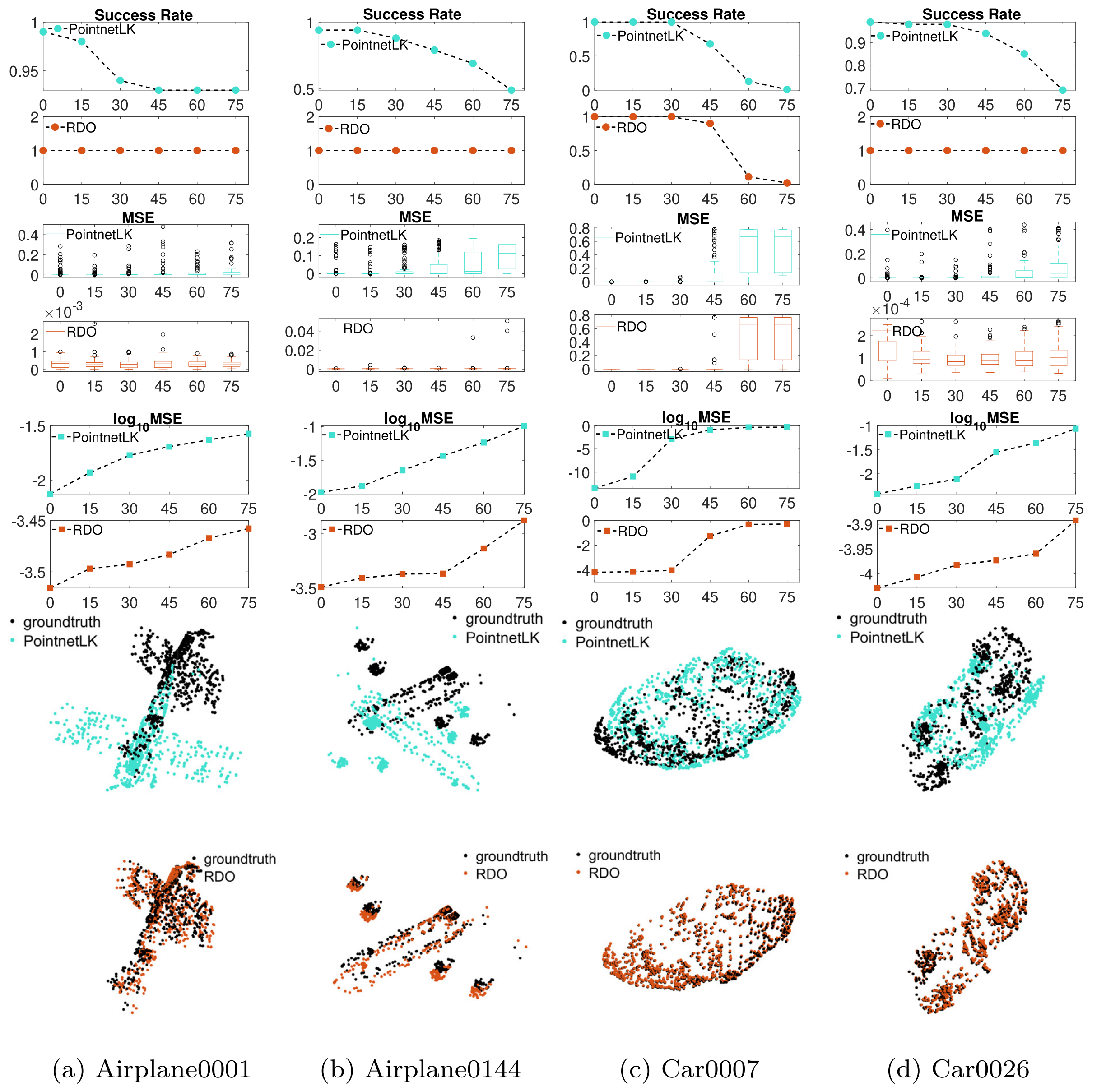


Fig. 21. The registration results of the single-class training scheme with perturbation setting *mode*$_3$. (Top) shows the Success Rate. (Second) shows the Mean Square Error (MSE). (Third) shows the log_{10} average MSE. (Bottom) displays the registration results of ModelNet40 with 75° rotation, and the turquoise shows the registration result of PointnetLK, and the dark orange illustrates the registration results of RDO. RDO keeps its higher stability and accuracy.

4.7.2. Stitching results

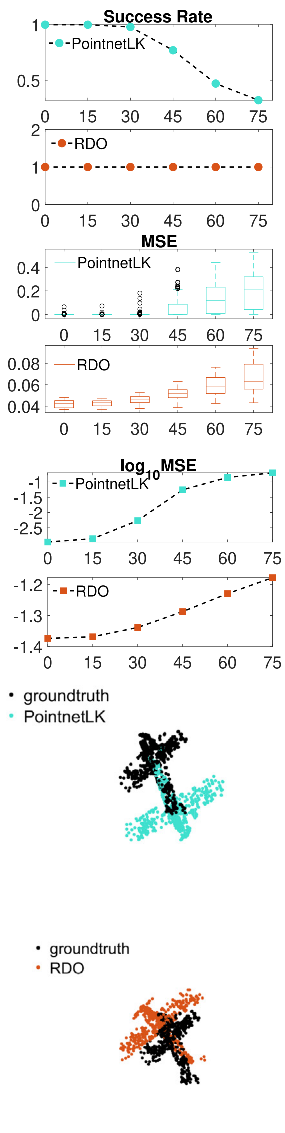
Fig. 25 shows the 3D stitching results of the two-view point clouds. The ellipses show the visible difference of reconstructed 3D scenes, and the obvious detailed information is exhibited in the rectangles. It can be seen that RDO and NDT performed well than other algorithms. RDO adjusts the degrees of rotation and translation by assigning weighting coefficients to transformation matrices, which causes the difference between the stitching results of DO and RDO.

4.7.3. Proof of convergence

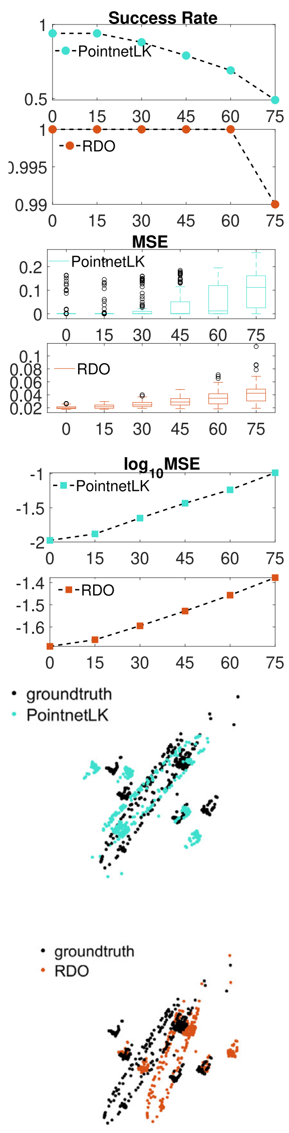
Fig. 26 shows the Convergence Criteria and Training Errors of RDO and DO on different data sets. We can find that our method meets the convergence condition $\sum_{i=1}^{N}(\mathbf{x}_*^i - \mathbf{x}_t^i)^T\hat{\mathbf{D}}\mathbf{h}_t^i > 0$ for all data sets (as shown in (a)), and the training error of RDO decreases in each iteration (as shown in (b)). (c) and (d) illustrate that DO and RDO converges sub-linearly and logarithmically. It can be seen that the addition of weighting coefficients does not have impact on the rate of convergence.

