Fast and Robust Iterative Closet Point

Juyong Zhang, *Member, IEEE,* Yuxin Yao, Bailin Deng†, *Member, IEEE*

Abstract—The Iterative Closest Point (ICP) algorithm and its variants are a fundamental technique for rigid registration between two point sets, with wide applications in different areas from robotics to 3D reconstruction. The main drawbacks for ICP are its slow convergence as well as its sensitivity to outliers, missing data, and partial overlaps. Recent work such as Sparse ICP achieves robustness via sparsity optimization at the cost of computational speed. In this paper, we propose a new method for robust registration with fast convergence. First, we show that the classical point-to-point ICP can be treated as a majorization-minimization (MM) algorithm, and propose an Anderson acceleration approach to improve its convergence. In addition, we introduce a robust error metric based on the Welsch's function, which is minimized efficiently using the MM algorithm with Anderson acceleration. On challenging datasets with noises and partial overlaps, we achieve similar or better accuracy than Sparse ICP while being at least an order of magnitude faster. Finally, we extend the robust formulation to point-to-plane ICP, and solve the resulting problem using a similar Anderson-accelerated MM strategy. Our robust ICP methods improve the registration accuracy on benchmark datasets while being competitive in computational time.

Index Terms—Rigid Registration, Robust Estimator, Fixed-point iterations, Majorlazer Minimization method, Anderson Acceleration.

1 Introduction

Rigid registration, which finds an optimal registration to align a source point set with a target point set, is a fundamental problem in many areas including digital geometry processing, computer vision, robotics and medical image processing. The iterative Closest Point (ICP) algorithm [1] is a classical method for rigid registration. It alternates between closest point query in the target set and minimization of distance between corresponding points, and is guaranteed to converge to a locally optimal alignment.

In practice, however, ICP can suffer from slow convergence. It has been shown that classical ICP converges linearly only [2]. Other registration methods have been developed with faster convergence. For example, in [3] the alignment is performed by minimizing a point-to-plane distance instead of the point-to-point distance, whereas in [4] a locally quadratic approximant of the squared distance function is iteratively minimized. Both approaches are shown to have higher convergence rate than classical ICP [2].

Another issue with ICP is that the alignment accuracy can be affected by imperfections of the point sets such as noises, outliers and partial overlaps. Such imperfections often occur in real-world acquisition processes. Various techniques have been developed to address this problem. One popular approach is to disregard erroneous correspondence between points, using heuristics based on their distance or the angle between their normals [5]. Recently, an ℓ_p-norm minimization approach is proposed in [6] to induce sparsity of the distance between corresponding point pairs, which aligns the points in true correspondence while allowing large distance between points due to outliers and incomplete data.

In this paper, we propose a novel and simple approach to address these two issues. Our key observation is that classical ICP is a majorization-minimization (MM) algorithm [7] for minimizing ℓ_2 distance between the two point sets, which iteratively constructs and minimizes a surrogate function and ensures monotonic decrease of the target energy. By treating this process as a fixed-point iteration, we speed up its convergence using *Anderson acceleration* (AA) [8], an established numerical technique that proves effective for a variety of optimization problems in computer graphics [9]. In each iteration, Anderson acceleration computes an accelerated iterate based on the history of m previous iterates. Compared with existing approaches such as [3], [4], our method does not require higher-order information such as normal or curvature which may not be available from the point cloud data and need to be estimated carefully in the presence of noise [10]. Moreover, different from previous attempt on Anderson acceleration of ICP [11] that uses Euler angles to represent rotation, we adopt a parameterization of rigid transformation that does not suffer from singularity of Euler angles. Using the same MM framework, we can replace the squared distance metric used in classical ICP with a robust metric that is insensitive to noises, outliers, and partial overlaps. In particular, we adopt a robust metric based on the Welsch's function, which allows for a simple quadratic surrogate function and can be minimized efficiently. Compared to the sparse ICP algorithm [6], our approach does not require introducing auxiliary variables for the solver, which leads to lower memory footprint and significantly faster convergence. We evaluate the performance of the proposed algorithms on a variety of examples including synthetic data and real scanned data. Our experiments demonstrate that it significantly reduces the computational time and improves the robustness for alignment.

Our approach of minimizing a Welsch's function-based robust error metric with an Anderson-accelerated MM solver can be extended to other ICP formulations. In particular, we apply it to the point-to-plane ICP from [3], and achieve better registration accuracy than the original method on benchmark datasets. This illustrates the effectiveness of our method in improving robustness of ICP-type registration algorithms.

To summarize, our main contributions include:

- *J. Zhang and Y. Yao are with School of Mathematical Sciences, University of Science and Technology of China.*
- *B. Deng is with School of Computer Science and Informatics, Cardiff University.*

†*Corresponding author. Email:* DengB3@cardiff.ac.uk.

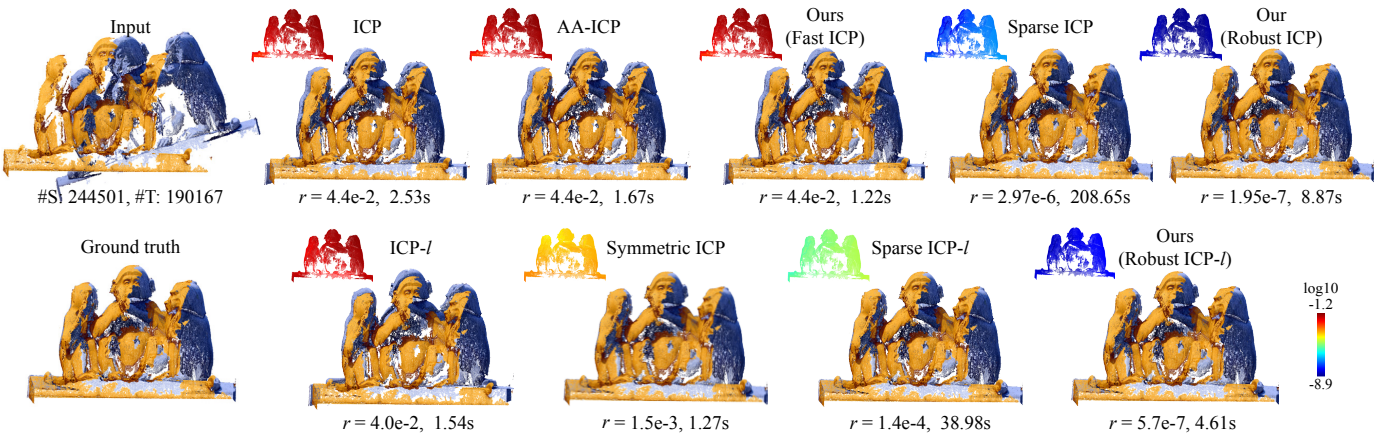

Fig. 1. Comparison between different registration methods on a pair of partially overlapping point clouds constructed using the monkey model from the EPFL statue dataset. #S and #T denote the number of points in the source and the target point clouds, respectively. The numbers at the bottom show the RMSE according to Eq. (14) and the computational time. The log-scale color-coding illustrates the deviation between the transformed source point clouds using the computed alignment and the ground-truth alignment. Our robust point-to-point and point-to-plane ICP methods result in the lowest RMSE values, while being an order of magnitude faster than Sparse ICP.

- We propose a new formulation for Anderson-accelerated point-to-point ICP method. We parameterize rigid transformations via Lie algebra instead of Euler angles as in [11], and use a more simple stabilization strategy than [11], resulting in faster convergence than the accelerated method in [11].
- We propose a robust metric for point-to-point alignment based on the Welsch's function, which is insensitive to outliers and partial overlaps and can be solved efficiently using the MM framework with Anderson acceleration. Our method achieves better registration accuracy than sparse ICP while being significantly faster.
- We extend the formulation to point-to-plane ICP, using the Welsch's function to define a robust error metric that is minimized with an Anderson-accelerated MM solver. Our formulation improves the robustness of point-to-plane ICP without the need for point pair rejection.

2 Related Work

Iterative Closest Point. Registration is a classical research topic in computer vision and robotics due to its numerous practical applications such as 3D scene reconstruction and localization. For a comprehensive review of rigid and nonrigd registration, the reader is referred to [12], [13]. Here, we focus on ICP for rigid registration. ICP and its variants [1], [3], [5], [14], [15] start from an initial alignment, and alternate between the update of correspondence using closest-points lookup and update of alignment based on the correspondence. Using this framework, an accurate registration relies on a good initial alignment as well as a robust way to update the alignment from correspondence.

For the initial alignment, Gelfand et al. [16] computed shape descriptors on the given point clouds, and determined a coarse alignment by matching a features points based on their descriptors. Rusu et al. [17] performed similar matching using the Point Feature Histograms defined at each point. Aiger et al. [18] aligned two point clouds by matching a pair of co-planar 4-point sets from them that are approximately congruent. Later, Mellado et al. [19] proposed a more efficient approach for such alignment that runs in the number of data points instead of the quadratic time required by [18].

To update the alignment, ICP minimizes the distance from the source points to their corresponding points [1] or to the tangent planes at the corresponding points [3]. Mitra et al. [20] proposed a framework that determines the alignment by minimizing a squared distance function between the two point clouds, which includes the point-to-point and point-to-plane distances used in classical ICP as special cases. They also proposed a local quadratic approximant to the squared distance function for efficient update of the alignment. A similar approach was taken in [4] for aligning a point cloud to a surface. Pottmann et al. [2] analyzed the convergence rates of different registration algorithms and showed that using a local quadratic approximant can lead to quadratic convergence. Recently, Rusinkiewicz [15] proposed a symmetrized objective function for ICP that yields faster convergence than point-to-point and point-to-plane objectives. Besides the convergence rate, another consideration for registration algorithms is their robustness to noises, outliers, and partial overlaps. A popular approach is to discard some point pairs from the alignment problem based on certain criteria about their distance [5], [21], [22]. Other methods take a statistical approach and align two point sets via their representations as Gaussian mixture models [23], [24]. Another approach is to optimize a robust objective that automatically reduces the influence from point pairs that are far apart [6], [25], [26], [27], [28]. In [6], the objective is defined as the ℓ_p-norm ($p < 1$) of the point-wise distances, which aligns nearby points while allowing for large deviation between point pairs due to outliers and partial overlaps. In this paper, we also optimize the sparsity of the point-wise distance, but with a more efficient solver with a stronger guarantee of convergence.

ICP is a local refinement process and relies on good initialization. Other methods formulate registration as a global optimization problem and solve it using branch-and-bound algorithms [29], [30], [31], which produces globally optimal results at the expense of higher computational costs.

Anderson Acceleration. Originally proposed in [32] for iterative solution of nonlinear integral equations, Anderson acceleration has proved effective for accelerating fixed-point iterations [8], [33], [34], [35]. It has been applied to accelerate various numerical solvers [36], [37], [38], [39], [40]. In the field of visual computing, Anderson acceleration has been applied recently to accelerate local-global solvers [9] and ADMM solvers [41]. One common issue for Anderson acceleration is that it can become unstable or stagnate [8], [42]. Peng et al. [9] proposed a simple strategy to improve its stability on optimization solvers by checking the decrease of the target function. Recently, Anderson acceleration has been used in [11] to accelerate the convergence of ICP. In this paper, we also apply Anderson acceleration to ICP, but using a different set of variables to represent the iteration, together with the simple stabilization strategy from [9].

3 Classical ICP Revisited

Given two sets of points $P = \{\mathbf{p}_1, \ldots, \mathbf{p}_M\}$ and $Q = \{\mathbf{q}_1, \ldots, \mathbf{q}_N\}$ in $\mathbb{R}^d$, we search for a rigid transformation on P, represented using a rotation matrix $\mathbf{R} \in \mathbb{R}^{d\times d}$ and a translation vector $\mathbf{t} \in \mathbb{R}^d$, to align P to Q. This is formulated as an optimization problem for $\mathbf{R}$ and $\mathbf{t}$:

$$\min_{\mathbf{R},\mathbf{t}} \quad \sum_{i=1}^{M} \big(D_i(\mathbf{R},\mathbf{t})\big)^2 + I_{SO(d)}(\mathbf{R}), \tag{1}$$

where $D_i(\mathbf{R},\mathbf{t})$ is the distance from the transformed point $\mathbf{R}\mathbf{p}_i + \mathbf{t}$ to the target set Q:

$$D_i(\mathbf{R},\mathbf{t}) = \min_{\mathbf{q}\in Q} \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}\|, \tag{2}$$

and $I_{SO(d)}(\cdot)$ is an indicator function for the special orthogonal group $SO(d)$, which requires $\mathbf{R}$ to be a rotation matrix:

$$I_{SO(d)}(\mathbf{R}) = \begin{cases} 0, & \text{if } \mathbf{R}^T\mathbf{R} = \mathbf{I} \text{ and } \det(\mathbf{R}) = 1, \\ +\infty, & \text{otherwise.} \end{cases} \tag{3}$$

The ICP algorithm [1] solves this problem using an iterative approach that alternates between the following two steps:

- *Correspondence step*: transform each point $\mathbf{p}_i \in P$ using the current rotation $\mathbf{R}^{(k)}$ and translation $\mathbf{t}^{(k)}$, and find its closest point $\widehat{\mathbf{q}}_i^{(k)}$ in Q:

$$\widehat{\mathbf{q}}_i^{(k)} = \operatorname*{argmin}_{\mathbf{q}\in Q} \left\| \mathbf{R}^{(k)}\mathbf{p}_i + \mathbf{t}^{(k)} - \mathbf{q} \right\|. \tag{4}$$

- *Alignment step*: update the transformation by minimizing the ℓ_2 distance between the corresponding points:

$$\begin{aligned} &(\mathbf{R}^{(k+1)}, \mathbf{t}^{(k+1)}) \\ &= \operatorname*{argmin}_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{M} \left\| \mathbf{R}\mathbf{p}_i + \mathbf{t} - \widehat{\mathbf{q}}_i^{(k)} \right\|^2 + I_{SO(d)}(\mathbf{R}). \end{aligned} \tag{5}$$

The alignment step can be solved in closed form via SVD [43].