4.7.4. Space complexity analysis

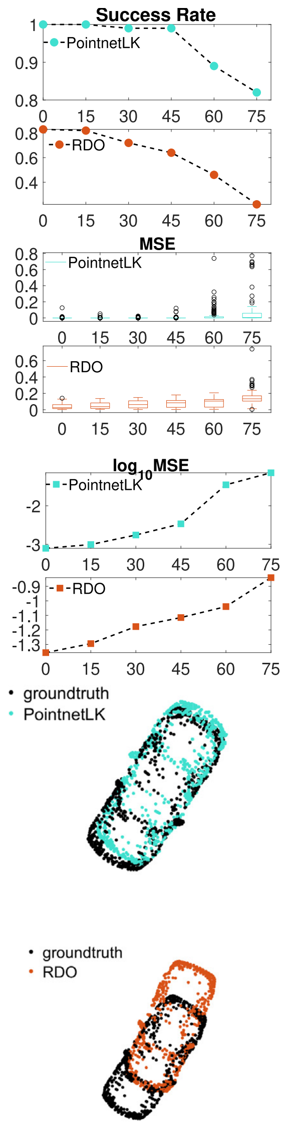
The memory requirement of RDO per iteration is determined by the variable storage. Calculating weighting matrix $\mathbf{W}_t$ needs maximum storage $O(c_1 \times N \times N_M)$; Calculating feature $\mathbf{h}$ needs maximum storage $O(c_2 \times N \times (N_M + N_S))$; Calculating the regressor $\mathbf{D}$ needs maximum storage $O(c_3 N_M \times N_M)$. Therefore, the memory requirement of RDO per iteration is $O(N(((c_1 + c_2)N_M + c_2 N_S))+ c_3 N_M^2)$, where N_M, N_S are the size of target model $\mathbf{M}$ and the size



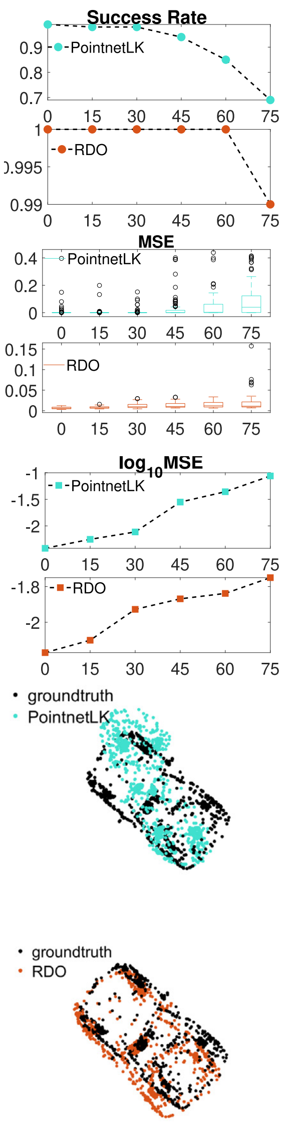
(a) Airplane0001



(b) Airplane0144



(c) Car0007



(d) Car0026

Fig. 22. The registration results of the multi-class training scheme with perturbation setting $mode_3$. (Top) shows the Success Rate. (Second) shows the Mean Square Error (MSE). (Third) shows the log_{10} average MSE. (Bottom) displays the registration results of ModelNet40 with 75° rotation, and the turquoise shows the registration result of PointnetLK, and the dark orange illustrates the registration results of RDO. The Success Rate shows higher stability of RDO, and the MSE illustrates the higher accuracy of RDO when dealing with registration with larger perturbations.

of moving model $\mathbf{S}, c_i, i = 1, 2, 3$ are constant, and N is the number of training samples.

5. Discussion for DO and RDO

DO, as a classical Supervised sequential update (SSU) algorithm, utilizes least-squares to solve parameter estimation in computer vision by learning update directions from training samples. DO is more robust to noise and outliers and other perturbations and efficient than other traditional methods. RDO is highly inspired by the DO algorithm. As an asymmetrical parameter treatment scheme, RDO aims to improve the robustness of parameter estimation in least-squares problems. RDO considers the different impact of each component of the parameter vector on the final fitting error, which is different from DO. Section 4 shows the comparison of the experimental results of RDO, DO, and other traditional registration methods and illustrates that RDO is more robust than DO.

This section discusses the substantial difference between DO and RDO, which focuses on the training stage. The computational efficiency of DO and RDO is different in the training stage and the same in the testing stage. The major difference in the training stage is the calculation of the weighting matrix. The computation of the weighting matrix is $O(6N)$, where N is the number of training samples. We compared the transformation status of DO and RDO at the end of the training process and when the iteration increases (as shown in Figs. 27 and 28). In addition, the registration performance under different perturbations is also discussed when the iteration number T increases (as shown in Fig. 29).