This iterative approach can be considered as applying the majorization-minimization (MM) algorithm [44] to solve the optimization problem (1). To minimize a target function $f(x)$, each iteration of the MM algorithm constructs from the current iterate $x^{(k)}$ a surrogate function $g(x \mid x^{(k)})$ that bounds $f(x)$ from above, such that:

$$\begin{aligned} f(x^{(k)}) &= g(x^{(k)} \mid x^{(k)}), \\ f(x) &\le g(x \mid x^{(k)}), \quad \forall\, x \neq x^{(k)}. \end{aligned} \tag{6}$$

The surrogate function is minimized to obtain the next iterate

$$x^{(k+1)} = \operatorname*{argmin}_{x} g(x \mid x^{(k)}). \tag{7}$$

Equations (6) and (7) imply that

$$f(x^{(k+1)}) \le g(x^{(k+1)} \mid x^{(k)}) \le g(x^{(k)} \mid x^{(k)}) = f(x^{(k)}).$$

Therefore, the MM algorithm decreases the target function monotonically until it converges to a local minimum. To see that classical ICP is indeed an MM algorithm, note that the target function for the alignment step is a surrogate function for the optimization target function in Eq. (1) and satisfies the conditions (6). More concretely, since the closest point $\widehat{\mathbf{q}}_i^{(k)}$ is determined from $\mathbf{R}^{(k)}, \mathbf{t}^{(k)}$, we denote each distance value in (5) as

$$d_i(\mathbf{R},\mathbf{t} \mid \mathbf{R}^{(k)},\mathbf{t}^{(k)}) = \left\| \mathbf{R}\mathbf{p}_i + \mathbf{t} - \widehat{\mathbf{q}}_i^{(k)} \right\|.$$

Then from Equations (4) and (2), we have

$$d_i(\mathbf{R}^{(k)},\mathbf{t}^{(k)} \mid \mathbf{R}^{(k)},\mathbf{t}^{(k)}) = D_i(\mathbf{R}^{(k)}\mathbf{p}_i + \mathbf{t}^{(k)}).$$

Moreover, form Equation (2), for any $\mathbf{R}, \mathbf{t}$:

$$\begin{aligned} D_i(\mathbf{R}\mathbf{p}_i + \mathbf{t}) &= \min_{\mathbf{q}\in Q} \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}\|^2 \\ &\le \left\| \mathbf{R}\mathbf{p}_i + \mathbf{t} - \widehat{\mathbf{q}}_i^{(k)} \right\| = d_i(\mathbf{R},\mathbf{t} \mid \mathbf{R}^{(k)},\mathbf{t}^{(k)}). \end{aligned}$$

Thus each squared distance term in Eq. (5) is a surrogate function for the corresponding term $\big(D_i(\mathbf{R},\mathbf{t})\big)^2$ in Eq. (1), and the target function of the alignment step is a surrogate function for the overall target function (1) constructed from the $\mathbf{R}^{(k)}$ and $\mathbf{t}^{(k)}$. Therefore, the ICP algorithm is an MM algorithm that decreases the target function of (1) in each iteration until convergence.

4 Fast and Robust ICP

Although the ICP algorithm enjoys fast decrease of target energy in the first few iterations, the decrease can slow down in subsequent iterations and result in slow convergence to the optimal alignment. Indeed, it has been proved in [2] that classical ICP only converges linearly. In this section, we interpret ICP as a fixed-point iteration of the transformation, and propose a method to improve its convergence rate using Anderson acceleration [8], [32], an established technique for accelerating a fixed-point iteration. In addition, classical ICP can lead to erroneous alignment in the presence of outliers and partial overlaps, due to the use of ℓ_2 distance as the error metric in the alignment step. We propose a robust error metric based on Welsch's function, to replace the ℓ_2 metric used in classical ICP. We derive an MM solver for the resulting optimization problem, and apply Anderson acceleration to speed up its convergence. In the following, we start by reviewing the basics of Anderson acceleration, followed by the presentation of our methods.

4.1 Preliminary: Anderson Acceleration

Given a fixed-point iteration $x^{(k+1)} = G(x^{(k)})$, we define its *residual* function as $F(x) = G(x) - x$, and denote $F^{(k)} = G(x^{(k)})$. Then by definition a fixed-point x^* of the mapping $G(\cdot)$ satisfies $F(x^*) = 0$. Anderson acceleration utilizes the latest iterate $x^{(k)}$ as well its preceding m iterates

$x^{(k-m)}, x^{(k-m+1)}, \ldots, x^{(k-1)}$ to derive a new iterate $x_{\text{AA}}^{(k+1)}$ that convergences faster to a fixed point [8]:

$$x_{\text{AA}}^{(k+1)} = (1-\beta)\Big(x^{(k)} - \sum_{j=1}^{m} \theta_j^*(x^{(k-j+1)} - x^{(k-j)})\Big) + \beta\Big(G(x^{(k)}) - \sum_{j=1}^{m} \theta_j^*\big(G(x^{(k-j+1)}) - G(x^{(k-j)})\big)\Big), \tag{8}$$

where $(\theta_1^*, \ldots, \theta_m^*)$ is the solution to the following linear least-squares problem:

$$(\theta_1^*, \ldots, \theta_m^*) = \operatorname{argmin} \Big\| F^{(k)} - \sum_{j=1}^{m} \theta_j \big(F^{(k-j+1)} - F^{(k-j)}\big) \Big\|^2,$$

and $\beta \in (0, 1]$ is a mixing parameter. In this paper, we follow the conventional choice $\beta = 1$ [8]. Anderson acceleration has been shown to be a quasi-Newton method for finding a root of the residual function, and a multi-dimensional generalization of the secant method for root-finding [34]. It has been proved recently that Anderson acceleration can improve the convergence rate of fixed-point iterations that converge linearly [45].

4.2 Applying Anderson Acceleration to ICP

We first note that the classical ICP as explained in Section 3 can be written as

$$(\mathbf{R}^{(k+1)}, \mathbf{t}^{(k+1)}) = G_{\text{ICP}}(\mathbf{R}^{(k)}, \mathbf{t}^{(k)}), \tag{9}$$

where

$$G_{\text{ICP}}(\mathbf{R}^{(k)}, \mathbf{t}^{(k)}) = \operatorname*{argmin}_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{M} \Big\| \mathbf{R}\mathbf{p}_i + \mathbf{t} - \Pi_Q(\mathbf{R}^{(k)}\mathbf{p}_i + \mathbf{t}^{(k)}) \Big\|^2 + I_{SO(d)}(\mathbf{R}),$$

with $\Pi_Q(\cdot)$ denoting the closest projection onto the point set Q. Eq. (9) shows that ICP is a fixed-point iteration of the rotation matrix $\mathbf{R}$ and the translation vector $\mathbf{t}$. However, we cannot directly apply Anderson acceleration to it. This is because Anderson acceleration will compute the new value of $\mathbf{R}$ as an affine combination of rotation matrices, which is in general not a rotation matrix itself. To address this issue, we can parameterize a rigid transformation using another set of variables $\mathbf{X}$, such that any value of $\mathbf{X}$ corresponds to a valid rigid transformation, and the ICP iteration can be re-written in the form of

$$\mathbf{X}^{(k+1)} = \overline{G}_{\text{ICP}}(\mathbf{X}^{(k)}). \tag{10}$$

Then we can apply Anderson acceleration to the variable $\mathbf{X}$ by performing the following steps in each iteration:

1) From the current variable $\mathbf{X}^{(k)}$, recover the rotation matrix $\mathbf{R}^{(k)}$ and translation vector $\mathbf{t}^{(k)}$.
2) Perform the ICP update $(\mathbf{R}', \mathbf{t}') = G_{\text{ICP}}(\mathbf{R}^{(k)}, \mathbf{t}^{(k)})$.
3) Compute the parameterization of $(\mathbf{R}', \mathbf{t}')$ to obtain $\overline{G}_{\text{ICP}}(\mathbf{X}^{(k)})$.
4) Compute the accelerated value $\mathbf{X}_{\text{AA}}$ with Eq. (8) using $\mathbf{X}^{(k-m)}, \ldots, \mathbf{X}^{(k)}$ and $G_{\text{ICP}}(\mathbf{X}^{(k-m)}), \ldots, G_{\text{ICP}}(\mathbf{X}^{(k)})$.

One possible way to parameterize rigid transformations is to represent $\mathbf{T}$ as the concatenation of the translation vector and the Euler angles for the rotation [46], [47]. This is the approach taken by the AA-ICP method [11] for applying Anderson acceleration to ICP in $\mathbb{R}^3$. However, it is well known that the Euler angle representation has singularities called the *gimbal lock* [46]. This can affect the performance of AA-ICP when the optimal rotation is close to a gimbal lock (see Fig. 2 for an example). An alternative representation of rotation in $\mathbb{R}^3$ without such singularity is the unit quaternions, which are identified with unit vectors in $\mathbf{R}^4$ [46]. However, this representation is not suitable for Anderson acceleration either, as an affine combination of unit vectors does not result in a unit vector in general. In this paper, we adopt a different parameterization that does not suffer from the above shortcomings of Euler angles and unit quaternions. Our key observation is that all rigid transformations in $\mathbb{R}^d$ form the *special Euclidean group* $SE(d)$, which is a *Lie group* and gives rise to a *Lie algebra* $se(d)$ that is a vector space [48]. From a differential geometry perspective, $SE(d)$ is a smooth manifold and $se(d)$ is its tangent space at the identity transformation. We can then parameterize rigid transformations using their corresponding elements in $se(d)$.

Specifically, if we represent each point $\mathbf{p} \in \mathbb{R}^d$ using its homogeneous coordinates $\tilde{\mathbf{p}} = [\mathbf{p}^T, 1]^T$, then a rigid transformation in $\mathbb{R}^d$ with rotation $\mathbf{R} \in \mathbb{R}^{d\times d}$ and translation $\mathbf{t} \in \mathbb{R}^d$ can be represented as a transformation matrix

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathbb{R}^{(d+1)\times(d+1)}$$

for the homogeneous coordinates. All such matrices form the special Euclidean group $SE(d)$. Its Lie algebra $se(d)$ contains matrices of the following form

$$\check{\mathbf{T}} = \begin{bmatrix} \mathbf{S} & \mathbf{u} \\ \mathbf{0} & 0 \end{bmatrix} \in \mathbb{R}^{(d+1)\times(d+1)}, \tag{11}$$

Each matrix $\check{\mathbf{T}} \in se(d)$ corresponds to a matrix $\mathbf{T} \in SE(d)$ via the matrix exponential:

$$\mathbf{T} = \exp(\check{\mathbf{T}}) = \sum_{i=0}^{\infty} \frac{1}{i!} \check{\mathbf{T}}^i. \tag{12}$$

The matrix exponential can be computed numerically using a generalization of Rodrigues' method as explained in [49]. On the other hand, given a matrix $\mathbf{T} \in SE(d)$, there may be more than one matrix $\check{\mathbf{T}} \in se(d)$ that satisfies Eq. (12). In Appendix A, we present a method to determine a unique value of $\check{\mathbf{T}}$. We call it the *logarithm* of $\mathbf{T}$, and denote it by

$$\check{\mathbf{T}} = \log(\mathbf{T}). \tag{13}$$

We then parameterize elements in $SE(d)$ using their logarithms in $se(d)$, and perform Anderson acceleration within $se(d)$. As $se(d)$ is a vector space, the accelerated value $\check{\mathbf{T}}_{\text{AA}}$, which is computed as an affine combination of elements in $se(d)$, also belongs to $se(d)$ and represents a rigid transformation $\mathbf{T}_{\text{AA}} = \exp(\check{\mathbf{T}}_{\text{AA}}) \in SE(d)$.

Simply applying Anderson acceleration as explained in Section 4.1 is often not sufficient for faster convergence. It is known that Anderson acceleration can suffer from instability and stagnation even for linear problems [42], thus safeguarding steps are often necessary to improve its performance [9], [41], [50]. To this end, we follow the stabilization strategy proposed in [9]: we accept the accelerated value as the new iterate only if it decreases the target function (1) compared

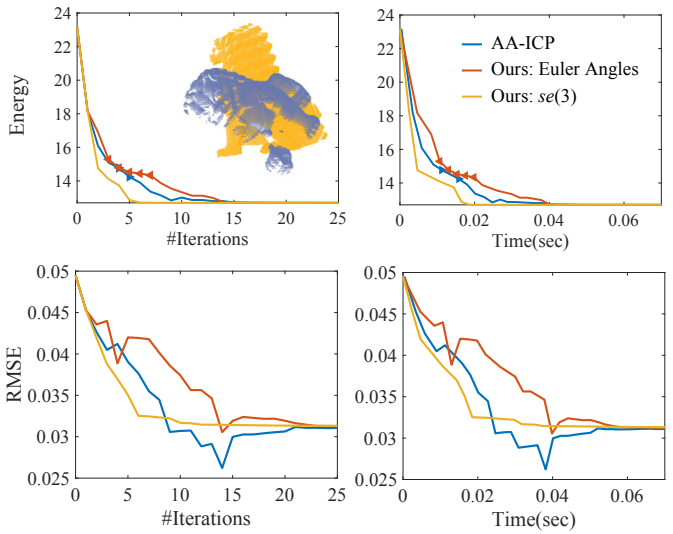

Fig. 2. Target energy and RMSE plots for Anderson-accelerated ICP methods on a pair of point clouds, using different transformation representations and stabilization strategies. Our formulation outperforms AA-ICP [11] as well as a Euler angle-based method using our stabilization strategy. For the two Euler angle-based methods, we use solid triangle symbols to highlight the iterations that are close to the gimbal lock in the energy plots.

with the previous iterate; otherwise, we resort to the unaccelerated ICP iterate as the new iterate. This approach is more simple than the multiple heuristics employed in [11], while ensuring monotonic decrease of the target energy. Following [9], we set the number of previous iterates for Anderson acceleration to $m = 5$ in all experiments.

Compared with the AA-ICP method [11] that also applies Anderson acceleration to ICP, our approach differs in two aspects. First, we apply Anderson acceleration via the Lie algebra $se(d)$ instead of the Euler angles used in [11], which is free from the singularities of gimbal locks. Second, we use a simple energy check to stabilize Anderson acceleration, rather than the multiple heuristical approaches taken in [11], which is more efficient to apply and ensures monotonic energy decrease. Fig. 2 compares the performance between our approach and AA-ICP. For a complete comparison, we also apply Anderson acceleration to rigid transformations representation based on Euler angles and displacement vectors together with our stabilization strategy. The comparison is done on a synthetic model from [28] for which the ground-truth alignment is known. The two point sets are pre-aligned using Super4PCS [19], and then registered using different methods. To show the convergence speed, we plot the value of target function (1) with respect to the iteration count and computational time, as well as the following root mean square error (RMSE) between the computed alignment $(\mathbf{R}, \mathbf{t})$ from the ground-truth alignment $(\mathbf{R}^*, \mathbf{t}^*)$:

$$r = \sqrt{\frac{1}{M} \sum_{i=1}^{M} \|\mathbf{R}^* \mathbf{p}_i + \mathbf{t}^* - \mathbf{R}\mathbf{p}_i - \mathbf{t}\|_2^2}. \qquad (14)$$

Fig. 2 shows that our approach based on the Lie algebra leads to faster convergence. In the energy-iteration plots, for each Euler angle-based approach we also use solid triangle symbols to highlight the iterations that are close to the gimbal lock (with the pitch angle less than 0.01π away from $\pm\pi/2$).

One benefit of our $se(d)$-based approach is that such representation provides a geometrically meaningful space of rigid transformations. Indeed, a similar approach was utilized in [51] to define linear combinations of transformations for 3D computer graphics applications.

4.3 Robust ICP via Welsch's Function

Classical ICP measures the alignment error using ℓ_2 distance, which penalizes large deviation from *any* point in the source set P to the target set Q. This allows for a closed-form solution in the alignment step, but may lead to erroneous alignment in the presence of outliers and partial overlaps. This is because in such cases some points in the source set may not correspond to any point in the target set, and the ground-truth alignment can induce a large error that would be prohibited by the ℓ_2 minimization. This issue can be resolved by adopting error metrics that promote *group-sparsity* of the point-wise distance between the point sets. Such metrics penalize the distance between points in true correspondence, while allowing for large deviation between some points such that outliers and partial overlaps can be accommodated. One example is the group ℓ_p-norm of the point-wise distance with $p \in (0, 1)$, resulting in the error metric $\sum_{i=1}^{M} \Big(D_i(\mathbf{R}, \mathbf{t})\Big)^p$. This formulation is used in [6] for the sparse ICP algorithm in $\mathbb{R}^3$. Similar to classical ICP, the sparse ICP algorithm alternates between closest point query and alignment update. The alignment update is similar to Eq. (5), with the squared ℓ_2 distance in the target function replaced by the group ℓ_p distance. The alignment update is computed using the alternating direction method of multipliers (ADMM), since there is no closed-form solution to the ℓ_p norm minimization problem in general. Although sparse ICP leads to more accurate results, the use of ADMM in the alignment incurs a much higher computational cost than classical ICP. Moreover, the ADMM formulation requires $d \cdot M$ auxiliary variables and $d \cdot M$ dual variables, which can significantly increase the memory footprint.

In this paper, we propose a robust error metric that does not incur high computational overhead. Specifically, we formulate the registration problem as

$$\min_{\mathbf{R},\mathbf{t}} \quad \sum_{i=1}^{M} \psi_\nu\big(D_i(\mathbf{R}, \mathbf{t})\big) + I_{SO(d)}(\mathbf{R}), \qquad (15)$$

where ψ_ν is the Welsch's function [52]:

$$\psi_\nu(x) = 1 - \exp\Big(-\frac{x^2}{2\nu^2}\Big), \qquad (16)$$

and $\nu > 0$ is a user-specified parameter. Fig. 3 shows the graphs of ψ_ν with different values of ν. Function $\psi_\nu(x)$ vanishes at $x = 0$, and is monotonically increasing on $[0, +\infty)$. Thus our formulation penalizes deviation between the point sets. At the same time, ψ_ν is upper bounded by 1, so that our metric is not sensitive to outliers and partial overlaps that cause large deviation between the point sets. Moreover, ν approaches zero, $\sum_{i=1}^{M} \psi_\nu\big(D_i(\mathbf{R}, \mathbf{t})\big)$ approaches the ℓ_0-norm of the distance vector $[D_1(\mathbf{R}, \mathbf{t}), \ldots, D_M(\mathbf{R}, \mathbf{t})]$. Thus our formulation promotes group sparsity of the point-wise deviation between the point sets. Recently, error metrics based on Welsch's function have been applied for robust filtering in image processing [53] and geometry processing [54].

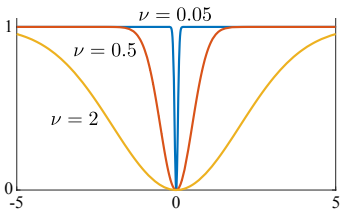

Fig. 3. The graphs of function $\psi_\nu(x)$ with different parameters. As ν decreases, the function ψ_ν approaches the ℓ_0 norm.

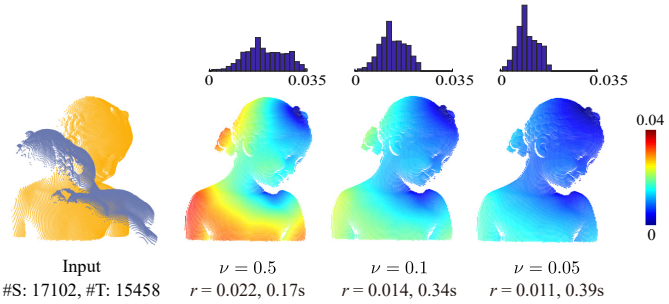

Fig. 4. Comparison of registration results via optimization (15) with different values of parameter ν, on a pair of point clouds with partial overlap. The color-coding shows the deviation between the transformed positions of each source point using the computed alignment and ground-truth alignment, with the histograms showing the distribution of the deviation among all source points. For this model, a smaller value of ν leads to a more accurate result.

Although our formulation (15) is non-linear and non-convex, the optimization can be performed using the same MM framework as classical ICP, by alternating between a correspondence step and an alignment step. The correspondence step is the same as classical ICP, finding the closest point $\widehat{\mathbf{q}}_i^{(k)} \in Q$ for each point $\mathbf{p}_i$ in the source set. Then in the alignment step, we utilize the closest points to the following surrogate for the target function (15) at the current transformation $(\mathbf{R}^{(k)}, \mathbf{t}^{(k)})$ (see Appendix B for a proof):

$$\sum\nolimits_{i=1}^{M} \chi_\nu\Big(\|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \widehat{\mathbf{q}}_i^{(k)}\| \;\Big|\; D_i(\mathbf{R}^{(k)}, \mathbf{t}^{(k)})\Big) + I_{SO(d)}(\mathbf{R}), \tag{17}$$

where $\chi_\nu(x \mid y)$ is a quadratic surrogate function for the Welsch's function at y with the following form [53]:

$$\chi_\nu(x \mid y) = \psi_\nu(y) + \frac{x^2 - y^2}{2\nu^2} \exp\Big(-\frac{y^2}{2\nu^2}\Big). \tag{18}$$

We minimize the surrogate function (17) to update the transformation, resulting in the following problem:

$$\begin{aligned} &(\mathbf{R}^{(k+1)}, \mathbf{t}^{(k+1)}) \\ &= \operatorname*{argmin}_{\mathbf{R},\mathbf{t}} \sum\nolimits_{i=1}^{M} \omega_i \left\|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \widehat{\mathbf{q}}_i^{(k)}\right\|^2 + I_{SO(d)}(\mathbf{R}), \end{aligned} \tag{19}$$

where $\omega_i = \exp\Big(-\|\mathbf{R}^{(k)}\mathbf{p}_i + \mathbf{t}^{(k)} - \widehat{\mathbf{q}}_i^{(k)}\|^2/(2\nu^2)\Big)$. The alignment step (19) minimizes a weighted sum of squared distance between the points $\{\mathbf{p}_i\}$ and $\{\widehat{\mathbf{q}}_i^{(k)}\}$. It can be solved in closed form via SVD [43]. Similar to classical ICP, our MM solver decreases the target energy in each iteration and converges to a local minimum. Using the same approach as in Section 4.2, we improve its convergence rate by applying Anderson acceleration to the parameterization of rigid transformations in $se(d)$, using the same stabilization strategy that checks the target function value for the accelerated value.

Our approach has a similar structure as the iteratively reweighted least squares (IRLS) method that minimizes the ℓ_p-norm ($p < 1$) for compressive sensing [55]. Similar to IRLS, we solve a weighted least squares problem, with the weights ω_i for a point $\mathbf{p}_i$ updated in each iteration according to its current distance to the corresponding point. Since the weight is a Gaussian function with variance ν, a point $\mathbf{p}_i$ with larger distance from the target point set receives a lower weight. Moreover, according to the well-know three-sigma rule, when the distance is larger than 3ν, the weight ω_i is small enough such that the term for $\mathbf{p}_i$ has little influence to the target function and $\mathbf{p}_i$ is effectively excluded from the current alignment problem. In this way, the optimization allows some source points to be far away from the target point set, and is robust to outliers and partial overlaps.

Some existing ICP variants improve robustness by excluding from the alignment step the point pairs whose difference in positions or normals is larger than a threshold [5]. It was observed in [6] that such approaches can be difficult to tune or increase the number of local minima. In contrast, our method also excludes point pairs with large positional difference, but using a Gaussian weight that gradually decreases as the point pairs becomes further apart. It can be considered as a soft thresholding approach that weakly penalizes outliers, which can lead to more stable results [6]. Indeed, we observe in experiments that our robust methods and sparse ICP tend to produce more accurate results than the symmetric ICP method [15] which is based on outlier rejection; see Section 6 for details.

Compared to robust registration based on ℓ_p-norm minimization ($0 < p < 1$), our formulation and solver also offer benefits in stability and convergence guarantee. For our weighted least-squares problem (19), all the Gaussian weights $\{\omega_i\}$ have values with the range $(0, 1]$. In contrast, an IRLS solver for ℓ_p-norm minimization would assign a weight $\|\mathbf{R}^{(k)}\mathbf{p}_i + \mathbf{t}^{(k)} - \widehat{\mathbf{q}}_i^{(k)}\|^{p-2}$ to the point $\mathbf{p}_i$, which could go to infinity and cause instability when the alignment error for $\mathbf{p}_i$ approaches zero [6]. According to [6], it is due to such concern about stability that the sparse ICP algorithm performs ℓ_p-norm minimization using ADMM instead of IRLS. In addition, the convergence of IRLS and ADMM for the non-convex ℓ_p-norm minimization ($0 < p < 1$) requires strong assumptions about the problem such as the Kurdyka-Łojasiewicz property [56], [57], whereas our MM solver is guaranteed to converge.

For our algorithm, the choice of the parameter ν plays an important role in achieving good performance. Setting a smaller value of ν helps to attenuate the influence from outliers and partial overlaps to achieve robustness (see Fig. 4 for an example). On the other hand, setting a larger value of ν in the initial stage helps to include more pairs of points in the alignment step and avoid converging to an undesirable local minimum. Therefore, we gradually decrease the value of ν during the iterations, so that the algorithm first performs more global alignment with a larger number of pairs, and then reduces the influence from the pairs with large deviation to achieve robust alignment. Specifically, we choose two

Algorithm 1: Robust point-to-point ICP using Welsch's function and Anderson acceleration.