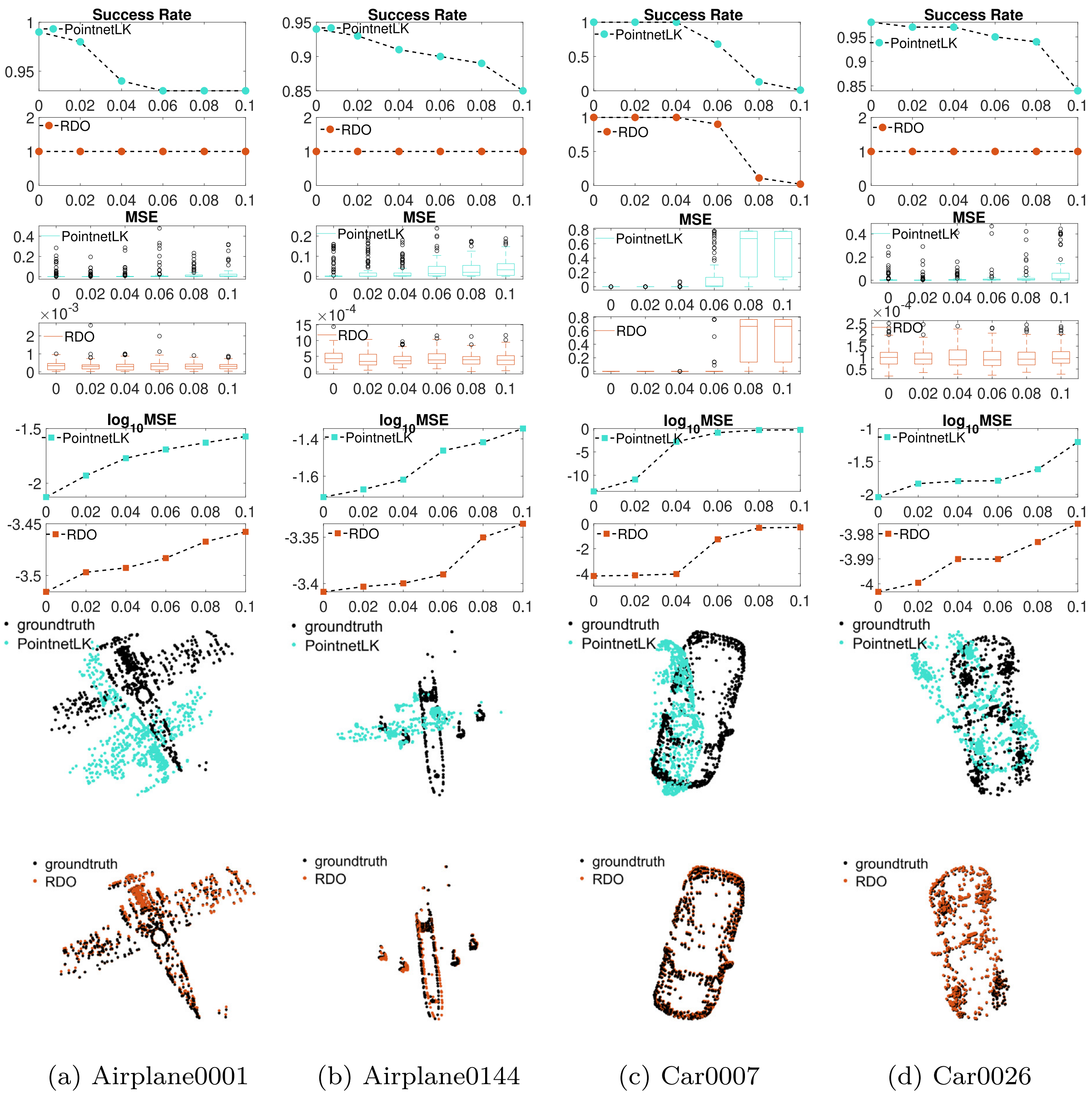


Fig. 23. The registration results of the single-class training scheme with perturbation setting $mode_3$. (Top) shows the Success Rate. (Second) shows the Mean Square Error (MSE). (Third) shows the log_{10} average MSE. (Bottom) displays the registration results with the perturbation of Gaussian noise with a standard deviation of 0.05, and the turquoise shows the registration result of PointnetLK, and the dark orange illustrates the registration results of RDO. The registration results show the higher robustness and accuracy of RDO.

Fig. 27 illustrates the transformation differences of DO, RDO, and *RDO without t* at the end of the training process. The *t* is from the Eqs. (12) and (13). The pink plane in the first row and third row represent the ground truth, where the lines drawn along rows and columns reflect the transformation difference. The lines in the second column correspond with the leftmost lines of each plane in the first column. The lines at the bottom are obtained by rotating the bottommost lines of the planes in the third column 90° clockwise. (a), (b) and (c) show the transformation of DO, RDO and *RDO without t* at the 30_{th} iteration respectively. (d) represents the transformation of *RDO without t* at the 29_{th} iteration.

Due to the higher robustness of rotation than translation shown in Fig. 1, the rotation plays a significant role in the transformation when handling perturbations. The rotation difference between ground truth and the transformation is reflected by the two adjacent edges of a plane. When the two adjacent edges of the plane are parallel or coincident with that of the ground truth plane (pink plane), the rotation angle is the same as that of the ground truth.

From (a) and (b), it can be seen that the rotation of RDO in the training process is closer to the rotation of ground truth, compared with that of DO. (c) and (d) illustrate that although allocating weights without *t* restriction can also guide the rotation to close to the ground truth (compared with DO), the weight without *t* will guide the plane to rotate by a large margin when the rotation has been close to the ground truth. (c) The shades in the first and third columns indicate that the two planes intersect, which means that after the 29_{th} iteration (d), the plane still rotates with a large degree.