Input: $\mathbf{T}^{(0)}$: initial transformation for P;
m: the number of previous iterates used for Anderson acceleration;
correspondence$(\mathbf{T})$: computation of all closest points according via Eq. (4) using transformation $\mathbf{T}$;
alignment$(\widehat{\mathbf{Q}}, \mathbf{T}, \nu)$: new transformation via Eq. (19) using current transformation $\mathbf{T}$ and closest points $\widehat{\mathbf{Q}}$;
$E_\nu(\widehat{\mathbf{Q}}, \mathbf{T})$: target energy for transformation $\mathbf{T}$ and closest points $\widehat{\mathbf{Q}}$: $E_\nu(\widehat{\mathbf{Q}}, \mathbf{T}) = \sum_{i=1}^{M} \psi_\nu(\|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \widehat{\mathbf{q}}_i\|)$;
I_ν, ϵ_ν: maximum number of iterations and the convergence threshold of $\mathbf{T}$ for a given parameter ν.

$k = 1$; $\nu = \nu_{max}$; $\widehat{\mathbf{Q}}^{(0)} = \textit{correspondence}(\mathbf{T}^{(0)})$;
while `TRUE` **do**
  $k_{\text{start}} = k - 1$; $E_{\text{prev}} = +\infty$;
  $\mathbf{T}' = \textit{alignment}(\widehat{\mathbf{Q}}^{(k-1)}, \mathbf{T}^{(k-1)}, \nu)$;
  $\mathbf{T}^{(k)} = \mathbf{T}'$; $\widehat{\mathbf{Q}}^{(k)} = \textit{correspondence}(\mathbf{T}^{(k)})$;
  $G^{(k-1)} = \log(\mathbf{T}')$; $F^{(k-1)} = G^{(k-1)} - \log(\mathbf{T}^{(k-1)})$;
  while $k - k_{\text{start}} \leq I_\nu$ **do**
    `// Ensure` $\mathbf{T}^{(k)}$ `decreases the energy`
    if $E_\nu(\widehat{\mathbf{Q}}^{(k)}, \mathbf{T}^{(k)}) \geq E_{\text{prev}}$ **then**
      $\mathbf{T}^{(k)} = \mathbf{T}'$;
      $\widehat{\mathbf{Q}}^{(k)} = \textit{correspondence}(\mathbf{T}^{(k)})$;
    end
    $E_{\text{prev}} = E_\nu(\widehat{\mathbf{Q}}^{(k)}, \mathbf{T}^{(k)})$;
    `// Check convergence`
    $\mathbf{T}' = \textit{alignment}(\widehat{\mathbf{Q}}^{(k)}, \mathbf{T}^{(k)}, \nu)$;
    if $\|\mathbf{T} - \mathbf{T}'\|_F < \epsilon_\nu$ **then**
      break;
    end
    `// Anderson acceleration`
    $G^{(k)} = \log(\mathbf{T}')$; $F^{(k)} = G^{(k)} - \log(\mathbf{T}^{(k)})$;
    $m_k = \min(k - k_{\text{start}}, m)$;
    $(\theta_1^*, \ldots, \theta_{m_k}^*) = \operatorname{argmin} \|F^{(k)} - \sum_{j=1}^{m_k} \theta_j (F^{(k-j+1)} - F^{(k-j)})\|_F^2$;
    $\mathbf{T}^{(k+1)} = \exp\big(G^{(k)} - \sum_{j=1}^{m} \theta_j^* (G^{(k-j+1)} - G^{(k-j)})\big)$;
    $\widehat{\mathbf{Q}}^{(k+1)} = \textit{correspondence}(\mathbf{T}^{(k+1)})$;
    $k = k + 1$;
  end
  if $\nu == \nu_{\min}$ **then**
    return $\mathbf{T}^{(k)}$;
  end
  $\nu = \max(\nu/2, \nu_{\min})$; $k = k + 1$;
end

values $\nu_{\max}$ and $\nu_{\min}$ as the upper and lower bounds of μ. We start by setting $\nu = \nu_{\max}$ and running our MM algorithm until the change in the transformation matrix $\mathbf{T}$ is smaller than a threshold (10^{-5} by default) or the iteration count exceeds an upper limit (1000 by default). Then we decrease the value of ν by half, and run the MM algorithm again until the same termination criterion is met. The process is repeated until the lower bound $\nu_{\min}$ is reached. Algorithm 1 summarizes our method with a decreasing ν.

For the value of $\nu_{\max}$, we compute the median $\overline{D}^{(0)}$ among all initial point-wise distance $\{D_i(\mathbf{R}^{(0)}, \mathbf{t}^{(0)})\}$, and set $\nu_{\max} = 3 \cdot \overline{D}^{(0)}$. In our experiments, this setting makes $\nu_{\max}$ large enough to include most point pairs into the alignment process except for outliers with significant deviation.

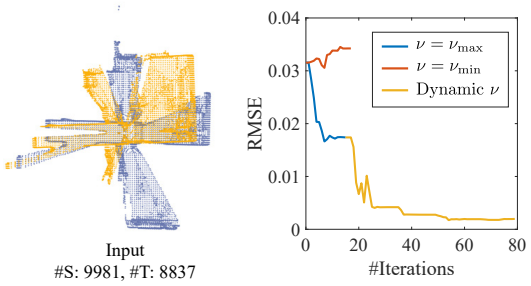

Fig. 5. Comparison of registration performance via optimization (15) with different settings of ν, on a pair of point clouds in the 'Apartment' sequence from the ETH laser registration dataset [58]. Gradually reducing ν from $\nu_{\max}$ to $\nu_{\min}$ results in the lowest RMSE.

For $\nu_{\min}$, we note that the two point sets may sample the same surface region at different locations, and the $\nu_{\min}$ should be large enough to accommodate the point-wise deviation due to sampling. Therefore, we first compute the median distance from each point $\mathbf{q}_i \in Q$ to its six nearest neighbors within the target set, and take the median $\overline{E}_Q$ of these median distance values across P. Then we set $\nu_{\min} = \overline{E}_Q / 3\sqrt{3}$. The rationale of this choice is explained in Appendix C.

Fig. 5 illustrates the effectiveness of our update strategy for parameter ν in improving the alignment accuracy, by comparing its RMSE plot with those resulting from a fixed parameter $\nu = \nu_{\max}$ and $\nu = \nu_{\min}$, respectively. Here a fixed $\nu = \nu_{\min}$ results in a large RMSE, since setting such a small ν will lead to a small weight for most point pairs except for those with a small distance, effectively excluding most points from the alignment step and producing an erroneous result. Setting $\nu = \nu_{\max}$ can reduce the final RMSE as it includes more points into the alignment; however, it fails to exclude some outliers so the RMSE is still large. Our dynamic update of ν gradually decreases ν and removes outliers from the alignment process, resulting in a much smaller RMSE.

5 Extension to Point-to-Plane ICP

The classical ICP algorithm discussed in Section 4 is often called the "point-to-point" ICP, since its alignment step minimizes the distance from the source points to their corresponding target points. Another popular ICP method for registration in $\mathbb{R}^3$, which is proposed in [3] and often called the "point-to-plane" ICP, minimizes the distance from to the tangent planes at the target points instead. Its alignment step can be written as

$$(\mathbf{R}^{(k+1)}, \mathbf{t}^{(k+1)}) = \operatorname*{argmin}_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{M} \left((\mathbf{R}\mathbf{p}_i + \mathbf{t} - \widehat{\mathbf{q}}_i^{(k)}) \cdot \widehat{\mathbf{n}}_i^{(k)} \right)^2 + I_{SO(3)}(\mathbf{R}), \tag{20}$$

where $\widehat{\mathbf{n}}_i^{(k)}$ is the normal at $\widehat{\mathbf{q}}_i^{(k)}$ for the underlying surface of the target point set. Point-to-plane ICP can be considered as solving an optimization problem

$$\min_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{M} (H_i(\mathbf{R}, \mathbf{t}))^2 + I_{SO(3)}(\mathbf{R}), \tag{21}$$

where $H_i(\mathbf{R}, \mathbf{t})$ is the signed distance from the transformed source point $\mathbf{R}\mathbf{p}_i + \mathbf{t}$ to the tangent plane at its closest

point in the target set Q. Since the distance to the tangent plane provides a linear approximation of the distance to the underlying surface, point-to-plane ICP can converge faster than its point-to-point counterpart [2]. On the other hand, it suffers from the same limitation as point-to-point ICP in terms of robustness to outliers and partial overlaps. Similar to Section 4, we can improve its robustness by replacing the squared point-to-plane distance in the target function by a robust metric based on the Welsch's function ψ_ν:

$$\min_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{M} \psi_\nu \left(H_i(\mathbf{R},\mathbf{t}) \right) + I_{SO(3)}(\mathbf{R}). \tag{22}$$

This is solved by alternating between a correspondence step the same as point-to-point ICP, and an assignment step that solves the following problem:

$$\min_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{M} \psi_\nu \left((\mathbf{R}\mathbf{p}_i + \mathbf{t} - \widehat{\mathbf{q}}_i^{(k)}) \cdot \widehat{\mathbf{n}}_i^{(k)} \right) + I_{SO(3)}(\mathbf{R}). \tag{23}$$

Similar to Section 4, we replace the target function above with a surrogate function to derive a proxy problem:

$$\min_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{M} \gamma_i \left((\mathbf{R}\mathbf{p}_i + \mathbf{t} - \widehat{\mathbf{q}}_i^{(k)}) \cdot \widehat{\mathbf{n}}_i^{(k)} \right)^2 + I_{SO(3)}(\mathbf{R}), \tag{24}$$

where $\gamma_i = \exp\left(-((\mathbf{R}^{(k)}\mathbf{p}_i + \mathbf{t}^{(k)} - \widehat{\mathbf{q}}_i^{(k)}) \cdot \widehat{\mathbf{n}}_i^{(k)})^2 / (2\nu^2) \right)$. However, there is no closed-form solution to this problem. To update the transformation, we rewrite the problem as an optimization for its $se(3)$ parameterization:

$$\min_{\widetilde{\mathbf{T}}} \sum_{i=1}^{M} \gamma_i \left(B_i^{(k)}(\widetilde{\mathbf{T}}) \right)^2. \tag{25}$$

Here $\widetilde{\mathbf{T}} \in \mathbb{R}^6$ denotes the actual variables for the $se(3)$ element $\check{\mathbf{T}}$ in Eq. (11) (three variables for each of the submatrices $\mathbf{S}$ and $\mathbf{u}$, respectively), and $B_i^{(k)}(\widetilde{\mathbf{T}})$ denotes the signed distance from $\mathbf{R}\mathbf{p}_i + \mathbf{t}$ to the tangent plane at $\widehat{\mathbf{q}}_i^{(k)}$. We then linearize $B_i^{(k)}$ using its first-order Taylor expansion

$$B_i^{(k)}(\widetilde{\mathbf{T}}) \approx B_i^{(k)}(\widetilde{\mathbf{T}}^{(k)}) + (\mathbf{J}_i^{(k)})^T (\widetilde{\mathbf{T}} - \widetilde{\mathbf{T}}^{(k)}),$$

where $\widetilde{\mathbf{T}}^{(k)}$ denotes the parameterization variable for the current transformation $(\mathbf{R}^{(k)}, \mathbf{t}^{(k)})$, and $\mathbf{J}_i^{(k)}$ is the gradient of $\mathbf{B}_i^{(k)}$ at $\widetilde{\mathbf{T}}^{(k)}$. Substituting the linearization into Eq. (25), we obtain a quadratic problem that reduces to a linear system

$$\begin{aligned} &\left(\sum_{i=1}^{M} \gamma_i \mathbf{J}_i^{(k)} (\mathbf{J}_i^{(k)})^T \right) \widetilde{\mathbf{T}} \\ &= \sum_{i=1}^{M} \gamma_i \mathbf{J}_i^{(k)} \left(H_i^{(k)}(\widetilde{\mathbf{T}}^{(k)}) - (\mathbf{J}_i^{(k)})^T \widetilde{\mathbf{T}}^{(k)} \right). \end{aligned} \tag{26}$$

The solution $\widetilde{\mathbf{T}}_*^{(k)}$ to this system will be taken as a candidate for the updated transformation. Due to the linearization, $\widetilde{\mathbf{T}}_*^{(k)}$ may increase the target function (22). Therefore, we perform line search along the direction $\widetilde{\mathbf{T}}_*^{(k)} - \widetilde{\mathbf{T}}^{(k)}$ to find a new transformation that decreases the target function. If such a transformation cannot be found after the maximum number of line-search steps is reached, then the step size with the lowest target function value will be used.

Similar to Section 4, we apply Anderson acceleration to speed up the convergence. We note that the mapping from the current variable $\widetilde{\mathbf{T}}^{(k)}$ to the candidate update $\widetilde{\mathbf{T}}_*^{(k)}$, which amounts to finding the closest points $\{\widehat{\mathbf{q}}_i^{(k)}\}$ according to $\widetilde{\mathbf{T}}^{(k)}$ and solving the linear system (26), can be written as

$$\widetilde{\mathbf{T}}_*^{(k)} = G_{\text{ppl}}(\widetilde{\mathbf{T}}^{(k)}). \tag{27}$$

Then for a local minimum $(\mathbf{R}^*, \mathbf{t}^*)$ of the target function (22), the corresponding parameterization variable $\widetilde{\mathbf{T}}^*$ should be a fixed point of G_{ppl}. Therefore, we can apply Anderson acceleration to the history $\widetilde{\mathbf{T}}^{(k-m)}, \ldots, \widetilde{\mathbf{T}}^{(k)}$ and $G_{\text{ppl}}(\widetilde{\mathbf{T}}^{(k-m)}), \ldots, G_{\text{ppl}}(\widetilde{\mathbf{T}}^{(k)})$ to obtain an accelerated value $\widetilde{\mathbf{T}}_{\text{AA}}$. If $\widetilde{\mathbf{T}}_{\text{AA}}$ decreases the target function (22), then we accept

Algorithm 2: Robust point-to-plane ICP using Welsch's function and Anderson acceleration.

Input: $\widetilde{\mathbf{T}}^{(0)}$: initial transformation parameters;
m: the number of previous iterates used for Anderson acceleration;
$G_{\text{ppl}}(\cdot)$: the mapping defined in Eq. (27);
$\widetilde{E}_\nu(\widetilde{\mathbf{T}})$: target energy of problem (22) for parameters $\widetilde{\mathbf{T}}$;
$l_{\max}$: maximum number of inner line search steps;
I_ν, ϵ_ν: maximum number of iterations and the convergence threshold for a given parameter ν.

$k = 1$; $\nu = \nu_{max}$;
while *TRUE* **do**
  $k_{\text{start}} = k - 1$; $E_{\text{prev}} = +\infty$; $\widetilde{\mathbf{T}}_*^{(k)} = G_{\text{ppl}}(\widetilde{\mathbf{T}}^{(k-1)})$;
  $G^{(k-1)} = \widetilde{\mathbf{T}}^{(k)} = \widetilde{\mathbf{T}}_*^{(k)}$; $F^{(k-1)} = G^{(k-1)} - \widetilde{\mathbf{T}}^{(k-1)}$;
  while $k - k_{\text{start}} \le I_\nu$ **do**
    // Check energy decrease
    $E = \widetilde{E}_\nu(\widetilde{\mathbf{T}}^{(k)})$;
    if $E \ge E_{\text{prev}}$ **then**
      // Perform line search
      $\tau = 1$; $l = 1$;
      while $l \le l_{\max}$ **do**
        $\widetilde{\mathbf{T}}_{\text{trial}} = \widetilde{\mathbf{T}}^{(k-1)} + \tau(\widetilde{\mathbf{T}}_*^{(k)} - \widetilde{\mathbf{T}}^{(k-1)})$;
        $E_{\text{trial}} = \widetilde{E}_\nu(\widetilde{\mathbf{T}}_{\text{trial}})$;
        if $E_{\text{trial}} < E$ **then**
          $E = E_{\text{trial}}$; $\widetilde{\mathbf{T}}^{(k)} = \widetilde{\mathbf{T}}_{\text{trial}}$;
        end
        if $E_{\text{trial}} < E_{\text{prev}}$ **then**
          break;
        end
      end
    end
    $E_{\text{prev}} = E$;
    // Check convergence
    $\widetilde{\mathbf{T}}_*^{(k+1)} = G_{\text{ppl}}(\widetilde{\mathbf{T}}^{(k)})$;
    if $\|\widetilde{\mathbf{T}}_*^{(k+1)} - \widetilde{\mathbf{T}}^{(k)}\| < \epsilon_\nu$ **then**
      break;
    end
    // Anderson acceleration
    $G^{(k)} = \widetilde{\mathbf{T}}_*^{(k+1)}$; $F^{(k)} = G^{(k)} - \widetilde{\mathbf{T}}^{(k)}$;
    $m_k = \min(k - k_{\text{start}}, m)$;
    $(\theta_1^*, \ldots, \theta_{m_k}^*) =$
     $\operatorname{argmin} \|F^{(k)} - \sum_{j=1}^{m_k} \theta_j (F^{(k-j+1)} - F^{(k-j)})\|^2$;
    $\widetilde{\mathbf{T}}^{(k+1)} = \exp\big(G^{(k)} - \sum_{j=1}^{m} \theta_j^* (G^{(k-j+1)} - G^{(k-j)})\big)$;
    $k = k + 1$;
  end
  if $\nu == \nu_{\min}$ **then**
    return $\widetilde{\mathbf{T}}^{(k)}$;
  end
  $\nu = \max(\nu/2, \nu_{\min})$; $k = k + 1$;
end

it as the new iterate $\widetilde{\mathbf{T}}^{(k+1)}$. Otherwise, we perform line search as described previously. Algorithm 2 summarizes our robust point-to-plane ICP solver. Similarly to the point-to-point solver, we start with a larger parameter $\nu = \nu_{\max}$, and gradually decreases it until it reaches the lower bound $\nu_{\min}$. For each given ν value, the solver is run until the change in the transformation matrix is smaller than a threshold (10^{-5} by default) or the iteration account reaches a limit (6 for $\nu_{\max}$, then incremented by 1 each time ν is changed, but no larger than 10). We set $\nu_{\max}$ to be three times the median distance from the source points to the tangent planes at their corresponding points in the initial iteration. To determine $\nu_{\min}$, we first compute for each point $\mathbf{q} \in Q$ the median distance from its six nearest neighbors in Q to its tangent plane, and take the median $\overline{H}_Q$ of all such values. Then we set $\nu_{\min} = \overline{H}_Q/6$ (see Appendix C for the rationale).

6 Results

In this section, we evaluate our methods in terms of convergence speed and robustness, and compare them with existing methods including AA-ICP [11], sparse ICP [6], and symmetric ICP [15]. Our comparison includes both point-to-point and point-to-plane ICP methods and their variants. In the figures and tables below, we will denote the point-to-point ICP and its variants as "ICP", whereas point-to-plane ICP and its variants will be denoted as "ICP-*l*". For sparse ICP and symmetric ICP, we use the source codes released by the respective authors[1,2] for the implementation. For symmetric ICP, we use their simple formulation that does not rotate the normals ($\mathcal{E}_{symm}$ as defined in the paper). We implement our methods in C++, using the Eigen library [59] for all linear algebra operations. All examples are run on a desktop PC with 16GB of RAM and a 6-core CPU at 3.6 GHz, using OpenMP for parallelization. For each test problem, we normalize the input data by aligning the centroid of each point cloud to the origin and then uniformly scaling all the points such that their bounding box has diagonal length 1. Unless stated otherwise, the point clouds are pre-aligned using the Super4PCS method from [19], before the alignment is refined using different methods. We test the methods on both synthetic and real-world datasets. Some methods (point-to-plane ICP and variants, as well as symmetric ICP) require normals at the points. For synthetic data where the ground-truth shape of the underlying surface is known, we use the surface normals as the input normals at the points. Otherwise, we use the Point Cloud Library[3] to estimate the normals using 10 nearest neighbors. The source codes for our methods are available at https://github.com/yaoyx689/Fast-Robust-ICP.

Robustness to Noises, Outliers and Partial Overlaps. In Fig. 1, we test the methods on two point sets with only a small overlap region, constructed the monkey model from the EPFL statue dataset[4]. Starting from the full point cloud model, we take the first 60% of the points to create the source set, and the last 47% of the points with a random rigid transformation to construct the target set. Our robust point-to-point and point-to-plane ICP methods achieve the lowest RMSE values among all methods, while taking a much shorter computational time than sparse point-to-point ICP, the method that achieves the next lowest RMSE. In addition, our fast ICP achieves the same RMSE as the classical ICP and AA-ICP with a shorter computational time. Another challenging case for rigid registration are point sets that contain noises and outliers. In Fig. 6, we test the methods on a pair of such point sets that partially overlap, constructed using the Aquarius model from the EPFL statue dataset. Starting from the clean point cloud of the full model, we take the first 60% of the points as the source point cloud, and add Gaussian noises on all points along their normal directions, with the standard deviation being the average value of all points' median distance to their six nearest neighbors. To emulate outliers, we also add $1\% \cdot \overline{M}$ random points using a uniform distribution within the source point cloud's bounding box, where $\overline{M}$ is the number of points in the source point cloud. For the target point cloud, we take the last 42% of the points from the full model, add Gaussian noises and outliers in the same way as the source point cloud, and apply a random transformation. Our robust point-to-plane ICP achieves the lowest RMSE, followed by sparse point-to-plane ICP which is more than an order of magnitude slower.

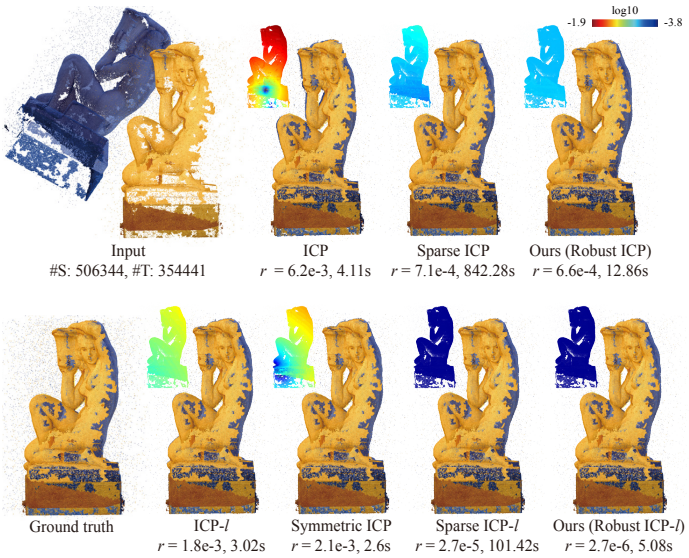

Fig. 6. Comparison of different registration methods on a pair of partially overlapping point clouds with noises and outliers, constructed using the Aquarius model from the EPFL statue dataset.

Synthetic Datasets. We further test the methods on 25 pairs of partially overlapping point clouds constructed from five models in [28], with five pairs for each model. Fig. 7 compares the registration results on some problem instances, showing the computational time and RMSE for each method, and using color-coding to visualize the deviation from the ground-truth alignment. Tab. 1 shows the average computational time and average/median RMSE on each model for different methods. Overall, our robust methods and sparse ICP lead to more accurate results. Our methods achieve best

1. https://github.com/OpenGP/sparseicp
2. https://gfx.cs.princeton.edu/pubs/Rusinkiewicz_2019_ASO/icptests-1.0.zip
3. https://pointclouds.org/
4. https://lgg.epfl.ch/statues_dataset.php

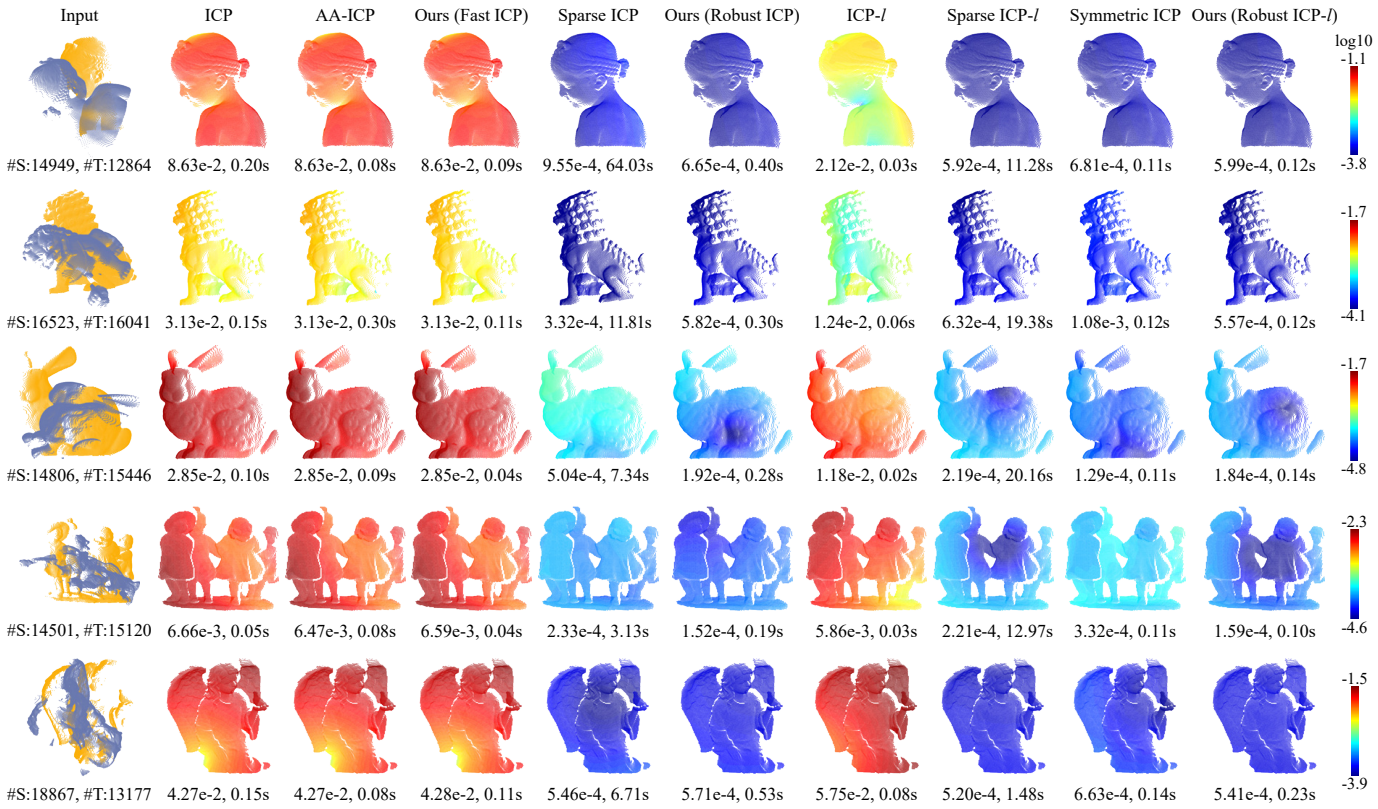

Fig. 7. Examples of registration results using different methods on partially overlapping point clouds, with RMSE and computational time shown below each result. The log-scale color-coding visualizes the deviation between the computed alignment and the ground-truth alignment.