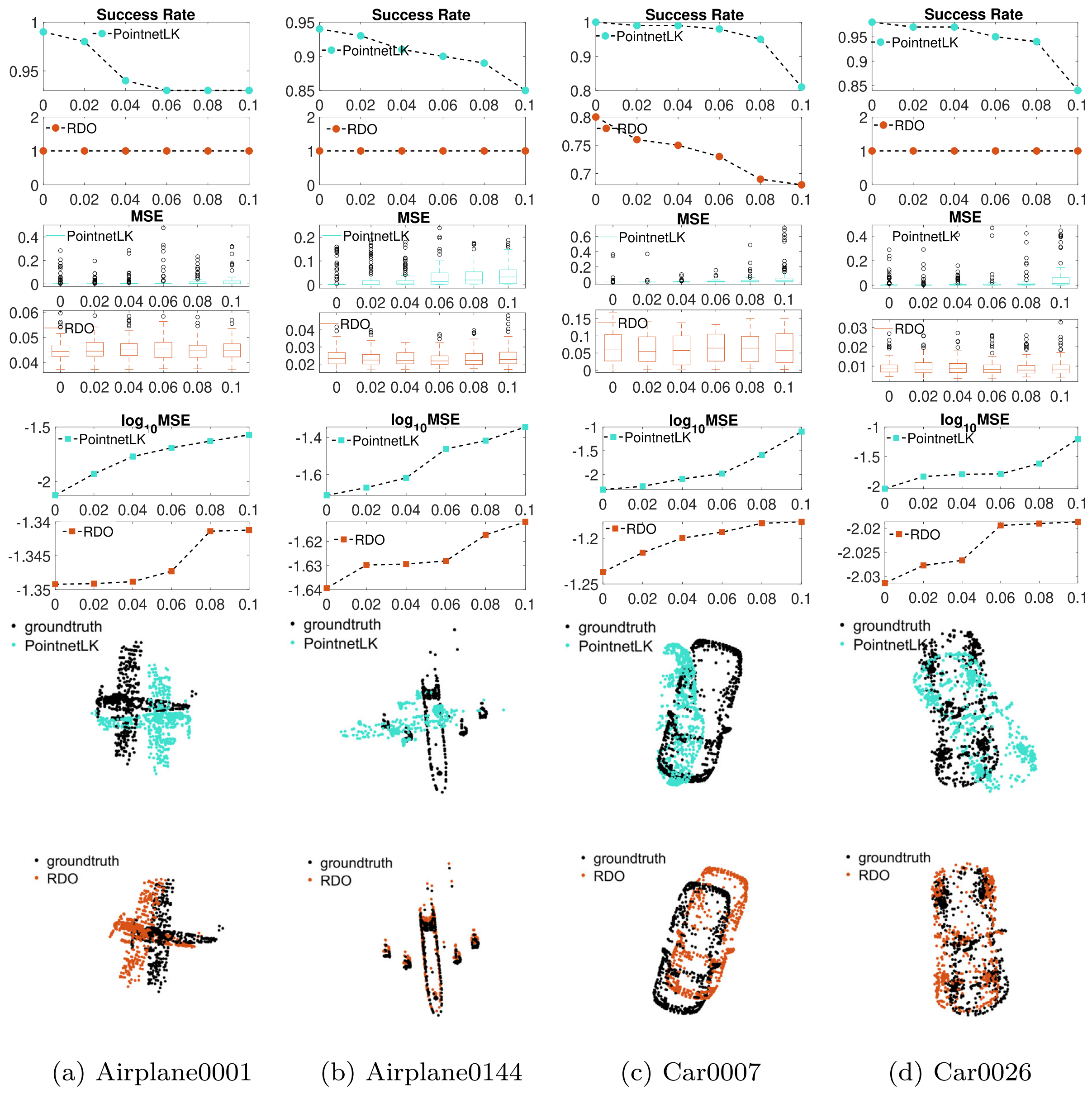


Fig. 24. The registration results of the multi-class training scheme with perturbation setting $mode_3$. (Top) shows the Success Rate. (Second) shows the Mean Square Error (MSE). (Third) shows the log_{10} average MSE. (Bottom) display the registration results with the perturbation of Gaussian noise with a standard deviation of 0.05, and the turquoise shows the registration result of PointnetLK, and the dark orange illustrates the registration results of RDO. It can be seen that RDO can keep its higher stability in the case of muti-class and multi-perturbations training.

Fig. 28 shows the transformation at different iterations in the training stage. The first row show the transformations of DO at the $30_{th}, 34_{th}, 37_{th}$ and 40_{th} respectively. The values above each sub-figure correspond to sequential rotations about the X, Y, and Z axes. The rotation of ground truth (pink plane) is expressed as a vector $[-0.0518\ -0.1513\ 0.1099]$. The second row displays the transformation of RDO. The shades on the figures show the intersection of the planes.

It can be seen that as the iteration number increases, the rotation of DO is far more different from that of the ground truth $[-0.0518\ -0.1513\ 0.1099]$. Compared with DO, RDO's transformation changed a little. And there is no intersection between the transformation plane and the ground truth plane (pink).

Fig. 29 shows the Mean Square Error of registration with Synthetic data under different perturbation when $T = 30$ and $T = 50$. The plus and circle represent the MSE when $T = 50$ and $T = 30$, respectively. It can be seen that in most cases, the MSE of RDO registration is less than that of DO registration when $T = 50$, except for the registration of the Bimba Model with different initial rotation and noises. Besides, we can find that if the MSE of RDO registration when $T = 30$ is higher than the MSE of RDO registration when $T = 50$, such as the MSE of Happy Buddha with different noises, the MSE of RDO registration is still the least. Likewise, if the MSE of RDO registration when $T = 30$ is lower than that of RDO registration when $T = 50$, the MSE of RDO registration is the smallest compared with the MSE of DO registration. In short, compared with DO, RDO estimates a transformation vector closer to the ground-truth, when DO and RDO registration errors are both smaller.

Besides, we also compared the decreasing rate of rotation matching error and translation matching error between our algorithm and DO algorithm in each iteration. The rotation matching error is calculated through fixing $\hat{\mathbf{x}}_k = 0, k \in \{1, 2, 3\}$ and the translation matching error is the mean square error of all samples when

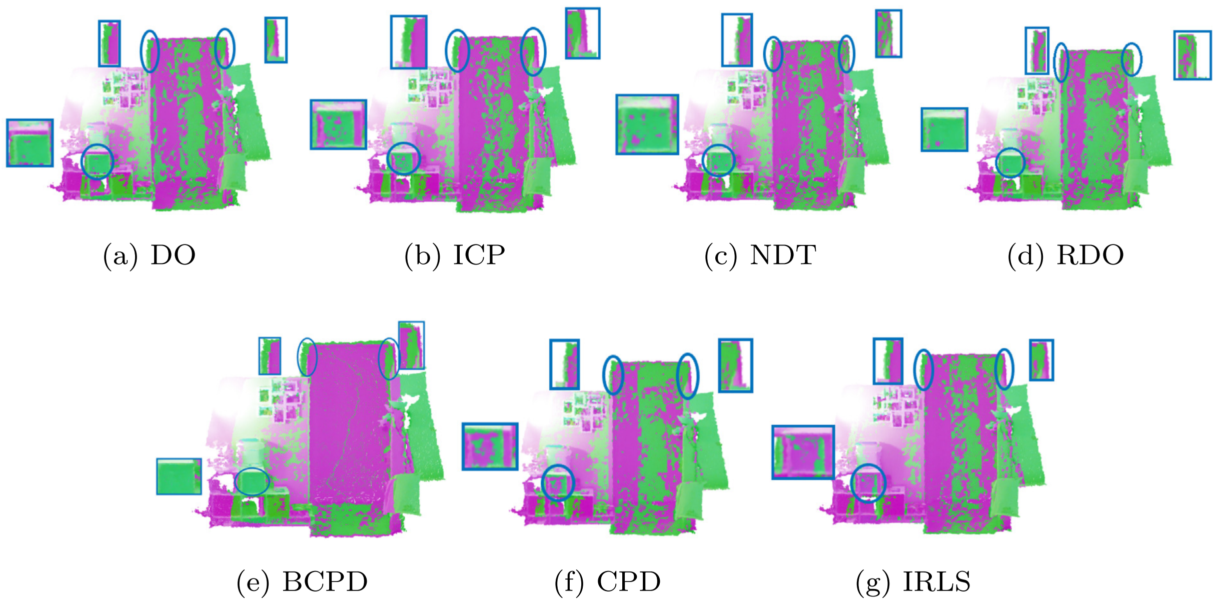


Fig. 25. Results of Stitching experiment on Matlab dataset. The ellipses show the visible difference of the constructed 3D scenes, and the obvious detailed information is exhibited in the rectangles.