TABLE 1
Average computational time (sec) and average/median RMSE ($\times 10^{-3}$) for different registration methods on partially overlapping point cloud pairs constructed from five models, with five pairs for each model (see Fig. 7). Best performance numbers are highlighted in bold fonts.

Dataset	Bimba		Children		Dragon		Angle		Bunny	
	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE
ICP	0.17	68/60	0.05	9.8/6.7	0.11	21/19	0.09	13/5.6	0.08	26/28
AA-ICP	0.09	68/60	0.05	9.8/6.5	0.12	21/19	0.06	13/5.6	0.06	26/28
Ours (Fast ICP)	**0.08**	68/60	**0.04**	9.8/6.6	**0.07**	21/19	0.06	13/5.6	0.04	26/28
Sparse ICP	35.88	38/0.95	2.71	0.96/0.81	8.01	0.92/0.95	4.99	0.83/**0.97**	7.68	0.94/0.71
Ours (Robust ICP)	0.45	**0.87**/0.67	0.17	0.89/**0.62**	0.21	0.93/**0.92**	0.23	0.83/0.98	0.27	0.85/0.69
ICP-*l*	0.29	78/20	0.04	16/5.9	0.50	14/11	**0.04**	13/2.5	**0.04**	13/12
Sparse ICP-*l*	9.99	3.4/**0.59**	12.39	0.92/0.67	8.40	0.96/0.94	4.12	**0.82**/0.98	6.76	0.81/**0.56**
Symmetric ICP	0.12	54/0.68	0.11	0.99/0.92	0.11	1.1/1.1	0.13	0.89/0.97	0.10	0.84/0.58
Ours (Robust ICP-*l*)	0.15	59/0.6	0.11	**0.88**/0.63	0.11	**0.92**/0.93	0.14	0.82/0.98	0.11	**0.79**/0.56

average/median RMSE measures in more instances, while being significantly faster than sparse ICP.

Real-World Datasets. To evaluate their performance on real-world problems, we test the methods on the RGB-D SLAM dataset [60] and the ETH laser registration dataset [58]. We used three point cloud sequences from RGB-D SLAM dataset: "xyz", "360" and "teddy". From each sequence, we take pairs of point clouds that are five frames apart, to obtain 2956 pairs of point clouds for registration. As each pair of point clouds are already close to each other, we directly apply the registration methods without pre-alignment. We also test all eight point cloud sequences from the ETH laser registration dataset, each containing between 31 and 45 point clouds. We align all pairs of adjacent point clouds from each sequence. All point clouds are pre-processed using a box grid filter to make the density more uniform, which is the same as the pre-processing operation in [61]. Tab. 2 and Tab. 3 show the average computational time and average/median RMSE for each method on the datasets, while Fig. 8 and Fig. 9 show some examples of registration results together with color-coding that visualizes the deviation from the ground-truth alignment. To visualize the distribution of RMSE within the datasets, we also compute the α-*recall rate* for each method, defined as $|S_\alpha|/|S|$, where $|S|$ is the total number of test cases, and $|S_\alpha|$ is the number of test cases where the final RMSE is less than α [28]. Intuitively, for a given α, a higher α-recall rate indicates more test cases with RMSE values bounded from above by α. The plots of α-recall rates for each method are included in Fig. 8 and Fig. 9. Overall, our robust methods achieve the best average/median RMSE in more instances than other methods, followed by sparse ICP and

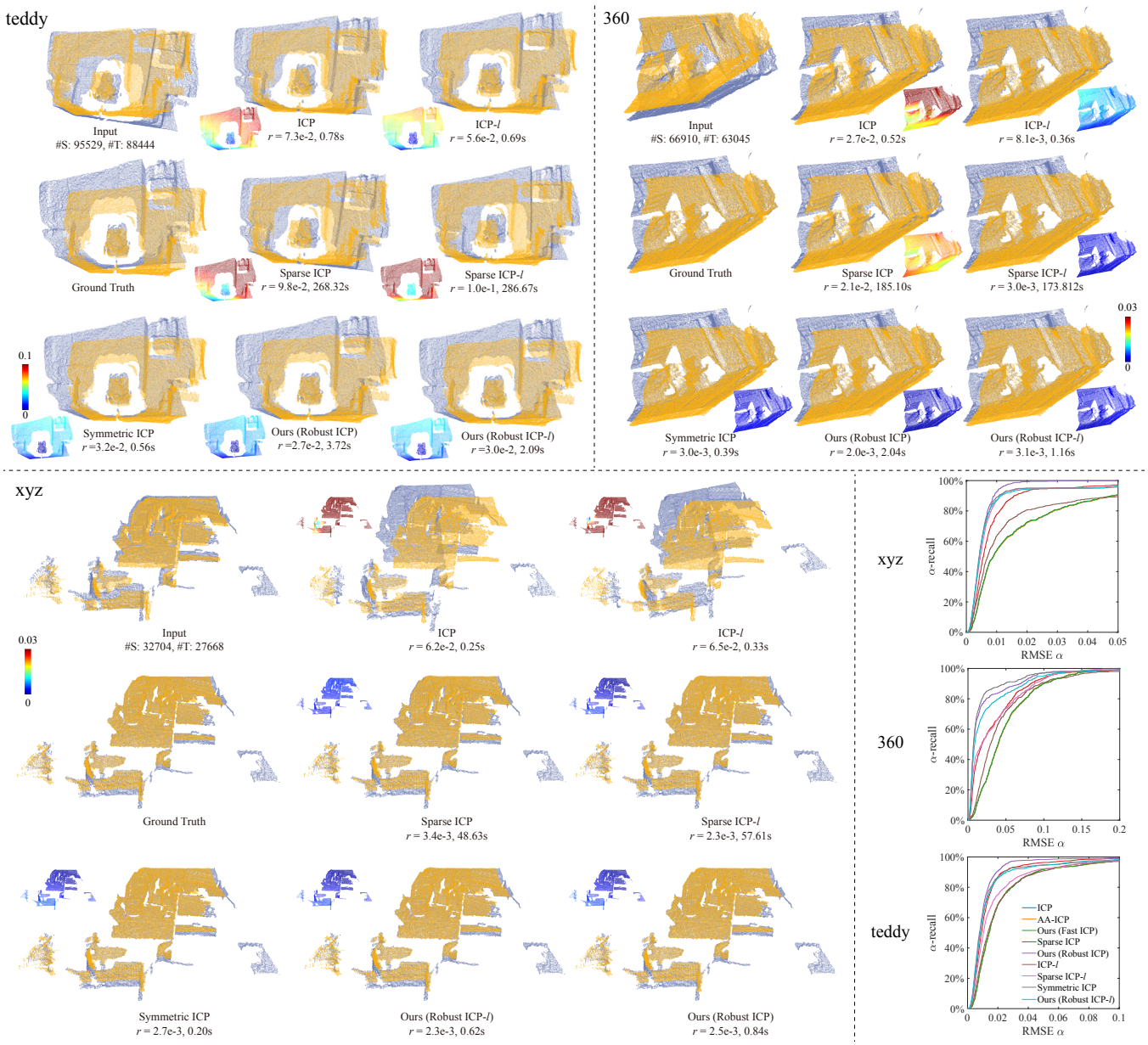

Fig. 8. Examples of registration results using different methods for point clouds from the RGBD-SLAM dataset, with color-coding to visualize the deviation from the ground-truth alignment. The plots on the bottom-right show the α-recall rates of different methods on point cloud pairs in three sequences from the dataset.

symmetric ICP. In terms of speed, our robust methods tend to be slower than symmetric ICP, while both are significantly faster than sparse ICP.

7 Conclusion

In this paper, we propose methods to improve the convergence speed and robustness of point-to-point and point-to-plane ICP methods. We first propose an Anderson-accelerated point-to-point ICP based on Lie algebra parameterization of rigid transformations, together with a stabilization strategy that ensures monotonic decrease of target energy. We also develop a robustified point-to-point ICP formulation based on the Welsch's function, and solve it using an Anderson-accelerated MM solver. Finally, we extend the robust formulation and the accelerated numerical solver

TABLE 2
Average computational time (sec) and average/median RMSE ($\times 10^{-3}$) using different registration methods for the RGBD-SLAM dataset, with best performance numbers highlighted in bold fonts.

Method	xyz		360		teddy	
	Time	RMSE	Time	RMSE	Time	RMSE
ICP	0.19	20/8.9	0.71	50/40	0.60	22/14
AA-ICP	0.14	20/8.9	0.46	50/40	0.40	22/14
Ours (Fast ICP)	**0.11**	20/8.9	**0.33**	50/40	**0.30**	22/14
Sparse ICP	17.71	9.3/5.6	111.58	34/18	90.81	14/9.2
Ours (Robust ICP)	0.56	**5.2**/4.3	2.71	21/**7.3**	2.60	**11**/**7.8**
ICP-*l*	0.99	22/7.1	2.01	45/29	1.25	22/13
Sparse ICP-*l*	27.43	8.9/4.6	69.29	34/18	93.98	19/10
Symmetric ICP	0.17	9.3/**4.2**	0.36	**19**/7.3	0.47	15/8.2
Ours (Robust ICP-*l*)	0.36	8.8/4.3	1.09	25/8.8	1.27	15/8.4

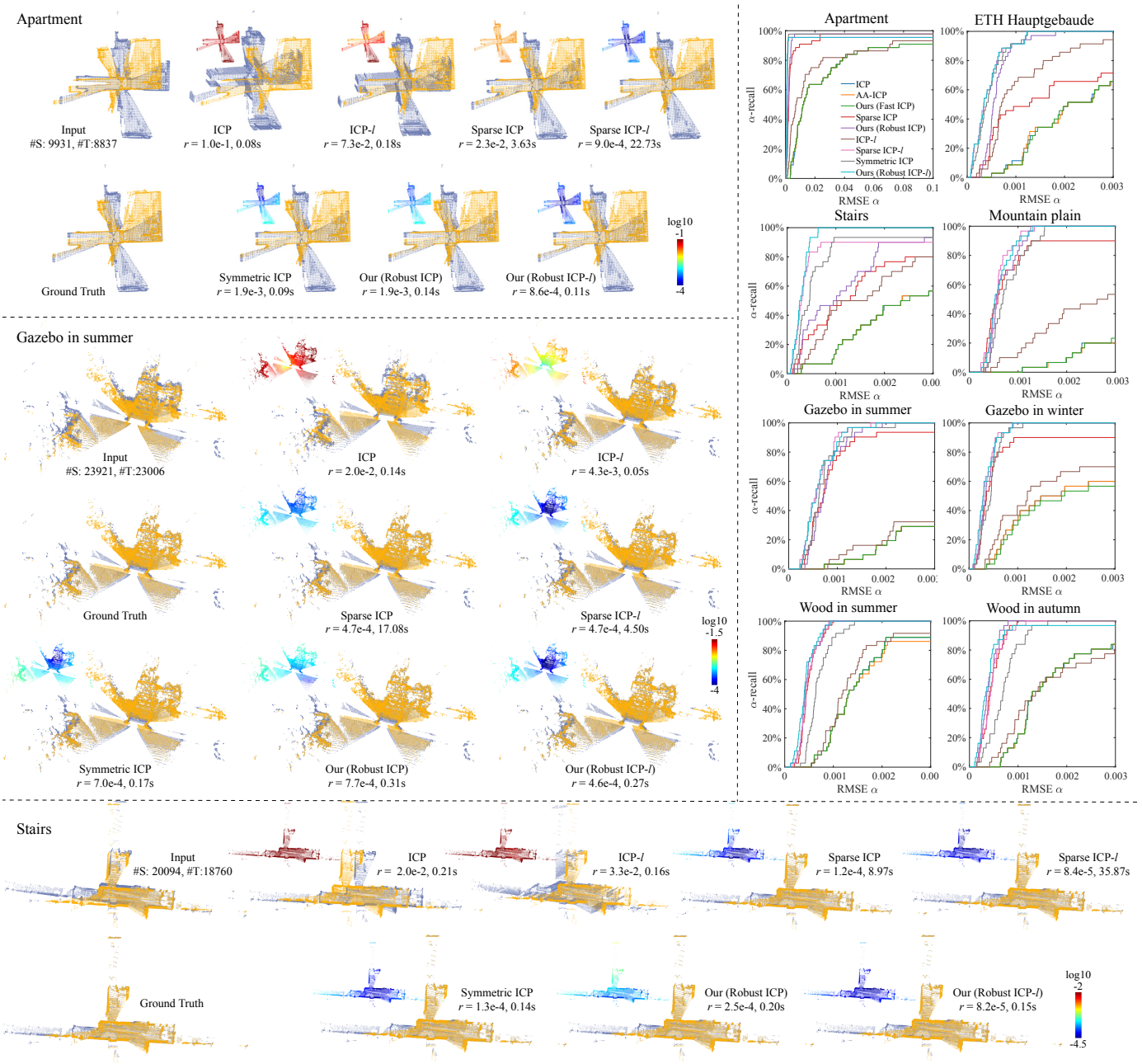

Fig. 9. Examples of registration results with different methods for point clouds from the ETH laser registration dataset, with color-coding for the deviation from the ground-truth alignment. The plots on top-right show the α-recall rates of each method for the point cloud sequences in the dataset.

TABLE 3
Average computational time (sec) and average/median RMSE ($\times 10^{-4}$) for different registration methods on point cloud pairs in eight sequences from the ETH laser registration dataset. Best performance numbers are highlighted in bold fonts.

Method	Apartment		ETH Hauptgebaude		Stairs		Mountains		Gazebo in summer		Gazebo in winter		Wood in summer		Wood in winter	
	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE
ICP	0.04	3.1e2/1.2e2	0.11	49/21	0.07	1.3e2/23	0.08	1.2e2/61	0.10	1.3e2/75	0.10	48/17	0.11	19/13	0.11	19/13
AA-ICP	0.04	3.1e2/1.2e2	0.13	49/21	0.05	1.3e2/24	0.07	1.2e2/62	0.08	1.3e2/75	0.09	48/17	0.12	19/13	0.11	19/13
Ours (Fast ICP)	**0.03**	3.1e2/1.2e2	**0.08**	50/21	**0.04**	1.3e2/24	**0.04**	1.2e2/61	**0.07**	1.3e2/75	**0.07**	49/19	**0.09**	19/13	**0.09**	19/13
Sparse ICP	1.96	68/23	6.63	26/13	2.22	22/12	4.77	12/5.8	4.54	13/7.1	6.84	13/3.6	7.57	5.0/4.4	8.46	4.6/4.1
Ours (Robust ICP)	0.12	51/20	0.31	6.2/5.4	0.12	12/9.8	0.15	6.8/5.5	0.21	7.8/7.2	0.21	3.8/3.4	0.31	4.7/4.2	0.32	**4.0**/3.6
ICP-*l*	0.65	2.9e2/72	0.25	15/7.1	0.52	69/13	0.26	53/27	0.22	1.1e2/36	0.35	44/12	0.30	16/12	0.30	18/14
Sparse ICP-*l*	94.00	94/**8.2**	7.86	4.3/**3.4**	61.31	12/3.0	61.22	**6.0**/5.7	14.50	**6.2**/5.8	24.44	3.5/3.0	7.19	4.8/4.3	54.26	4.6/4.4
Symmetric ICP	0.08	**41**/11	0.20	4.6/4.0	0.10	12/4.6	0.10	7.9/6.9	0.16	7.1/6.2	0.18	4.4/4.5	0.24	7.0/6.4	0.23	7.1/6.9
Ours (Robust ICP-*l*)	0.10	88/8.4	0.20	**4.3**/3.5	0.11	**2.9**/**2.7**	0.11	6.2/**5.3**	0.19	6.4/**5.6**	0.21	**3.5**/**2.9**	0.35	**4.3**/**4.0**	0.34	11/**3.4**

to point-to-plane ICP. The resulting robust ICP schemes similar or better accuracy than the sparse ICP method, while being at least an order of magnitude faster. The methods provide efficient and robust solutions to rigid registration problems where the data may be noisy, contain outliers, and overlap partially.

ACKNOWLEDGMENTS

This work was supported by National Natural Science Foundation of China (No. 61672481), and Youth Innovation Promotion Association CAS (No. 2018495).

REFERENCES

[1] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, 1992.

[2] H. Pottmann, Q. Huang, Y. Yang, and S. Hu, "Geometry and convergence analysis of algorithms for registration of 3d shapes," *International Journal of Computer Vision*, vol. 67, no. 3, pp. 277–296, 2006.

[3] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image and Vision Computing*, vol. 10, no. 3, pp. 145–155, 1992.

[4] H. Pottmann, S. Leopoldseder, and M. Hofer, "Registration without ICP," *Computer Vision and Image Understanding*, vol. 95, no. 1, pp. 54–71, 2004.

[5] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *3rd International Conference on 3D Digital Imaging and Modeling (3DIM 2001), 28 May - 1 June 2001, Quebec City, Canada*, 2001, pp. 145–152.

[6] S. Bouaziz, A. Tagliasacchi, and M. Pauly, "Sparse iterative closest point," *Comput. Graph. Forum*, vol. 32, no. 5, pp. 113–123, 2013.

[7] K. Lange, *MM Optimization Algorithms*. SIAM, 2016.

[8] H. F. Walker and P. Ni, "Anderson acceleration for fixed-point iterations," *SIAM Journal on Numerical Analysis*, vol. 49, no. 4, pp. 1715–1735, 2011.

[9] Y. Peng, B. Deng, J. Zhang, F. Geng, W. Qin, and L. Liu, "Anderson acceleration for geometry optimization and physics simulation," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 42:1–42:14, 2018.

[10] N. J. Mitra and A. Nguyen, "Estimating surface normals in noisy point cloud data," in *Proceedings of the Nineteenth Annual Symposium on Computational Geometry*, ser. SCG '03. New York, NY, USA: ACM, 2003, pp. 322–328.

[11] A. L. Pavlov, G. V. Ovchinnikov, D. Y. Derbyshev, D. Tsetserukou, and I. V. Oseledets, "AA-ICP: iterative closest point with anderson acceleration," in *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, 2018, pp. 1–6.

[12] G. K. L. Tam, Z. Cheng, Y. Lai, F. C. Langbein, Y. Liu, A. D. Marshall, R. R. Martin, X. Sun, and P. L. Rosin, "Registration of 3d point clouds and meshes: A survey from rigid to nonrigid," *IEEE Trans. Vis. Comput. Graph.*, vol. 19, no. 7, pp. 1199–1217, 2013.

[13] A. Tagliasacchi, S. Bouaziz, M. Pauly, and H. Li, "Modern techniques and applications for real-time non-rigid registration," in *SIGGRAPH ASIA 2016, Macao, December 5-8, 2016 - Courses*, 2016, pp. 11:1–11:25.

[14] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP variants on real-world data sets - open-source library and experimental protocol," *Auton. Robots*, vol. 34, no. 3, pp. 133–148, 2013.

[15] S. Rusinkiewicz, "A symmetric objective function for ICP," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 85:1–85:7, 2019.

[16] N. Gelfand, N. J. Mitra, L. J. Guibas, and H. Pottmann, "Robust global registration," in *Proceedings of the Third Eurographics Symposium on Geometry Processing*, ser. SGP '05. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2005.

[17] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3d registration," in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 3212–3217.

[18] D. Aiger, N. J. Mitra, and D. Cohen-Or, "4-points congruent sets for robust pairwise surface registration," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 85:1–85:10, 2008.

[19] N. Mellado, D. Aiger, and N. J. Mitra, "Super 4PCS fast global pointcloud registration via smart indexing," *Computer Graphics Forum*, vol. 33, no. 5, pp. 205–215, 2014.

[20] N. J. Mitra, N. Gelfand, H. Pottmann, and L. Guibas, "Registration of point cloud data from a geometric optimization perspective," in *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, ser. SGP '04. New York, NY, USA: ACM, 2004, pp. 22–31.

[21] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *International Journal of Computer Vision*, vol. 13, no. 2, pp. 119–152, 1994.

[22] D. Chetverikov, D. Stepanov, and P. Krsek, "Robust euclidean alignment of 3d point sets: the trimmed iterative closest point algorithm," *Image and Vision Computing*, vol. 23, no. 3, pp. 299–309, 2005.

[23] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 12, pp. 2262–2275, 2010.

[24] B. Jian and B. C. Vemuri, "Robust point set registration using gaussian mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1633–1645, 2011.

[25] T. Masuda and N. Yokoya, "A robust method for registration and segmentation of multiple range images," *Computer Vision and Image Understanding*, vol. 61, no. 3, pp. 295–307, 1995.

[26] E. Trucco, A. Fusiello, and V. Roberto, "Robust motion and correspondence of noisy 3-d point sets with missing data," *Pattern Recognition Letters*, vol. 20, no. 9, pp. 889–898, 1999.

[27] A. W. Fitzgibbon, "Robust registration of 2d and 3d point sets," *Image Vision Comput.*, vol. 21, no. 13-14, pp. 1145–1153, 2003.

[28] Q. Zhou, J. Park, and V. Koltun, "Fast global registration," in *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II*, 2016, pp. 766–782.

[29] H. Li and R. Hartley, "The 3D-3D registration problem revisited," in *2007 IEEE 11th International Conference on Computer Vision*, 2007, pp. 1–8.

[30] C. Olsson, F. Kahl, and M. Oskarsson, "Branch-and-bound methods for euclidean registration problems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 783–794, 2009.

[31] J. Yang, H. Li, D. Campbell, and Y. Jia, "Go-ICP: A globally optimal solution to 3d ICP point-set registration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 11, pp. 2241–2254, 2016.

[32] D. G. Anderson, "Iterative procedures for nonlinear integral equations," *J. ACM*, vol. 12, no. 4, pp. 547–560, 1965.

[33] V. Eyert, "A comparative study on methods for convergence acceleration of iterative vector sequences," *Journal of Computational Physics*, vol. 124, no. 2, pp. 271–285, 1996.

[34] H.-r. Fang and Y. Saad, "Two classes of multisecant methods for nonlinear acceleration," *Numerical Linear Algebra with Applications*, vol. 16, no. 3, pp. 197–221, 2009.

[35] A. Toth and C. T. Kelley, "Convergence analysis for anderson acceleration," *SIAM Journal on Numerical Analysis*, vol. 53, no. 2, pp. 805–819, 2015.

[36] H. D. Sterck, "A nonlinear gmres optimization algorithm for canonical tensor decomposition," *SIAM Journal on Scientific Computing*, vol. 34, no. 3, pp. A1351–A1379, 2012.

[37] K. Lipnikov, D. Svyatskiy, and Y. Vassilevski, "Anderson acceleration for nonlinear finite volume scheme for advection-diffusion problems," *SIAM Journal on Scientific Computing*, vol. 35, no. 2, pp. A1120–A1136, 2013.

[38] P. P. Pratapa, P. Suryanarayana, and J. E. Pask, "Anderson acceleration of the jacobi iterative method: An efficient alternative to Krylov methods for large, sparse linear systems," *Journal of Computational Physics*, vol. 306, pp. 43–54, 2016.

[39] N. Ho, S. D. Olson, and H. F. Walker, "Accelerating the Uzawa algorithm," *SIAM Journal on Scientific Computing*, vol. 39, no. 5, pp. S461–S476, 2017.

[40] P. Suryanarayana, P. P. Pratapa, and J. E. Pask, "Alternating anderson-richardson method: An efficient alternative to preconditioned krylov methods for large, sparse linear systems," *Computer Physics Communications*, vol. 234, pp. 278–285, 2019.

[41] J. Zhang, Y. Peng, W. Ouyang, and B. Deng, "Accelerating admm for efficient simulation and optimization," *ACM Trans. Graph.*, vol. 38, no. 6, pp. 163:1–163:21, 2019.

[42] F. A. Potra and H. Engler, "A characterization of the behavior of the anderson acceleration on linear problems," *Linear Algebra and its Applications*, vol. 438, no. 3, pp. 1002–1011, 2013.