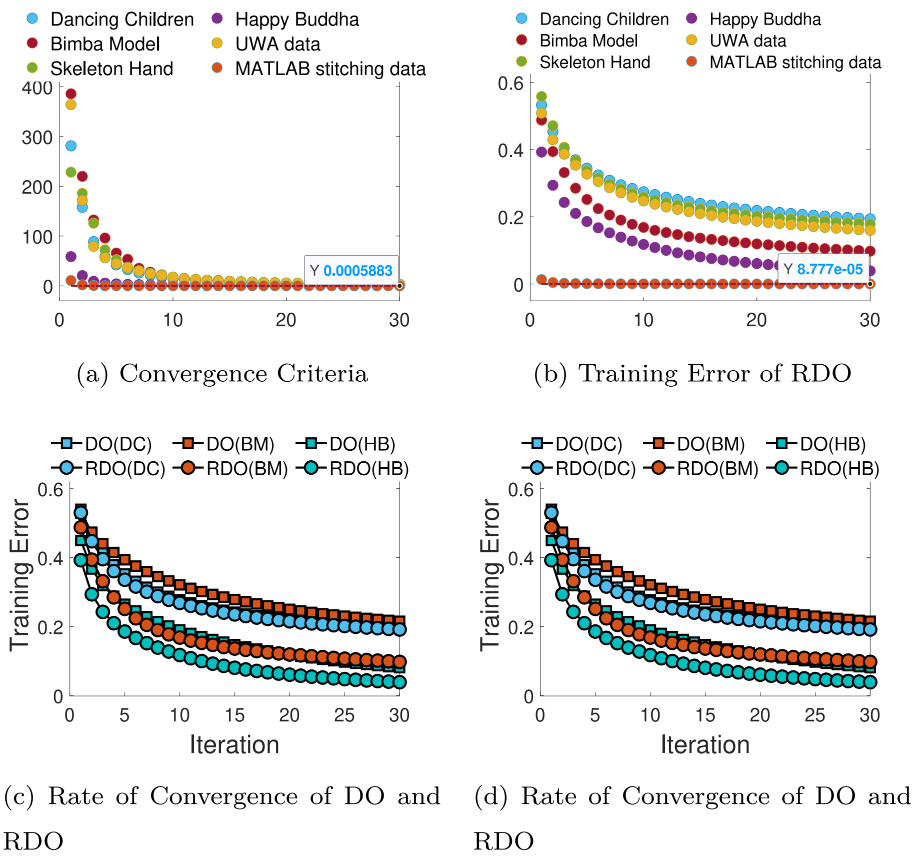
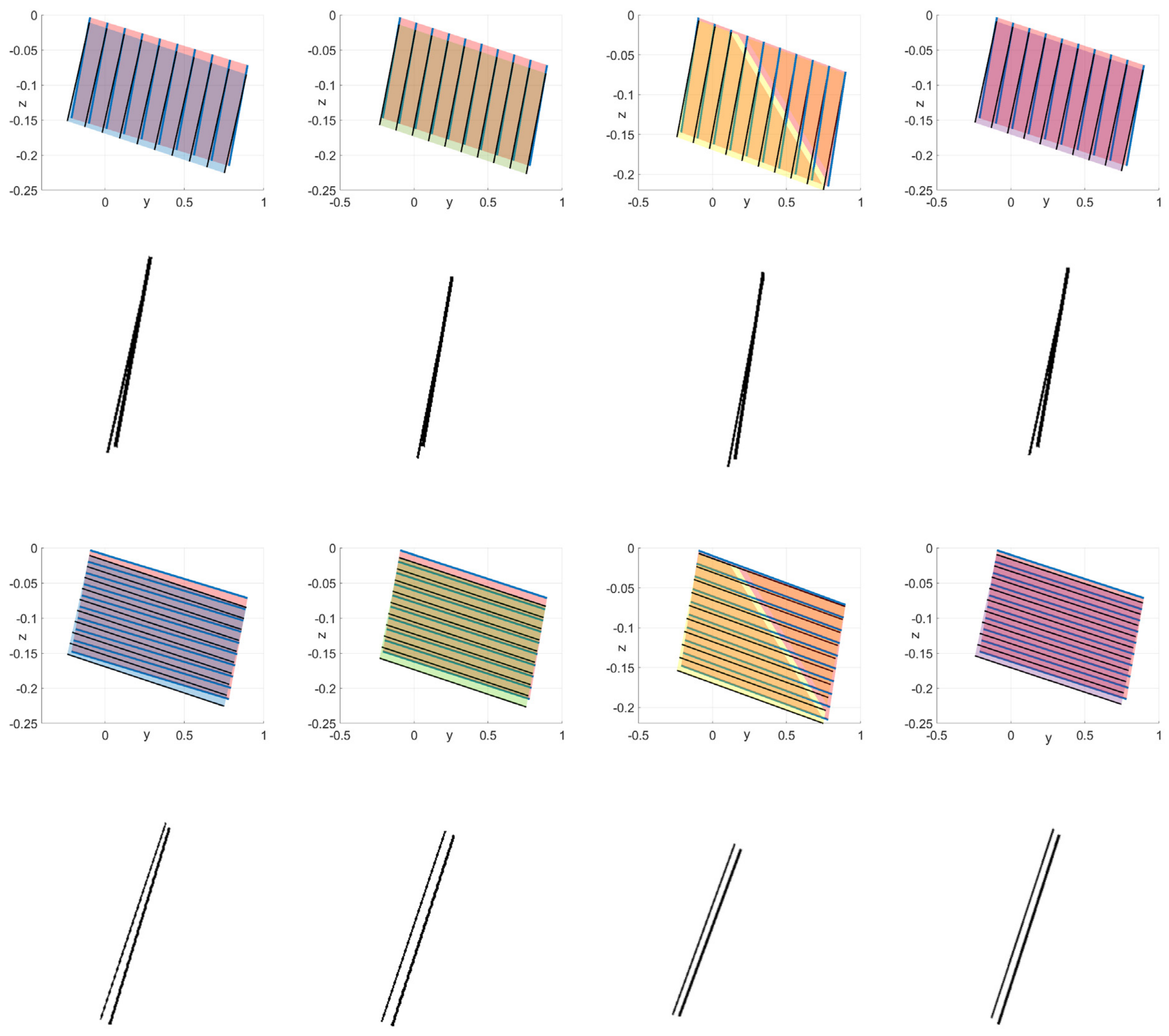


Fig. 26. The Convergence Criteria and Training Errors of RDO and DO on different data sets. (a) The value of $\sum_{i=1}^{N}(\mathbf{x}_*^i - \mathbf{x}_t^i)^\top \hat{\mathbf{D}}\mathbf{h}_t^i$ on different data sets. (b) The training error of our method on different data sets. (c) and (d) Rate of Convergence of DO and RDO on different data sets. The labels (DC, BM, etc.) are the abbreviations for the names of data sets; the rates of the convergence are marked by squares for DO and circles for RDO, respectively.



(a) DO Iteration=30 (b) RDO Iteration=30 (c) RDO without t Iteration=30 (d) RDO without t Iteration=29

Fig. 27. The transformation differences of DO, RDO, and *RDO without t*.

$\hat{\mathbf{x}}_k = 0, k \in \{4, 5, 6\}$. Fig. 30 shows the decreasing rate of matching error on rotation and translation. It can be seen that the weighting scheme in RDO accelerates the iterative process of convergence.

5.1. Summary

1) The asymmetrical parameter treatment scheme in RDO is able to adjust the scales of transformation in registration and make the rotation closer to the ground truth in the training process (as shown in (b), (c), (d) of Fig. 27) and the weighting matrix can accelerate the convergence process.

2) The t in the weights assignments Eqs. (12), . (13) controls the scale of transformation in each iteration (compared (b), (c), (d) in Fig. 27, as shown in Fig. 28). As the iteration number increases, DO's transformation with the uniform weighting scheme has been far more different from the ground truth in terms of rotation, and the difference is more and more obvious. In contrast, RDO's transformation has only changed a little bit at each iteration. Besides, the accumulation of the little change of RDO's transformation still has successfully avoided the larger difference from the ground truth.3) Increasing the number of iterations will influence the matching error (as shown in Fig. 29). In most cases, the benefit of RDO will remain when the iteration number T increases. Besides, whatever the iteration number, when DO and RDO registration errors are both smaller, we can find that the RDO always estimates the transformation vector closer to the ground truth.

6. Discussion for RDO and PointnetLK

RDO and PointnetLK are both learning-based methods, and they have both similarities and differences. The most obvious similarity of the RDO and PointnetLK is that they have a similar structure-multiple layers of regressors. *Training*: PointnetLK can be trained "end-to-end" through backpropagation; specifically, all layers can be affected by minimizing the loss function at once. RDO needs to be trained layer-by-layer, and the loss function in each layer is different, but the essence is in common- make the currently estimated parameters approach to the ground truth. *Flexibility*: Point-

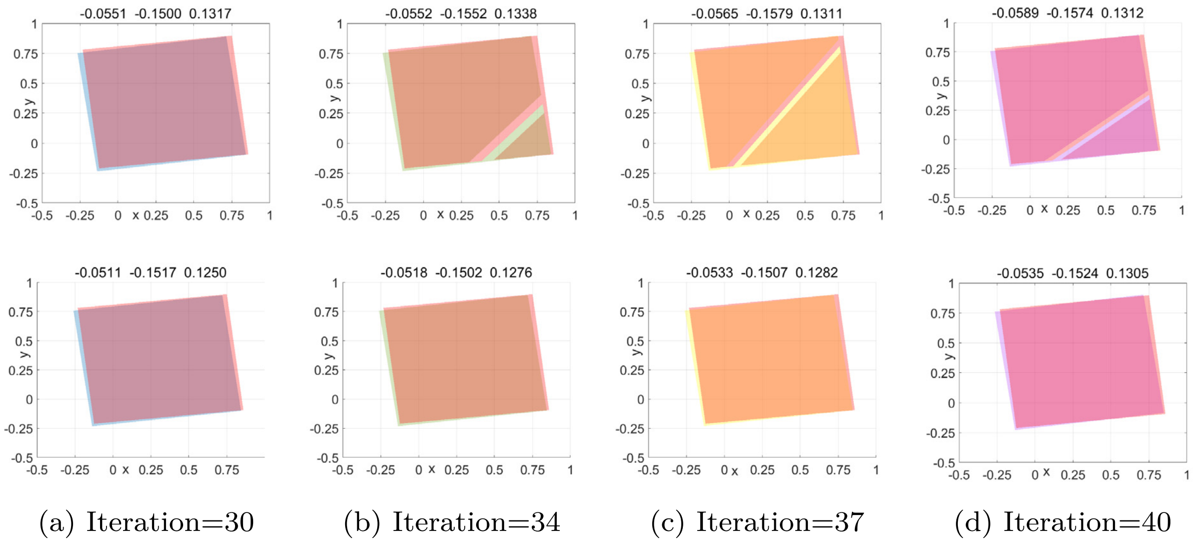


Fig. 28. The transformation with large *t*.