[43] O. Sorkine-Hornung and M. Rabinovich. (2017) Least-squares rigid motion using svd. [Online]. Available: https://igl.ethz.ch/projects/ARAP/svd_rot.pdf
[44] K. Lange, *Optimization*. Springer New York, 2004, ch. The MM Algorithm, pp. 119–136.
[45] C. Evans, S. Pollock, L. G. Rebholz, and M. Xiao, "A proof that Anderson acceleration improves the convergence rate in linearly converging fixed-point methods (but not in those converging quadratically)," *SIAM J. Numer. Anal.*, vol. 58, no. 1, pp. 788–810, 2020.
[46] J. Diebel, "Representing attitude: Euler angles, unit quaternions, and rotation vectors," 2006.
[47] D. K. Hoffman, R. C. Raffenetti, and K. Ruedenberg, "Generalization of euler angles to ndimensional orthogonal matrices," *Journal of Mathematical Physics*, vol. 13, no. 4, pp. 528–533, 1972.
[48] V. S. Varadarajan, *Lie groups, Lie algebras, and their representations*, ser. Graduate Texts in Mathematics. New York: Springer-Verlag, 1984, vol. 102.
[49] J. Gallier and D. Xu, "Computing exponentials of skew symmetric matrices and logarithms of orthogonal matrices," *International Journal of Robotics and Automation*, vol. 18, no. 1, pp. 10–20, 2002.
[50] A. Fu, J. Zhang, and S. Boyd, "Anderson accelerated Douglas-Rachford splitting," *arXiv preprint arXiv:1908.11482*, 2019.
[51] M. Alexa, "Linear combination of transformations," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 380–387, 2002.
[52] P. W. Holland and R. E. Welsch, "Robust regression using iteratively reweighted least-squares," *Communications in Statistics - Theory and Methods*, vol. 6, no. 9, pp. 813–827, 1977.
[53] B. Ham, M. Cho, and J. Ponce, "Robust guided image filtering using nonconvex potentials," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 1, pp. 192–207, 2018.
[54] J. Zhang, B. Deng, Y. Hong, Y. Peng, W. Qin, and L. Liu, "Static/dynamic filtering for mesh geometry," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 4, pp. 1774–1787, 2019.
[55] R. Chartrand and W. Yin, "Iteratively reweighted algorithms for compressive sensing," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2008, pp. 3869–3872.
[56] P. Ochs, A. Dosovitskiy, T. Brox, and T. Pock, "On iteratively reweighted algorithms for nonsmooth nonconvex optimization in computer vision," *SIAM Journal on Imaging Sciences*, vol. 8, no. 1, pp. 331–372, 2015.
[57] Y. Wang, W. Yin, and J. Zeng, "Global convergence of ADMM in nonconvex nonsmooth optimization," *Journal of Scientific Computing*, vol. 78, no. 1, pp. 29–63, 2019.
[58] F. Pomerleau, M. Liu, F. Colas, and R. Siegwart, "Challenging data sets for point cloud registration algorithms," *The International Journal of Robotics Research*, vol. 31, no. 14, pp. 1705–1711, 2012.
[59] G. Guennebaud, B. Jacob *et al.*, "Eigen v3," http://eigen.tuxfamily.org, 2010.
[60] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12, 2012*, 2012, pp. 573–580.
[61] J. Vongkulbhisal, B. I. Ugalde, F. D. la Torre, and J. P. Costeira, "Inverse composition discriminative optimization for point cloud registration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2993–3001.
[62] N. J. Higham, *Functions of Matrices: Theory and Computation*. Society for Industrial and Applied Mathematics, 2008.
[63] G. H. Golub and C. F. V. Loan, *Matrix computions*. The Johns Hopkins University Press, 1983.
[64] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge university press, 2012.

APPENDIX A
COMPUTING MATRIX LOGARITHMS

To compute matrix logarithms, Existing numerical methods such as the *inverse scaling and squaring method* [62] requires the matrix to have no negative eigenvalues, which may not hold for the transformation matrices considered in this paper. In the following, we derive a numerical method for computing logarithms of transformation matrices without such restrictions.

Given a transformation matrix

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}$$

where $\mathbf{R}$ is a rotation matrix, it can be shown that its real Schur decomposition has the following form [63], [64]:

$$\mathbf{T} = \mathbf{Q}\mathbf{U}\mathbf{Q}^T = \mathbf{Q}\begin{bmatrix} \mathbf{U}_{11} & \mathbf{U}_{12} & \mathbf{U}_{13} & \dots & \mathbf{U}_{1m} \\ \mathbf{0} & \mathbf{U}_{22} & \mathbf{U}_{23} & \dots & \mathbf{U}_{2m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{U}_{mm} \end{bmatrix}\mathbf{Q}^T,$$

where $\mathbf{Q} \in \mathbb{R}^{(n+1)\times(n+1)}$ is an orthogonal matrix, and each diagonal block $\mathbf{U}_{ii}$ is either a 1-by-1 matrix or a 2-by-2 matrix. Due to the special form of $\mathbf{T}$, we can permute the rows and columns of $\mathbf{U}$ as well as the columns of $\mathbf{Q}$ to obtain the following decomposition:

$$\mathbf{T} = \mathbf{Q}'\mathbf{U}'{\mathbf{Q}'}^T \tag{28}$$

where

$$\mathbf{Q}' = \begin{bmatrix} \mathbf{Q}_1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{U}' = \begin{bmatrix} \mathbf{D} & \mathbf{y} \\ \mathbf{0} & 1 \end{bmatrix},$$

and $\mathbf{D}$ is a block diagonal matrix [64]:

$$\mathbf{D} = \begin{bmatrix} \mathbf{D}_1 & & & \\ & \ddots & & \\ & & \mathbf{D}_p & \\ & & & \mathbf{I}_{n-2p} \end{bmatrix}.$$

Here $\mathbf{D}_i$ is 2-by-2 rotation matrix and can be written as

$$\mathbf{D}_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{bmatrix} \tag{29}$$

with $\theta \in [0, \pi]$. Using the decomposition (28), the logarithm of $\mathbf{T}$ can be computed as [63], [64]:

$$\log(\mathbf{T}) = \log(\mathbf{Q}'\mathbf{U}'{\mathbf{Q}'}^T) = \mathbf{Q}'\log(\mathbf{U}'){\mathbf{Q}'}^T.$$

To compute $\log(\mathbf{U}')$, we first note that the rotation angle θ_i in (29) can be determined from the entries of $\mathbf{D}_i$. We can then calculate the logarithm of $\mathbf{D}_i$ as

$$\mathbf{B}_i = \log(\mathbf{D}_i) = \begin{bmatrix} 0 & -\theta_i \\ \theta_i & 0 \end{bmatrix}, \tag{30}$$

Then we compute $\log(\mathbf{U}')$ according to [49] as

$$\log(\mathbf{U}') = \log\begin{bmatrix} \mathbf{D} & \mathbf{y} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{B} & \mathbf{V}\mathbf{y} \\ \mathbf{0} & 0 \end{bmatrix}, \tag{31}$$

where

$$\mathbf{B} = \log(\mathbf{D}) = \begin{bmatrix} \mathbf{B}_1 & & & \\ & \ddots & & \\ & & \mathbf{B}_p & \\ & & & \mathbf{0}_{n-2p} \end{bmatrix}, \tag{32}$$

and

$$\mathbf{V} = \mathbf{I}_n + \sum_{i=1}^{p}\left(-\frac{\theta_i}{2}\mathbf{B}_i + (1 - \frac{\theta_i\sin\theta_i}{2(1-\cos\theta_i)})\mathbf{B}_i^2\right). \tag{33}$$

Algorithm 3 provides the psuedo-code for computing $\log(\mathbf{T})$.

Algorithm 3: Logarithm for matrices in $SE(d)$.

Input: Transform matrix $\mathbf{T} \in SE(d)$.
Output: $\hat{\mathbf{T}} = \log(\mathbf{T})$.

1 Compute the real Schur decomposition $(\mathbf{Q}, \mathbf{U})$ of $\mathbf{T}$;
2 Rearrange $\mathbf{Q}, \mathbf{U}$ to obtain matrices $\mathbf{Q}', \mathbf{U}'$ in Eq. (28);
3 Calculate $\log(\mathbf{U}')$ according to Eqs. (30)–(33);
4 $\hat{\mathbf{T}} = \mathbf{Q}' \log(\mathbf{U}') \mathbf{Q}'^T$;

APPENDIX B
SURROGATE FUNCTION FOR EQ. (15)

In this section, we show that the function in Eq. (17) is a surrogate function of the target function in Eq. (15) at the current transformation $\mathbf{R}^{(k)}, \mathbf{t}^{(k)}$. To simplify notation, we denote $D_i = D_i(\mathbf{R}, \mathbf{t})$ and $D_i^{(k)} = D_i(\mathbf{R}^{(k)}, \mathbf{t}^{(k)})$. By definition, since $\chi_\nu(D_i \mid D_i^{(k)})$ is a surrogate function for $\psi_\nu(D_i)$ at $D_i^{(k)}$, we have

$$\begin{aligned} \psi_\nu(D_i^{(k)}) &= \chi_\nu(D_i^{(k)} \mid D_i^{(k)}), \\ \psi_\nu(D_i) &\le \chi_\nu(D_i \mid D_i^{(k)}), \quad \forall\, D_i \neq D_i^{(k)}. \end{aligned} \tag{34}$$

Note that the function $x_i(\mathbf{R}, \mathbf{t}) = \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \hat{\mathbf{q}}_i^{(k)}\|$ satisfies

$$D_i \le x_i(\mathbf{R}, \mathbf{t}), \quad D_i^{(k)} = x_i(\mathbf{R}^{(k)}, \mathbf{t}^{(k)}). \tag{35}$$

Moreover, $\chi_\nu(x \mid y)$ is a monotonically increasing function on $x \in [0, +\infty)$, which together with Eqs. (34) and (35) means that

$$\begin{aligned} \psi_\nu(D_i^{(k)}) = \chi_\nu(D_i^{(k)} \mid D_i^{(k)}) &= \chi_\nu(x_i(\mathbf{R}^{(k)}, \mathbf{t}^{(k)}) \mid D_i^{(k)}), \\ \psi_\nu(D_i) \le \chi_\nu(D_i \mid D_i^{(k)}) &\le \chi_\nu(x_i(\mathbf{R}, \mathbf{t}) \mid D_i^{(k)}). \end{aligned}$$

It shows that $\chi_\nu(x_i(\mathbf{R}, \mathbf{t}) \mid D_i^{(k)})$ is a surrogate function for $\psi_\nu(D_i)$ at $D_i^{(k)}$. Then substituting each term $\psi_\nu(D_i)$ in Eq. (15) with $\chi_\nu(x_i(\mathbf{R}, \mathbf{t}) \mid D_i^{(k)})$, we can see that Eq. (17) is a surrogate function at $(\mathbf{R}^{(k)}, \mathbf{t}^{(k)})$.

APPENDIX C
CHOICES OF $\nu_{\min}$

In this section, we explain the rationale for our choices of $\nu_{\min}$ for our robust ICP methods.

For the point-to-point method, our intention is to set $\nu_{\min}$ large enough such that point pairs with deviation due to difference in sampling locations will be included in the alignment step. For ease of discussion, we first assume that the target set has uniform sampling density, and the source set is sampled from a triangulated surface using the target set as vertices. Then within the overlapping region the distance from a point in $\mathbf{p} \in P$ to the set Q can be up to $E/\sqrt{3}$ where E is the distance between neighboring points within Q (e.g., when $\mathbf{p}$ lies at the center of an equilateral triangle T with edge length E from the triangulation of Q). In order to include $\mathbf{p}$ and its closest point into the alignment step, $\nu_{\min}$ should be no smaller than $1/3$ of the distance between them due to the three-sigma rule, i.e., $\nu_{\min} \ge E/(3\sqrt{3})$. In practice, the sampling density of Q may not be uniform. Therefore, we compute the representative distance $\overline{E}_Q$ between neighboring points in Q as explained in Section 4.3, and set $\nu_{\min} = \overline{E}_Q/(3\sqrt{3})$.

For the point-to-plane method, we first use the same assumption as the point-to-point method: the target set has uniform sampling density, and the source set is sampled from a surface triangulated from the target set. For a source point $\mathbf{p} \in P$, suppose its closest point in Q is $\mathbf{q}$. Then for

Such a point should be included into the alignment process. Recall that a point $\mathbf{p}_i \in P$ will be effectively excluded from the alignment step if the distance between its current transformed position and the point set Q is larger than 3ν. Therefore, to ensure the points in the overlapping region are included for alignment, $\nu_{\min}$ should be no smaller than $\frac{E}{3\sqrt{3}}$. In reality, the sampling density of the point sets may not be uniform, thus we adapt the above heuristics as follows. We first compute the median distance from each point $\mathbf{p}_i \in P$ to its six nearest neighbors in P, and take the median $\overline{E}_P$ of these median distance values across P. In the same way, we compute a value $\overline{E}_Q$ for the set Q. Let $\mathbf{s} \in Q$ be the neighbor point of $\mathbf{q}$ that is the farthest away from the tangent plane at $\mathbf{q}$. Denote by $H_\mathbf{q}(\cdot)$ the distance to the tangent plane at $\mathbf{q}$. Then we must have $H_\mathbf{q}(\mathbf{p}) \le \frac{1}{2} H_\mathbf{q}(\mathbf{s})$, otherwise $\mathbf{q}$ would not be the closest point to $\mathbf{p}$ in Q. Then order to include the pair $(\mathbf{p}, \mathbf{q})$ into the alignment step, we can set $\nu_{\min} \ge \frac{1}{6} H_\mathbf{q}(\mathbf{s}) \ge \frac{1}{3} H_\mathbf{q}(\mathbf{p})$. To handle non-uniform sampling of Q, we compute the representative distance $\overline{H}_Q$ to a neighboring point's tangent plane in Q, and set $\nu_{\min} = \overline{H}_Q/6$.

Juyong Zhang is an associate professor in the School of Mathematical Sciences at University of Science and Technology of China. He received the BS degree from the University of Science and Technology of China in 2006, and the PhD degree from Nanyang Technological University, Singapore. His research interests include computer graphics, computer vision, and numerical optimization. He is an associate editor of The Visual Computer.

Yuxin Yao is currently working toward the master's degree in the School of Mathematical Sciences, University of Science and Technology of China. Her research interests include computer graphics and 3D registration.

Bailin Deng is a lecturer in the School of Computer Science and Informatics at Cardiff University. He received the BEng degree in computer software (2005) and the MSc degree in computer science (2008) from Tsinghua University (China), and the PhD degree in technical mathematics (2011) from Vienna University of Technology (Austria). His research interests include geometry processing, numerical optimization, computational design, and digital fabrication. He is a member of the IEEE.