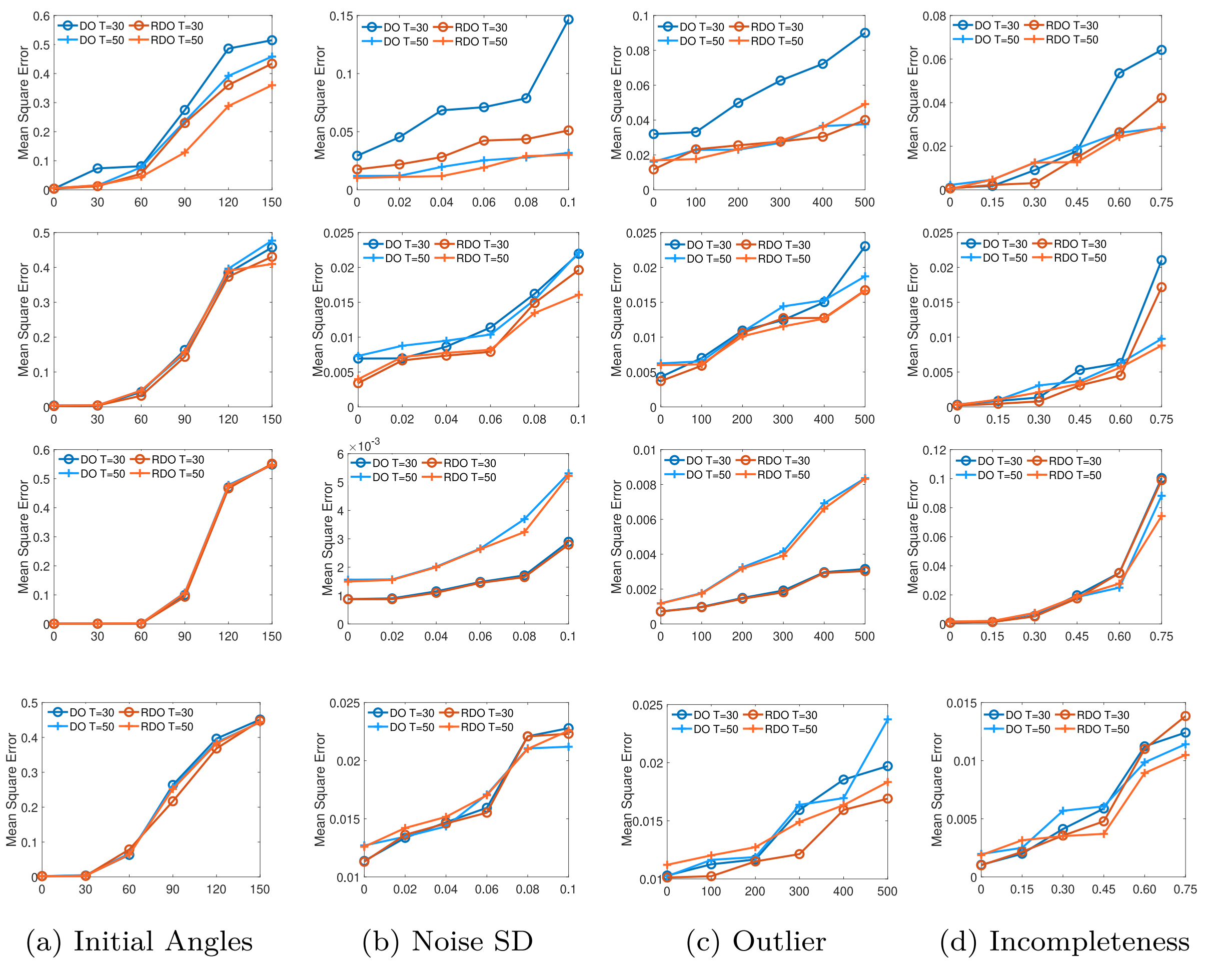


Fig. 29. The Mean Square Error of 3D registration with Synthetic data under different perturbations with different iteration numbers; (Top) Happy Buddha. (Second row) Skeleton Hand. (Third row) Bimba Model. (Bottom) Dancing Children.

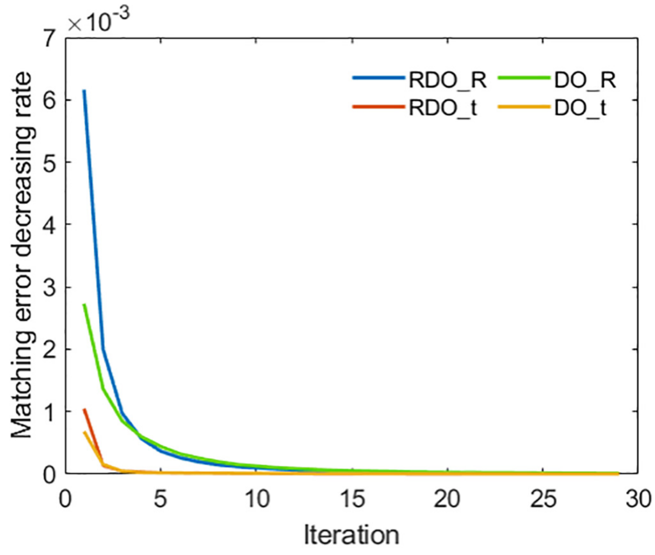


Fig. 30. The decreasing rate of rotation matching error and translation matching error.

netLK is more flexible. It can deal with different kinds of data sets (such as ModelNet40) at the same time. The memory requirement for learning $\mathbf{D}_{t+1}$ is $O\left(N(((c_1+c_2)N_M+c_2N_S))+c_3N_M^2\right)$, which does not apply to dealing with many data sets at once like deep learning. Namely, PointnetLK is almost purely data-driven, and RDO is more model-driven, where model means the feature. Although PointnetLK can process a variety of data simultaneously, its stability is far less than that of RDO. Even in the case of dealing with large perturbations, RDO can still maintain stability, and the registration error is lower than that of PoinnetLK. Besides, multi-perturbations training will affect the performance of PoinnetLK, but RDO will maintain stability and robustness under the same conditions.

7. Conclusion

This paper proposes a novel Reweighted Discriminative Optimization (RDO), an asymmetrical parameter treatment scheme to improve the accuracy of parameter estimation in least-squares problems. Specifically, RDO assigns different weights to components of parameter vectors according to the characteristics of the fitting errors over parameter vectors space to emphasize certain components of parameter vectors. We provide theoretical proof on the convergence of RDO under mild conditions. We demonstrate the potential of RDO in computer graphics and visualization applications through the problems of 3D point cloud registration and multi-view stitching. Our comparative study with state-of-the-art algorithms illustrates that RDO produces more accurate and stable results. Future work is to design a generalized representation of parameter vectors that is suitable for computer vision and graphics applications other than those specific to the Lie Algebra for a rigid transformation matrix. On this basis, we believe RDO can be applied to a much wider range of problems in computer graphics and computer vision, such as non-rigid registration, image denoising, and so on.

CRediT authorship contribution statement

Yan Zhao: Conceptualization. **Wen Tang:** Methodology. **Jun Feng:** Formal analysis. **TaoRuan Wan:** Supervision. **Long Xi:** Software.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

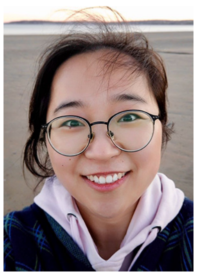
Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.neucom.2021.08.080.

References

[1] Y. Xiao, B. Jiang, A. Zheng, A. Zhou, A. Hussain, J. Tang, Saliency detection via multi-view graph based saliency optimization, Neurocomputing 351 (2019) 156–166.
[2] J. Su, B. Chen, H. Qiao, Z.-Y. Liu, Caging a novel object using multi-task learning method, Neurocomputing 351 (2019) 146–155.
[3] X. Yan, Y. Zhang, D. Zhang, N. Hou, Multimodal image registration using histogram of oriented gradient distance and data-driven grey wolf optimizer, Neurocomputing. .
[4] B. Goldlücke, M. Aubry, K. Kolev, D. Cremers, A super-resolution framework for high-accuracy multiview reconstruction, International Journal of Computer Vision 106 (2) (2014) 172–191.
[5] L. Lin, B. Liu, Y. Xiao, Cob method with online learning for object tracking, Neurocomputing 393 (2020) 142–155.
[6] Y. Yang, Y. Su, D. Cai, M. Xu, Nonlinear deformation learning for face alignment across expression and pose, Neurocomputing 195 (2016) 149–158.
[7] M. Aigner, B. Juttler, Gauss-newton-type techniques for robustly fitting implicitly defined curves and surfaces to unorganized data points, in: 2008 IEEE International Conference on Shape Modeling and Applications, IEEE, 2008, pp. 121–130.
[8] A.M. Buchanan, A.W. Fitzgibbon, Damped newton algorithms for matrix factorization with missing data, 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 2, IEEE, 2005, pp. 316–322.
[9] R.R. Paulsen, J.A. Baerentzen, R. Larsen, Markov random field surface reconstruction, IEEE Transactions on Visualization and Computer Graphics 16 (4) (2009) 636–646.
[10] A. Fischer, A special newton-type optimization method, Optimization 24 (3–4) (1992) 269–284.
[11] H. Qinghui, W. Shiwei, L. Zhiyuan, L. Xiaogang, Quasi-newton method for lp multiple kernel learning, Neurocomputing 194 (2016) 218–226.
[12] M. Jing, X. Zhou, C. Qi, Quasi-newton iterative projection algorithm for sparse recovery, Neurocomputing 144 (2014) 169–173.
[13] X. Xiong, F. De la Torre, Supervised descent method and its applications to face alignment, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 532–539.
[14] X. Xiong, F. De la Torre, Global supervised descent method, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 2664–2673.
[15] J. Vongkulbhisal, F. De la Torre, J.P. Costeira, Discriminative optimization: Theory and applications to computer vision, IEEE Transactions on Pattern Analysis and Machine Intelligence 41 (4) (2019) 829–843.
[16] C.S. Burrus, Iterative reweighted least squares, OpenStax CNX. Available online: http://cnx. org/contents/92b90377-2b34-49e4-b26f-7fe572db78a1 12. .
[17] P. Bergström, O. Edlund, Robust registration of point sets using iteratively reweighted least squares, Computational Optimization and Applications 58 (3) (2014) 543–561.
[18] P.J. Besl, N.D. McKay, Method for registration of 3-d shapes, in: Sensor Fusion IV: Control Paradigms and Data Structures, vol. 1611, International Society for Optics and Photonics, 1992, pp. 586–606. .
[19] L. Zhu, J. Barhak, V. Srivatsan, R. Katz, Efficient registration for precision inspection of free-form surfaces, The International Journal of Advanced Manufacturing Technology 32 (5–6) (2007) 505–515.
[20] A. Segal, D. Haehnel, S. Thrun, Generalized-icp, in: Proceedings of Robotics: Science and Systems, Seattle, USA, 2009. doi:10.15607/RSS.2009.V.021..
[21] R.A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A.J. Davison, P. Kohi, J. Shotton, S. Hodges, A. Fitzgibbon, Kinectfusion: Real-time dense surface mapping and tracking, in: 2011 10th IEEE International Symposium on Mixed and Augmented Reality, IEEE, 2011, pp. 127–136. .
[22] F. Pfeuffer, M. Stiglmayr, K. Klamroth, Discrete and geometric branch and bound algorithms for medical image registration, Annals of Operations Research 196 (1) (2012) 737–765.

[23] P. Biber, W. Straßer, The normal distributions transform: A new approach to laser scan matching, in: Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No. 03CH37453), vol. 3, IEEE, 2003, pp. 2743–2748. .
[24] A. Myronenko, X. Song, Point set registration: Coherent point drift, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (12) (2010) 2262–2275.
[25] O. Hirose, A bayesian formulation of coherent point drift, IEEE Transactions on Pattern Analysis and Machine Intelligence (2020) 1–1 doi:10.1109/TPAMI.2020.2971687. .
[26] S. Ruder, An overview of gradient descent optimization algorithms, arXiv preprint arXiv:1609.04747. .
[27] T. Zhao, S. Li, K.N. Ngan, F. Wu, 3-d reconstruction of human body shape from a single commodity depth camera, IEEE Transactions on Multimedia 21 (1) (2019) 114–123.
[28] I. Khan, Robust sparse and dense nonrigid structure from motion, IEEE Transactions on Multimedia 20 (4) (2018) 841–850.
[29] A. Griewank, A. Walther, Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, vol. 105, Siam, 2008. .
[30] P. Dollár, P. Welinder, P. Perona, Cascaded pose regression, in: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE, 2010, pp. 1078–1085.
[31] G. Tzimiropoulos, Project-out cascaded regression with an application to face alignment, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3659–3667.
[32] X. Cao, Y. Wei, F. Wen, J. Sun, Face alignment by explicit shape regression, International Journal of Computer Vision 107 (2) (2014) 177–190.
[33] O. Tuzel, F. Porikli, P. Meer, Learning on lie groups for invariant detection and tracking, in: 2008 IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.
[34] J. Vongkulbhisal, F. De la Torre, J.P. Costeira, Discriminative optimization: Theory and applications to point cloud registration, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4104–4112.
[35] E. Eade, Lie groups for 2d and 3d transformations, http://ethaneade. com/lie. pdf, revised Dec. .
[36] M. Levoy, J. Gerth, B. Curless, K. Pull, The stanford 3d scanning repository, http://www-graphics. stanford. edu/data/3dscanrep 5. .
[37] G. Turk, B. Mullins, Large geometric models archive, georgia institute of technology (1998). .
[38] A.S. Mian, M. Bennamoun, R. Owens, Three-dimensional model-based object recognition and segmentation in cluttered scenes, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (10) (2006) 1584–1601.
[39] Y. Aoki, H. Goforth, R.A. Srivatsan, S. Lucey, Pointnetlk: Robust & efficient point cloud registration using pointnet, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 7163–7172.
[40] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, J. Xiao, 3d shapenets: A deep representation for volumetric shapes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1912–1920.



Yan Zhao received the B.A. degree in 2016 from School of Mathematics, Northwest University, Xi'an, China. She is a PhD candidate at Bournemouth University and Northwest University (China). Her research interests include optimization, computer graphics and augmented reality.

Wen Tang received her B.A. degree in 1984 from Xi'an Polytechnic University, Xi'an, China, and PhD degree from University of Leeds, UK, in 1998. She is currently a full professor at Bournemouth University, UK, and a guest professor at Xi'an Polytechnic University, China. Her research interests include interactive virtual reality software technologies, physically-based simulation, computer animation algorithms and computer games technology.

Jun Feng received her M.S. degree in 1997 from Northwest University, Xi'an, China, and PhD degree from City University of Hong Kong in 2006. She is a full professor at Northwest University, China. Her research interests include image processing, 3D reconstruction, artificial intelligence and pattern recognition.

TaoRuan Wan is a senior lecturer in the School of Informatics at the University of Bradford, Director of Post Graduate Research EIMC, and a guest professor at Xi'an Polytechnic University, China. His research interests are focused in the areas of Digital Health, VR/AR technologies, Computer Vision, AI and deep learning, Computational Models for visualization, Real-time Modeling and Simulation.

Long Xi received his M.S. degree in 2019 from the department of computer science, Xi'an Polytechnic University, Xi'an, China. He is currently a PhD candidate at Bournemouth University, UK. His research interests include computer vision, artificial intelligence and pattern recognition.